A NOVEL PRACTICE-BASED PROCESS MODEL FOR SECURE AGILE SOFTWARE DEVELOPMENT

Abdulhamid Aliyu Ardo

SCHOOL OF SCIENCE, ENGINEERING AND ENVIRONMENT

MARCH 15, 2024

This document is toward the degree of Doctor of Philosophy

Table of Contents

List of Figures

List of Tables

Dedication

I dedicate this PhD research to all those who died or were displaced in Nigeria, due to kidnapping, banditry, and other forms of criminality; and to all those that lost their lives during the Covid-19 pandemic.

Acknowledgement

So, which of the favours of your Lord would you deny? Qur'an Surah Ar-Rahman 55:13

I wish to thank God Almighty the Lord of the universe for sparing my life and keeping me healthy upto this moment. My sincere gratitude goes to my parents; Alhaji Aliyu Ardo Jada and Hajiya Bilkisu Aliyu Jada for their constant prayers towards the successful completion of this research. To my best friend and darling wife, Fatima, and our lovely children Bilkisu (Samha) and Aliyu (Sabir), I say thank you for being my number one fans always. My siblings Aminu, Aisha, Abubakar and Fadimatu have been great pillars of support throughout the journey. There are other family and friends who have supported but too numerous to mention, I say May Allah reward you all.

I am extremely indebted to my supervisory team, Professor Julian M. Bass and Dr. Tarek Gaber for the encouragement, guidance, and constructive criticisms to ensure they bring the best out of me. Your continuous support outside the research space have sharpen my skills and provided a boost to my employability.

Finally, my profound gratitude goes to the Nigerian government through the Petroleum Technology Development Fund (PTDF) for sponsoring me without which the PhD research would not have been possible. I am also thankful to all the practitioners that contributed to this research, I hold you all in high esteem.

THANK YOU

DECLARATION OF ORIGINALITY

I, Abdulhamid Aliyu Ardo, hereby unequivocally declare that this PhD thesis is based on my original work except where acknowledgements indicate otherwise. All information sources, ideas and quotation used from interviewed practitioners and other people's research have been properly acknowledged.

To the best of my knowledge, this report has not been submitted to University of Salford or any other institution to the award of an academic degree. This thesis also does not infringe on anyone's copyright. I further certify that I have followed the guidance for conducting PhD research as prescribed by University of Salford rules and regulations under the guidance of my supervisors.

ABSTRACT

Nigeria is ranked second globally after India in reported incidences of cyberattacks. Attackers usually exploit vulnerabilities in software which may not have considered security features during the development process. Agile methodologies are a well-established paradigm in the software development field. Its adoption has contributed to improving software quality. However, agile software products remain vulnerable to security challenges and susceptible to cyberattacks. Agile methods also tend to neglect non-functional requirements such as security. Despite its significance, there is paucity of research addressing security. The problem tackled in this research is the lack of security practices integration in agile software development. Thus, this thesis aims to improve security of the software development process when using agile methods through the developed secure process model.

The methodology arising from the research context is a multi-methods qualitative approach divided into four phases involving 35 practitioners from 17 organisations. The first phase describes an exploratory case study conducted to empirically explore the agile security practices adopted by software developers and security professionals in United Kingdom (UK). The second phase involves conducting semi-structured interviews to investigate the impact of regulatory policy for building secure agile software in Nigeria. The third phase developed a novel practice-based agile software development process model derived from the results of the interview data analysis conducted. Finally, the model was preliminarily validated through a focus group comprising of 5 senior agile cybersecurity professionals to evaluate its relevancy and novelty. The focus group was conducted online, comprising predominantly UK practitioners previously interviewed, along with a few participants who were not involved in the earlier stages of data collection. The model was also applied at a Nigerian company involved in secure agile software development.

Using the adopted methodology, this thesis presents a taxonomy of security practices identified in the UK research sites. They were categorized according to agile use in organisation - roles, ceremonies, and artefacts. Based on the analysis of interviews conducted in Nigeria, a grounded theory of the security challenges confronting agile practitioners was also developed which was termed Policy Adherence Challenges (PAC) model. The four challenges identified are: (a) a lack of collaboration between security and agile teams; (b) the tendency to use foreign software hosting companies; (c) a poor cybersecurity culture; and (d) the high cost of building secure agile software. Also, the model developed in this thesis used swim lane diagrams to highlight the process flow of security activities. 24 security practices were identified and organized into a process flow. The practices were mapped onto five swim lanes each representing an agile role. The preliminary model evaluation conducted through a focus group workshop proposed a new practice, in response to an observed lack of collaborative ceremonies, to disseminate awareness of and hence compliance with security standards. Further evaluation of the secure process model led to several positive changes in the chosen organisation. These include enhanced collaboration through introducing security retrospectives sessions, intervention to reduce manager's work tasks by introducing a security champion role, action to enhance team security competence by reducing collaborative gap with senior roles which form mitigation mechanisms to improve regulatory compliance in the global south context. This research recommends practitioners integrate practices such as the proposed "compliance sprint" to improve the security of their products thereby reducing the incidences of cyberattacks. Also, there is need for government action by creating the enabling environment to ensure compliance to regulatory policies and security standards for practitioners developing secure software products.

Chapter 1

Introduction

1.1 Problem Context

The increased reliance of people and organizations on software products and services can be attributed to advances in technology and greater connectivity (Valdés-Rodríguez et al., 2023). Despite its benefits, the security threats posed by attackers has been on the increase. The 2022 European Union Agency for Cybersecurity (ENISA) annual threat landscape report revealed that changes in working conditions and dependence on software in the aftermath of COVID-19 pandemic has further exacerbated the rate of cyberattacks (ENISA, 2022). The introduction of stricter regulatory policies and growing fines is another motivation for organizations to focus more attention on cybersecurity issues (Breaux & Antón, 2008; Moyón et al., 2020). Software design and implementation needs to consider security to ensure systems continue to function as intended and quickly recover in the event of an attack. As cyberattacks are becoming common occurrence in organisations, there is an increasing need to adopt proactive measures. One of such measures is integrating security practices into the development process known as secure-by-design methodology. It is a systematic method which integrates security practices across the entire software development lifecycle (Casola et al., 2020). Cybersecurity is considered an important aspect of the software development process due to increased connectivity. Thus, a better organizational security culture reduces the challenges faced in security practices adoption and integration in the development process (Aalvik et al., 2023).
Building secure software by integrating security practices into the development process can help mitigate cyberattacks (McGraw, 2006). However, agile principles conflict with security requirements as processes can be unpredictable at the beginning of a project (Rindell et al., 2021). This creates problems in the development process such as increase in time to perform security tests and additional documentation. The lack of integrating security practices has been reported in existing studies as a challenge in agile software development (Keramati & Mirian-Hosseinabadi, 2008; Khaim et al., 2016; Rindell et al., 2021; Siponen et al., 2005). Security

practices are defined as activities that invariably enhance software security (Nägele et al., 2022b). Some of these activities include threat modelling (Bernsmed et al., 2022), penetration testing (Arkin et al., 2005) and code reviews (Sadowski et al., 2018). Many researchers agree that integrating security practices in agile software development have become necessary to support the identification of security risks and threats (Moyón et al., 2020; Newton et al., 2019). Tøndel et al. (2022) proposed a model of influencing factors for security prioritisation in agile software development. The study found a collaborative gap between agile and security team which was influence by unclear security requirements and lack of security practices integration. While the integration of security practices in agile software development is crucial, there is a limited number of empirical studies in the current literature (Nägele et al., 2022b; van der Heijden et al., 2018). Furthermore, even the few existing ones are specifically focused on the global north context.

Global south countries, particularly those in Africa, have lagged developed nations in adopting and utilizing technology. Despite the benefits of software systems, security issues continue to be a concern in both large and small organizations. For example, the threat from attackers, spammers, and criminal corporations. Developing countries are particularly vulnerable to these threats, and more so for small and medium-sized enterprises (SMEs) who tend to have limited resources to acquire and implement cybersecurity mechanisms in their organizations (Sulayman et al., 2012). Cybercrime in companies has caused significant financial losses, software malfunctions, destruction of critical information systems. It would be beneficial for companies to understand security practices as it can significantly influence the adoption of a secure organizational culture.

In agile software development, adherence to regulatory policies ensures the mitigation of cybersecurity risks and threats. Regulatory compliance is described as the act of adhering to standards, policies, and procedures by organizations (Akhigbe et al., 2019). Software development is driven by the needs of various stakeholders formulated as requirements. The stakeholders include software users and governments responsible for issuing regulations and policies. One of the challenges when employing agile software development approaches is the issue of ensuring compliance with security standards and legislations (Moyón et al., 2020). Existing studies have explored compliance and security in agile software development separately. This makes security compliance cumbersome in agile software development as security is not integrated into the process. Oueslati et al. (2016) explored the challenges of secure agile software development, however, compliance to security standards and regulations were not in the study scope. Villamizar et al. (2018) investigated the lack of security practices integration in agile software development. The study introduced new guidelines for handling security issues. The study, however, did not report how security compliance can be achieved through the introduced guidelines. Although there are existing studies on security compliance in agile software development, empirical evidence on practitioner's adherence is still lacking (Usman et al., 2020).

1.2 Research Motivation

The integration of security practices and compliance to regulatory policies and standards is becoming an important concern in agile software development. In practice, software developers usually adopt reactive rather than a proactive approach by focusing on security at later stages. This can be a costly approach especially in situations when security flaws needs to be fixed (Valdés-Rodríguez et al., 2023). According to a global risk management survey by AON, the biggest challenges to organizations are cyberattacks, data/software breaches and reputational damage (Nägele et al., 2022b). Villamizar et al. (2020) in their work reported the challenges at the requirements specification stage which was attributed to lack of expertise and misunderstood security needs. Leite et al. (2021) investigated the impact of General Data Protection Regulation (GDPR) with specific focus on traditional software development. The study by Bodden (2018), highlighted the root cause of insecure software as the lack of empirical research to understand why security flaws arise in the software development lifecycle. Several authors agree to the need for more empirical research in the domain of secure agile software development (Moyón et al., 2020; Nägele et al., 2022b; Rindell et al., 2021; Sharma & Bawa, 2020; Valdés-Rodríguez et al., 2023; van der Heijden et al., 2018).

The forgoing motivated this research to explore the current state of agile security practices implementation, investigate factors impeding compliance to regulatory policies and standards, and develop a model for secure agile software development process. The dependence on software coupled with the rise in Internet of Things (IoTs) adoption increases connectivity. However, this increases vulnerabilities and possibilities of cyberattacks. This research aims to contribute to the need for more empirical studies on security practices integration and compliance to regulatory standards in agile software development to reduce cyberattacks.

1.3 Aim, Objectives, and Research Questions

1.3.1 Aim

This research aims to develop a secure agile software process by investigating the state of security practices integration in organizations to reduce the incidences of cyberattacks. In this research, the phrase "development process" refers to the activities undertaken across the entire software development lifecycle phases to create a secure software product.

1.3.2 Objectives

This study attempts to investigate the security practices adopted by agile teams and the factors influencing adherence to regulatory policies and standards. To achieve the aim of this study, the following research objectives were formulated:

Objective 1: Describe the practices used, in companies to improve the security of agile software development process.

Objective 2: Investigate the factors that influence adherence to regulatory policies and standards in organizations building secure software using agile methods.

Objective 3: Develop and evaluate secure agile software development process by implementing selected practices in an organisation to positively influence company processes.

1.3.3 Research Questions

This study chose an investigation into the security practices adopted by organizations as a general area of interest. Four research questions emerged from the investigation which were answered in this study. To achieve the objectives listed in the previous section, the following questions were designed:

Research Question 1: How do selected practitioners describe the current state of agile security practices implementation for software development?

Research Question 2: What are the impacts of regulatory policies and standards for developing secure software using agile methods in the case study companies investigated?

Research Question 3: How can the identified security practices in the case study companies be integrated to create an agile process model that suits software development needs and organisational context?

Research Question 4: What are the selected practitioners' perceptions of the effect of implementing a secure agile process model on improved software development process?

1.4 Research Design

To achieve the aim and objectives of this research, a qualitative multi-methods design was adopted. It consisted of two-studies: an exploratory case study in UK and an in-depth study in Nigeria with practitioners engaged in secure software development using agile methods. Data was collected through semi-structured interviews and analysed using a method informed by grounded theory (Hoda et al., 2012; Stray et al., 2016). Thus, the aim and objectives of my research are identified in the form of four (4) phases to answer the research questions as discussed in this section.

Phase 1: Exploratory case study involved conducting interviews to explore practitioners' perceptions of agile security practices implementation (RQ1). This phase is presented in chapter 4.

Phase 2: In-depth study by collecting data through semi-structured interviews to determine the impacts of regulatory policy for building secure agile software in Nigeria as described by practitioners (RQ2). This phase is presented in chapter 5.

Phase 3: Analysis of interview transcripts from the first two phases of this research which led to the development of a novel practice-based process model (RQ3). This phase is presented in chapter 6.

Phase 4: Evaluating and updating the developed process model through a focus group workshop in the first instance. Parts of the model was subsequently implemented in a company to critically examine the impacts of security practices implementation on organizational processes (RQ4). This phase is presented in chapter 7.

1.5 Research Contributions

This research provides an in-depth study of the practices adopted by companies for building secure software using agile methods. Also, the factors influencing non-adherence to regulatory policy were explored. The specific contributions to knowledge and practice are as follows:

1. An empirical taxonomy of security practices categorized into agile roles, ceremonies, and artefacts. The classification identifies the practices adopted for improving security of the development process to reduce incidences of cyberattacks.

2. The study developed a Grounded Theory (GT) of the security challenges confronting agile practitioners. The GT identified factors influencing non-adherence to legislations through the Policy Adherence Challenges (PAC) model presented.

3. A practiced-based process model based on swim lane diagram notation.

4. The model proposed a new practice termed "compliance sprint" due to the observed lack of collaborative ceremonies, to disseminate awareness on security knowledge, security standards and regulatory policy. The model evaluation also led to the introduction of the security champion role which was aimed at improving organisational security culture and reducing collaborative gap between practitioners.

1.6 List of Publications

The following papers were published based on the findings of the PhD research:

- **Ardo, A. A.,** Bass, J. M., & Gaber, T. (2023). Implications of regulatory policy for building secure agile software in Nigeria: A grounded theory. *The Electronic Journal of Information Systems in Developing Countries, (EJISDC)* (Ardo et al., 2023).

- **Ardo, A. A.,** Bass, J. M., & Gaber, T. (2022). Towards Secure Agile Software Development Process: A Practice-Based Model. In *2022 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)* (Ardo et al., 2022).

- **Ardo, A. A.,** Bass, J. M., & Gaber, T. (2021). An empirical investigation of agile information systems development for cybersecurity. In *Information Systems: 18th European, Mediterranean, and Middle Eastern Conference on Information Systems (EMCIS)* (Ardo et al., 2021).

- Rahy, S., Kreps, D., Bass, J. M., Gaber, T., & **Ardo, A.** (2020). A post-colonial analysis of agile software development methods in ICT4D. In *Information and Communication Technologies for Development: 16th IFIP WG 9.4 International Conference on Social Implications of Computers in Developing Countries* (Rahy et al., 2020).

Blog

- **Ardo, A.A.,** Bass, J.M., Gaber, T. (2022). Secure by Design. [online], Available from: https://www.salford.ac.uk/working-with-business/greater-manchester-cyber-foundry/secure-by-design

1.7 Thesis Structure

This thesis consists of nine chapters organized as follows:

Chapter One – Introduction discusses the problem context and motivation for the research. The aim and objectives of conducting the research together with brief description of the methodological design and contributions are presented.

Chapter Two – Literature review provided a detailed description of agile concepts and related work in the domain of secure agile software development. Then, published studies on agile methods implementation in developing countries and the challenges of compliance to security standards and regulatory policies were discussed.

Chapter Three – Research design discussed the methodological choices adopted for the research. The chapter also provides the philosophical assumptions and perspectives underpinning the research as well as justification for the research approach. The data collection techniques and analysis methods are presented. The research model development and validation process are also discussed.

Chapter Four – The findings of the exploratory case study conducted with agile practitioners in UK companies were analysed and critically examined. Using a method informed by grounded theory, the security practices adopted by practitioners were identified. The analysis produced part of the contribution of this thesis in the form of a taxonomy of agile security practices which has been published in the 18th European, Mediterranean, and Middle Eastern Conference on Information Systems (Ardo et al., 2021).

Chapter Five – Analysis of the data collected in phase 2 of the research which investigated the factors influencing non-adherence to regulatory policies and software security standards were identified through practitioner interviews. It provided a detailed view on secure agile software development challenges in Nigeria. Like the preceding chapter, this one also consolidates the thesis contributions drawing from the paper published in Electronic Journal of Information Systems in Developing Countries (Ardo et al., 2023).

Chapter Six – Secure software model development process is presented. The mapping relationship from memos presented in the two preceding chapters to elements of the model are illustrated (Ardo et al., 2022).

Chapter Seven – Model evaluation discussed the focus group model validation and the intervention process at the chosen research site in Nigeria. Findings from the intervention process are presented. Different versions of the refined model are presented. Finally, findings from post-implementation interviews were analysed. The preliminary evaluation of the practised-based model developed in the preceding chapter has been published in the 48th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) (Ardo et al., 2022)

Chapter Eight – Discussion critically analysed the study findings and answers the research questions. The chapter brings together all the theoretical, and practical contributions that filled the gap in knowledge identified. The chapter also discussed the research limitations.

Chapter Nine – Conclusion summarizes the thesis, presents reflection on the entire PhD thesis, and provides direction for future research.

Chapter 2

Literature Review

2.1 Introduction

In chapter one, the research context and rationale were discussed. The research aim and objectives were also outlined. This chapter has two main sections: Part one presents an overview of software development methodologies and agile methods adoption in developing countries where there is paucity of research. Part two provides a critical analysis of existing studies related to the research aim and objectives. The related works includes studies on secure software development models and secure agile concepts. An in-depth discussion on security practices adopted by organisations is presented. This chapter further explored security compliance in agile software development highlighting limitations and gaps. The overall aim of this chapter is to provide direction justifying the need for integrating security practices into the agile software development process. Finally, the research gap is discussed.

2.2 Software Development Methodologies

There are various software methodologies proposed in literature to provide structure to the development process activities. Generally, the methodologies are divided into three groups: traditional methods, component based and iterative/incremental development (Robey et al., 2001b). Some of the traditional life-cycle methodologies include the waterfall model (Royce, 1987), and the spiral model (Boehm, 1986). The component-based method focuses on development of system parts as re-usable entities (Crnkovic, 2001). While the iterative/incremental is focused on continuous customer involvement which agile methods follow (Taylor et al., 2006). Since this research focuses on secure agile process, an in-depth discussion of the concept is provided in this chapter.

2.2.1 Agile Software Development

Agile Software Development is an iterative and incremental approach where self-organizing teams adjust to meet customer changing requirements. It is based on the principles of the Agile Manifesto which was collaboratively written by a group of software developers (Fowler & Highsmith, 2001). The key value behind the agile manifesto was quick delivery of working software. The emergence of the agile manifesto has brought unprecedented changes to the software engineering research domain. The changes to the software industry were specifically in the areas of how software developers plan, coordinate, and communicate with customers and other stakeholders (Dingsøyr et al., 2010).

Agile software development methods were developed to help solve the inherent problems observed in the traditional development methods (Cohen et al., 2004; Robey et al., 2001a). Some of these problems include inflexibility and rigidity to introduced changes, high delivery cost and time consuming, wastages and huge documentations, lack of team collaboration, lack of customer involvement and top management bottlenecks (Pikkarainen et al., 2008). To improve software development process, the above enumerated problems need to be solved. Thus, agile values are focused on customer satisfaction, embracing change, delivery of high-quality software and light-weight documentation (Cohen et al., 2004).

Several methods adhering to the agile manifesto have been introduced (Dingsøyr et al., 2010; Dingsøyr et al., 2012; Dybå & Dingsøyr, 2008). The earliest developed method is the Dynamic Software Development Method (Abrahamsson et al., 2017; Dybå & Dingsøyr, 2008; Stapleton, 1997), a group of methods having concepts that can be customized to projects. The most common and widely adopted agile methods include Scrum which focus on project management (Cohn, 2010; Schwaber, 2004), eXtreme Programming which is focused on activities at the development team level (Beck, 1999, 2000) and Lean software development method which focus on eliminating wastages (Poppendieck & Poppendieck, 2003, 2009). Other methods include the Feature Driven Development (Palmer & Felsing, 2001), which is based on features to be completed by a group to increase efficiency and Adaptive Software Development (Abrahamsson et al., 2017; Highsmith, 2013), which is based on project concepts and culture. Out of the several agile methodologies, the three more popular one's are scrum, extreme programming and lean.

Scrum

The Scrum method was the initial idea of a software developer at Easel Corporation named Jeff Sutherland in the early 1990s. Working together with Ken Schwaber, they refined and formalized the method (Schwaber, 2004). Scrum involves working in sprint cycles to deliver software products typically within a predefined time-box of (usually between two to four weeks) (Rubin, 2012). It is an empirical approach which applies the principles of industrial control to system design and development which leads to flexibility and adaptability in software development. The development team handles the projects with a prioritized customer requirement list (Product Backlog) to ensure the delivery of features according to their value to customers. Therefore, the development team delivers a shippable product at the end of every sprint cycle. Among the benefits of the Scrum method is the incorporation of some of the parameters used for analysing the delivery plan of agile projects (Chaouch et al., 2019). These parameters include the project cost, delay, delivered functionalities and project quality. Scrum is suitable for use in small teams having less than 10 members because the method does not place much emphasis on process driven architecture. Also, the method's centralization of processes may cause power struggles especially in large projects.

Extreme Programming (XP)

It was developed by an American software engineer named Kent Beck while working on a payroll project at Chrysler Corporation (Beck, 2000). It is a lightweight methodology suitable for small to mid-sized projects with rapidly changing requirements. XP was developed primarily to address the problems of schedule slips and cancelled projects associated with the traditional development models (Beck, 1999). To solve or minimize the problems, the method advocates shortening the release cycles of products as well as ensuring maximum value to customers. The novelty of the XP method remains the ability of collating its different practices to function well with each other to form a new software development methodology. Some important practices in XP include close customer participation, continuous integration and testing, collective code ownership and pair programming among others (Dingsøyr et al., 2010). Most of the practices in XP are focused on the team level while scrum focus more on managing an entire project.

Lean

The idea of Lean thinking was conceptualized by Taiichi Ohno, commonly referred to as the father of “Toyota Production Systems” (Poppendieck & Poppendieck, 2003). The principle is based on identification and elimination of waste. Ohno refers to anything that do not add value to customers, anything not immediately needed and any extra processing step as waste (Morgan & Liker, 2020). The Lean principles were introduced to the software development domain as “Lean Software Development”. Software development was considered a subset of any product development activity or process. The principle initially identified seven wastes in the manufacturing industry which were translated and later refined to better suit the software development domain with the addition of concepts such as “Relearning” and “Handoffs”. The seven software development industry wastes include incomplete/partial work done, extra or not needed feature, additional processing/documentation, task switching, waiting/delay, hand-off, and defect (Poppendieck & Poppendieck, 2007).

Agile software development methods have been well adopted in organisations. The next section discusses the adoption process in software development companies.

2.2.2 Agile Software Development Adoption

The adoption of agile methods by software development companies involves several stages (Wang et al., 2012). The adoption process starts with an initial commitment by software practitioners to use agile practices, either the way it is in the books or through a tailored fashion (Campanelli & Parreiras, 2015). The second stage is the routinisation phase where agile practices are used and embraced. The third stage is the infusion phase where agile methods are comprehensively used. Several studies indicate that companies are incorporating practices from widely recognized agile methodologies, such as scrum and XP (Extreme Programming) (Rindell et al., 2021). When adopting these practices, companies often choose practices that align with their organizational culture and local context. Software companies are embracing agile methodologies for product quality improvement and to measure their overall excellent performance.

Existing literature in the software industry have shown very few studies on agile adoption frameworks (Gandomani & Nafchi, 2015). However, even the proposed frameworks appear difficult to be adopted in small and medium sized enterprises because of their inflexible and complex nature. Transitioning from traditional software methodologies to agile in a company requires the collaboration of all stakeholders including top managers and customers as it takes significant time and effort. Previous studies have shown no standard framework or specific model for agile transition process (Abdelnour-Nocera & Sharp, 2012; Pikkarainen et al., 2012; Sidky et al., 2007). Indeed, the challenges faced by companies can be specific in each environment with unique issues around cultural and technical problems, management issues, processes, and people specific. Answers to questions related to managing the transitioning process, its key features as well as the activities to be accomplished needs to be properly considered (Gandomani & Nafchi, 2015).

There are many studies conducted to investigate agile methods adoption and its challenges especially in the global north (Conboy et al., 2011; Hajjdiab & Taleb, 2011; Nerur et al., 2005; Srinivasan & Lundqvist, 2010). Even with the many studies from the global north, most of them are focused on investigating agile adoption in some certain software development companies as well as challenges faced during the transitioning process. There is still paucity of a standard framework to support the agile transitioning process (Esfahani, 2012). In contrast to studies from the developed world, only a few empirical studies can be said to be emanating from the global south (Mohallel & Bass, 2019; Regassa et al., 2017). The adoption studies in the global south investigated challenges faced by companies in developing countries while adopting agile approaches. The introduction of new initiatives has raised new challenges not just in the software industry (Ezenwa & Brooks, 2014). There are also studies that looked at agile team performance and the factors contributing to team autonomy (Sharp et al., 2009) and member diversity (Nerur & Balijepally, 2007). Other studies have explored how different teams communicate in a large project which included inter-team, team, and customer and as well as teams separated by geographical location (Rahy & Bass, 2018). Also, the paucity of empirical studies also poses another challenge to the agile software development research domain.

2.2.3 Agile Methods in Developing Countries

Agile methods adoption is not a particular new phenomenon in developing countries as there is widespread use of agile practices by software companies. Prior studies in some countries such as Ethiopia have provided empirical evidence on how agile methods adoption can be influenced by user involvement and the nature of clients and contracts involved (Regassa et al., 2017). The research conducted by (Mohallel & Bass, 2019), have discovered the positive impact of agile methods adoption on customer satisfaction and the development process in Egypt. The study however, highlighted several problem areas of the Egyptian software development companies. These problems include the use of inaccurate estimation effort, lack of sprint planning, the relationship between constant pressure on development team and software quality. The study by (Akinnuwesi et al., 2013), reported the perception gap existing between software developers and end-users which in-turn hinders the software development process. This have led to poor acceptance of developed software application and sometimes project failure. Agile methods implementation in Lebanon was explored by (Rahy & Bass, 2020). The study identified factors enabling and impeding agile information systems development in Lebanon as described by practitioners. Some of the identified impediments include the misunderstanding of the agile methodology by both management and team members as well as the political and economic crisis in the country. A study by (Sebega & Mnkandla, 2017), explored the issues of agile requirements engineering in the South African software industry. The study discovered a strong likeness for agile principles by practitioners. The study however, found divergent views amongst practitioners with regards to integrating non-functional requirements (i.e., security, reliability & performance) to the agile software development process. Thus, a common problem of all the agile methods studies in developing countries some of which had been discussed in this section is the lack of security practices integration into the development process. The next section discusses the significance of security integration in agile software development.

2.3 Overview of Software Security

The ubiquitous nature of software is bringing enormous benefits to the way humans transact their daily activities. However, this also comes with the consequence of increasing software flaws and misuse by malicious users. With the daily reported cases of cybersecurity breaches in the news and social media, software security has attracted the attention of industry players and the academia (Amoroso, 2018). The increasing software vulnerability can also be attributed to the astronomic increase in computer connectivity as well as complexity of information systems. The concept of "building security in" is one strategy advocated by researchers towards confronting the challenges of cyberattacks (McGraw, 2006).

The term "Cybersecurity" is defined as the set of practices, guidelines, technologies, and tools used for protecting organizational assets (Von Solms & Van Niekerk, 2013). Cybersecurity needs to guarantee three security attributes which includes confidentiality, integrity, and availability. Most computer security vulnerabilities are attributed to the use of poor

programming practices by software practitioners (Stallings et al., 2012). The CWE/SANS lists the top 25 software errors causing majority of cyber-attacks. Similarly, the Open Web Application Security Project (OWASP) also lists the top 10 web application security risks for managing websites (Siavvas et al., 2020). The purpose of annually publishing the lists is to serve as a guide for practitioners writing codes to be acquainted of current software bugs to avoid. All reported vulnerabilities are usually assigned an identification number and stored in a database known as the National Vulnerability Database (NVD) (Zhang et al., 2011).
Looking at the cybersecurity issues discussed in this section, there is a need to study what efforts have been made in literature to develop security models.

2.3.1 Secure Software Development Models

Existing secure software development models have been divided into linear heavy-weight waterfall-like and light-weight agile lifecycle models (Alenezi et al., 2022; McGraw, 2006). Some of the linear lifecycle models include Microsoft secure development lifecycle (MSSDL), McGraw's touchpoints (MTP), Comprehensive, lightweight application security process (CLASP), and Software Assurance Forum for Excellence in Code (SAFECode).

- **Microsoft Secure Development Lifecycle (MSSDL):** Developed by Microsoft, the model was built to reduce software maintenance cost and to be resilient to cyberattacks. The model proposes 12 high-level practices for the traditional software development process (Microsoft, 2021). The model emphasizes security training for all stakeholder at the beginning of the SDLC to ensure the development of secure software. The model focuses on developing a software that is secure by design by adopting secure coding practices and designs. To ensure *secure by design* software development, some of the practices the MSSDL adopts includes defining security requirements, creation of threat models, using approved tools, and penetration testing. Due to the dynamism of security vulnerabilities, MSSDL aim to develop software that are *secure by default* by building an incident response team to proactively handle emerging threats.
- **McGraw's Touchpoints (MTP):** Proposes security best practices in the software development process (McGraw, 2006). It recommends creation and combining security activities with software artifacts. They include abuse cases, design artifacts, code review, and risk analysis among others. All the practices were proposed based on industrial experience.

- **Comprehensive, Lightweight Application Security Process (CLASP):** Is an activity driven model designed for OWASP to address security issues at the various phases of software development lifecycle (CLASP, 2009; Foundation, 2023). The model defines 24 high-level activities to be adopted by organizations based on their security needs. Some of the high-level activities include the definition of security objectives and security policy, assessment of organizational security posture, verifying security policy, and solving client's security issues.
- **Software Assurance Forum for Excellence in Code (SAFECode):** Proposed in 2008, the model promotes the adoption of secure software development practices. The model aims to promote a good security culture in organizations. Like other *secure by design* models, SAFECode recommends the integration of security practices throughout the development process. Security training is an important practice in the SAFECode model as it ensures that all team members to assign some security tasks during software development (SAFECode, 2009).

The various Secure Development Life Cycles (SDL) discussed in this section followed Security-by-Design principle. Each of these models incorporates threat modelling at the initial stages of the development process, along with ongoing security testing throughout all phases of software product development. A major limitation of these approaches is the associated cost. They presume the engagement of a "costly" team of security experts throughout the entire development life cycle and/or the availability of developers with security expertise. In practice, only a limited number of companies implement these methodologies as a standard procedure, typically for the development of security-critical applications (Casola et al., 2020).
Another problem common to all linear models discussed above is that they were developed with heavy-weight activities (Alenezi et al., 2022). The models require comprehensive documentation, adopting predictive approach with the tendency to design large part of the software from the beginning of software development. Most linear models were also not developed using real observed data except for BSIMM (Rindell, Ruohonen, et al., 2018). However, the light-weight (iterative/incremental) models which are the focus of this thesis are people oriented, adaptive, and collaborative through self-organizing teams. The existing studies discussed in the next section used tailored practices to develop secure agile models.

2.4 Secure Agile Concepts

Conventionally, agile software development methods are composed of three elements - roles, ceremonies, and artefacts (Bass, 2016; Mihelič et al., 2023). Integrating these elements into the software development process improves security (Mihelič et al., 2023), and leads to better intra and inter teams' collaboration (Moyón et al., 2020). This section discusses the elements of the popular agile methods.

2.4.1 Secure Agile Roles

Typical agile roles include scrum master, product owner and development team. In addition to these roles, several studies have suggested the incorporation of a security resource to the development teams (Ardo et al., 2021; Ardo et al., 2022; Baca et al., 2015). The nomenclature of the security resource varies across different companies. Some of the proposed variations includes the following:

- **Security Guru:** The person responsible for handling security due to their expertise knowledge. The role has several variations depending on the organization – security master, security manager and security expert (Mihelič et al., 2023). The main essence of introducing a security role is to handle risk points and document same in the security backlog.
- **Security Developer:** Responsible for designing security test cases, analysing potential threats and conducting attack surface analysis (Baca et al., 2015). The role is relatively low cost compared to other security-specific one's and help teams maintain agility in software projects. Additionally, they help in spreading security expertise within the team by collaborating with others such as penetration testers.
- **Penetration Tester:** Responsible for testing software vulnerabilities. They are also responsible for exploring possible weaknesses in software system and suggest proactive measures to prevent attacks. They use the risk analysis conducted by security developers or adopt automated testing tools to verify the security of software systems (Tomanek & Klima, 2015).
- **Security Team:** A group comprising of different security-specific roles. The roles are broadly grouped into two – team-internal and team-external (Nägele et al., 2022b).
    - **Team-internal:** They are people grounded in security knowledge. Their responsibility includes coordinating security activities in an organization as well as

serving as intermediary between team-internal and team-external members and others like product owners. The team-internal members are given different designation within the company such as security specialist or software security engineer (Nägele et al., 2022b).

- **Team-external:** They help agile teams with their security expertise and conducting activities such as threat modelling in collaboration with development team. Some of the designations/roles in team-external are security consultants and security engineers. Other duties are raising awareness about regulatory policies, standards and best practices to ensure adherence (Nägele et al., 2022b).

2.4.2 Secure Agile Ceremonies

To improve team communication and ensure collaboration amongst project stakeholders, ceremonies/activities are held. They are some unique kinds of meetings held by various teams. The secure agile ceremonies include security planning and threat modelling, software security audits, security risk management, security analysis, security requirements engineering, security monitoring and security training (Mihelič et al., 2023).

- **Security planning & threat modelling:** It describes meetings conducted with the aim of identifying software operational threats and to ascertain necessary safeguards not implemented. According to (Bernsmed et al., 2022), threat modelling is considered one of the most important ceremonies in secure agile software development at the design phase. It empowers practitioners to think like an attacker and explores threats from different system assets.
- **Software security audits:** This activity involves reviewing software to ascertain organizational overall security readiness. It usually includes security checks on code and its quality, adherence to standards and regulations and test cases review. It may be conducted by internal as well as external auditors.
- **Security risk management:** In agile projects, different risks are considered including security. It involve activities aimed at identifying security vulnerabilities and defining mitigation mechanisms (Tøndel, Jaatun, et al., 2019). Some of the different levels of security risk management includes identification, assessment, and hazard analysis. While risk management is crucial in software projects, there are evidence of agile practitioners

not considering security risk estimation as an ongoing activity throughout the project lifecycle (Tøndel et al., 2017).

- **Security analysis:** Meetings dedicated for testing and analysing software performance. For instance, penetration testing sessions, risk-based security tests and vulnerability analysis (Mihelič et al., 2023).
- **Security requirements engineering:** Ceremonies conducted with the objective of defining and documenting security activities during software engineering design. Some of the typical security-related activities include defining evil user stories and writing abuser stories (Mihelič et al., 2023).
- **Security monitoring:** It defines a group of activities conducted for prioritizing and monitoring of security requirements, goals, and objectives to ensure proper implementation. These activities include protection poker (Tøndel, Jaatun, et al., 2019), continuous re-aligning of security objectives and security/business impact analysis (Tøndel, Cruzes, et al., 2019).
- **Security training:** Sessions aimed at educating agile team members the importance of integrating security throughout the development process (Oyetoyan et al., 2016).

2.4.3 Secure Agile Artefacts

Agile artefacts describe the elements used for sharing project information between team members. The shared information may include project status, activities planned in upcoming sprints and tasks completed. Some of the agile artefacts described in scrum include product backlog, sprint backlog and the burn down chart (Bass, 2016). The scrum framework also defined other artefacts such as product vision, sprint goal, definition of done and the product increment. In secure agile software development, security-focused artefacts have been defined in literature. These artefacts include the following:

- **Security policies:** Security-related documents guiding secure software development. They also describe security standards, solutions and best practices (Bezerra et al., 2020). These documentations may range from secure coding policies/templates/standards, and security test and audit plans.
- **Security requirement artefacts:** It describes security-related items used in requirements gathering. The elicited security requirements are prioritized in a security backlog. Example

of security requirement artefacts include abuse/misuse cases, abuser stories, and security user stories (Mihelič et al., 2023).

  - **Abuse/Misuse cases:** It describes how the software or it's users may be attacked from an attacker perspective (Jabangwe & Nguyen-Duc, 2020).
  - **Abuser stories:** It describes the malicious intent of an unauthorized user for launching an attack. It is unlike user stories which describes software features and functionalities from the user's perspective. Usually, they are developed through collaboration with all team members i.e., development and security (Jabangwe & Nguyen-Duc, 2020).
  - **Security user stories:** Illustrates the possible threats to software systems. The threats can either be from internal or external sources. Usually, they are developed exclusively by the practitioners with security-specific roles for the use of development teams (Maria et al., 2015; Siiskonen et al., 2014).

- **Security repositories:** Provides a well-managed checklist of risk for taking proactive measures against software attacks (Mihelič et al., 2023). Example of these security repositories include security backlog (Bezerra et al., 2020), safety product backlog (Doss et al., 2017) and security requirements repository.
- **Security reports:** Documents written after performing security activities during the software development process (Rindell et al., 2021). These reports include security testing, auditing, and code review reports.

To ensure the security of SDLC and the eventual software to be produced, a secure-by-design process is adopted. The next section discusses the secure-by-design agile process.

2.5 Secure-By-Design Agile Process

The Security-by-Design concept advocates adopting proactive measures against security threats. It involves embedding security into the design of systems through the adoption of both software security assurance processes and trusted hardware (Casola et al., 2020). Software assurance processes within Security-by-Design encompass various activities. These include conducting a thorough threat analysis, integrating countermeasures against existing threats into the system architecture, implementing regular code reviews and audits, and conducting rigorous security testing. It is an all-encompassing process which exceeds the common notion of limiting secure software to just the design phase. Thus, security issues are considered at the requirements, design, implementation, testing, deployment, and maintenance. In agile software development, more time tends to be devoted to implementation because of the risk of introducing new flaws at this stage (Kreitz, 2019). To enhance the effectiveness of implemented security measures, it is crucial to incorporate security practices throughout the development process. In a secure-by-design process to agile development, different techniques are performed at each phase of the SDLC.

2.5.1 Security Requirements

Defining security requirements in clear and consistent manner is key to developing secure software applications (Bezerra et al., 2020). The incidences of cybersecurity breaches can be reduced by adopting the following practices.

- **Security backlog:** It is a list of activities like the product backlog; however, it contains only security specific items. Depending on the needs of a project, the product owner together with the development team may decide to separate it from the product backlog. The security backlog is periodically updated by adding and prioritizing items in every requirement definition cycle (Azham et al., 2011).
- **Evil user stories:** It is based on security backlogs and focuses on describing threats scenarios of how software systems maybe attacked. These threat scenarios help in identifying software vulnerabilities that can be exploited by malicious users (Barbosa & Sampaio, 2015). It is always a good practice to have it documented just like user stories by a security expert or any experienced member of the agile team with security knowledge so modelled scenarios depicts real attacks. Evil stories are mostly defined alongside the product and security backlogs and reviewed in each iterative cycle.
- **Hacker personas:** It is a kind of user experience or interaction technique used for designing a non-existent user that might be interested in using a software. The Hacker persona is modelled to have users with different unique mindsets (Gondree et al., 2013).

2.5.2 Security Design and Planning

The security design and planning phase is crucial to avoid problems in the future. Poorly designed software systems are mostly vulnerable to cyberattacks. Adopting good software architecture, proper evaluation of risks and project monitoring reduces security flaws during implementation. Additionally, increasing the security competence of agile team members significantly reduces security challenges. Some of the security practices at the design phase includes the following:

- **Protection Poker:** This technique is based on the planning poker concept which is aimed at prioritizing software development activities according to the risks associated with them. During the software development planning session, security activities are explained to team members and every item is scored. Final score from all members is aggregated with the greater risk activities assigned higher values. This technique is helpful for prioritizing project security requirements (Williams et al., 2010).
- **Performing Risk Analysis:** The risk assessment documentation contains a detailed analysis of the risks associated with each security requirement which are reviewed after every iteration (Franqueira et al., 2011).
- **Misuse Cases:** They are undesirable behaviours in software systems that can cause security breaches. They are sometimes described as negative use cases and are represented using UML diagrams (Jabangwe & Nguyen-Duc, 2020; Sindre & Opdahl, 2005).

2.5.3 Security Implementation

Most vulnerabilities in software systems are attributed to lack of security practices integration in the development process. Implementation of security practices can reduce the incidences of cyberattacks. However, despite efforts at implementing security, improper implementation of security features leads to software breaches (Kanniah & Mahrin, 2016). Proper implementation of security mechanisms during software development can help identify and prevent software attacks. These mechanisms include the following:

- **Secure Coding Rules:** Software security good practices can be adopted to prevent coding flaws. Well established secure coding rules should cover important areas including adopting cryptographic standards, using secured third-party libraries, and avoiding the use of obsolete software functions. Other rules may include using programming guides and approved software tools (Sodanil et al., 2015). These rules should be adopted throughout the software implementation phase especially in handling high risk work items.

- **Secure Code Review:** Session devoted to discussing security issues identified in team member’s codes. It gives practitioners the opportunity to learn new security skills from peers through giving and receiving feedback. Secure code review sessions also give organizations the opportunity to prevent software failures before they happen. Involving reviewers with security expertise is always a good practice to avoid software breaches (Bernhart et al., 2010).

- **Hardening Sprints:** It is an agile sprint which focuses on stabilizing software code to get it ready for release. It is dedicated mainly to security reviews, software performance and quality improvement as well as delivery (Bell et al., 2017).

2.5.4 Security Testing

There are two major reasons for performing security testing. First, it is performed to test security requirements which concerns (availability, authentication, authorization, integrity, confidentiality, and non-repudiation) (Felderer et al., 2016). Second, security testing is done to validate a software system’s resilience to withstand attacks (Paul, 2013). Security specialized testing techniques are used to determine if software is securely implemented.

- **Penetration Testing:** A form of black-box security testing where an actual attack is set-up from a malicious third-party (Felderer et al., 2016).

- **Security Automated Testing:** Involves implementing automated tests which includes a process for validating security-related features. Security checks implementation prevents distortion of codes and ensures fixed software vulnerabilities do not occur again (Nicolaysen et al., 2010).

- **Secure Code Analysis:** Use tools to assess the existence of security holes to validate the security of implemented software (Baca & Carlsson, 2011).

2.5.5 Security Deployment and Maintenance

This phase is concerned with checking all implemented security control once again to ensure the software is ready for deployment. Secure review sessions are performed to check security controls including static and dynamic configurations and container security before final software deployment. After deployment, a process of continuous software application monitoring is implemented to identify security vulnerabilities and address them promptly.
While integrating security practices into different phases of the development process is important, there is paucity of research in the domain. However, there are few studies on agile

security practices implementation in the existing literature (Rindell et al., 2021; Rindell, Ruohonen, et al., 2018).

2.6 Agile Security Practices

The implementation of agile methods in the software industry appears to conflict with security practices and requirements (Rindell et al., 2021). Developing secure software means performing a set of security practices or activities while following a software development lifecycle process. However, an agile development process is not always determined in advance. Thus, the main objective of integrating security practices within the development process is to address any security vulnerabilities identified to build an efficient and effective software system.

Rindell et al. (2021), conducted a survey among Finnish agile developers to determine the security practices they adopt during software development. The study used 40 security engineering practices in conjunction with 16 agile software development activities to investigate their perceived impacts. The study findings showed discrepancies between the perceived impact of the security activities used and their level of use. Also, the study findings were not compared with any existing baseline studies in other contexts. Newton et al. (2019) presented 12 critical success factors (CSF) used by agile practitioners to address problems associated with traditional software development process. Although the CSFs also helped to address tension between agile software development and software security teams, they remain conceptual as none of the constructs were validated to determine their relationship. While the emergence of the agile manifesto was a potential source of motivation for researchers to adopt agile development techniques, practitioners still report challenges associated with building secure agile software.

2.6.1 Challenges of Secure Agile Software Development

Existing studies have highlighted some of the challenges of developing secure agile software which include a lack of cross-team collaboration around security issues (Sánchez-Gordón & Colomo-Palacios, 2020), the cost factor (Venson et al., 2019), compliance issues Agbali et al. (2020); (Presthus et al., 2018), and the lack of practitioners with skills in both cybersecurity and agile software development Alshaikh (2020); Egere (2020); Nägele et al. (2022b) to mention a few.

Nägele et al. (2022b), conducted an empirical investigation of security issues in large-scale agile environments and discovered a conflict between security governance and the autonomy of agile teams. While the authors tried to encourage cross-team collaboration through regular meetings and training, security knowledge sharing remained a key challenge. Similarly, Tøndel et al. (2022), referred to the collaborative nature between agile teams and security specialist as 'not always optimal'. This was potentially influenced by unclear security requirements and not integrating security practices in the development process.

Sánchez-Gordón and Colomo-Palacios (2020), considered team collaboration in agile software development as a culture. They introduced the idea of DevSecOps in their study as a way of integrating security principles by collaborating with cross-teams (security, development, and operations). However, the review highlighted a lack of collaboration between agile and security teams.

A systematic literature review was conducted by Venson et al. (2019), to classify existing studies related to secure software development project costs. The study used different search strategies such as manual, automated, and snowballing to select 54 papers from 47 distinct software engineering and security journals, and conferences relating to sources of cost in secure software development. While there was evidence that agile teams adopt security practices which adds to the cost of developing secure software, only a few cost estimation models consider security, which could be due to the prevailing organizational cybersecurity culture.

Cybersecurity culture has been described by Da Veiga et al. (2020), as the behaviour of humans to protect the information they process through compliance with organizational security policy. This is achieved through regular and effective communication, awareness, training, and educational programs. Security skills and training are considered vital for building a good organizational cybersecurity culture. Alshaikh (2020), reported that five initiatives were adopted by some Australian companies to develop cybersecurity culture. However, a lack of security skills and knowledge continued to be a major cause of software vulnerability (Egere, 2020; Jøsang et al., 2015).

To develop better and more secure software systems, organizations are required to adhere to baseline security practices. However, compliance does not automatically equate to secure practices (Zerlang, 2017). As cybercrimes globally continue to evolve at a fast pace, it is difficult for government policies to keep up with the dynamic security landscape. The implications of the Nigeria Data Protection Regulation (NDPR) recently launched in Nigerian were investigated by Agbali et al. (2020). The study used an institutional theory lens to interpret the policy's level of adoption and implementation. In another study, the issues confronting

Norwegian practitioners in complying with GDPR legislation were explored by Presthus et al. (2018), while Li et al. (2019) investigated the impact of GDPR legislation on global technology development and how the world leading economies, such as China and the USA could respond to GDPR challenges and opportunities. Thus, from all the studies discussed in this section, both the GDPR and NDPR face certain implementation challenges peculiar to their context. This thesis explored the challenges confronting agile practitioners which has resulted to non-adherence to the NDPR during secure software development.

2.6.2 Nigerian Data Protection Regulation (NDPR) Act 2019

The widespread use of information technology has facilitated the broad collection and sharing of personal data, commonly known as Personally Identifiable Information (PII) (Krishnamurthy & Wills, 2009). This encompasses various details that can identify an individual, including full name, email, phone number, national ID, gender, race, religion, debit card number, and date of birth. Globally, authorities have established regulatory policies addressing personal data and privacy protection. These policies aim to protect data subjects from potential abuses of their Personally Identifiable Information (PII) and to reduce security and privacy compromises. In Nigeria, the National Information Technology Development Agency (NITDA), formulated the Nigerian Data Protection Regulation (NDPR). The legislation is aimed at safeguarding the rights of people living or transacting business in Nigeria to data privacy and security. The policy also aims to enhance secure data exchange and preserve the integrity of the business environment (NDPR, 2019). The policy is a requirement for all organization handling data.

Several studies have examined the effectiveness of regulatory policies regarding the protection of personal data. The operational methods of contact tracing software applications employed in Nigeria and worldwide, to combat the dreaded COVID-19 pandemic have been examined (Ekong et al., 2020). The authors' concluded that the implementation of contact-tracing apps often conflicts with the privacy rights of data subjects, resulting in security breaches. However, their deployment largely adhered to the NDPR Act. Similarly, an analysis of the timeline regarding the security and privacy of Facebook users has shown that the NDPR is inadequate in safeguarding Nigerians against potential software attacks (Ehondor & Ogbu, 2020). Likewise, an assessment of NDPR compliance has been undertaken in Nigerian Ministries, Departments, and Agencies (MDAs) (Abubakar et al., 2022). The findings indicated that around 30% of the surveyed MDAs were unaware of the provisions of the NDPR. The study highlighted the low level of compliance with NDPR among MDAs, which is consistent with findings from other existing studies (Abdulrauf, 2021).

The introduction of the NDPR has significant implications for practitioners, as companies are obligated to safeguard client data from cyberattacks. With the recent incidences of cyberattacks in Nigeria, there is a need for companies to increase their efforts at protecting against cybersecurity threats. The NDPR has further increased the need for companies to engage professional cybersecurity experts as well as data protection officers. However, there is currently a shortage of these professionals in Nigeria (Egere, 2020). The shortage of cybersecurity professionals is not particular to Nigeria, as a study by Li et al. (2019), noted the need for both governments and technology company managers to invest more in cybersecurity education and training to comply with the regulatory policy and security standards. Apart from adhering to regulatory policy, adopting secure models also proactively guards against cyberattacks in the development process.

2.7 Secure Agile Software Development Models

Section 2.3.1 discussed secure software development models based of heavy-weight principles. However, the trend in global software industry have shifted towards light-weight development models (Alenezi et al., 2022). This can be attributed to their heavy-weight activities, inflexibility, and huge documentation needs amongst other challenges. Thus, this section explores different models developed based on light-weight agile lifecycle. The models use tailored practices to improve security and prevent malicious activities.

Baca et al. (2015), proposed a security-enhanced agile software development process model (SEAP) for a money transfer system. The study defined four security competences including penetration tester, security architect, security manager and security master. Results shows the time for conducting the risk analysis process for a previous baseline model was 2.7 employee hours as against SEAPs 1.5 hours. While the study focused on reducing the employee hours for conducting risk analysis which was obviously achieved, the issue of incorporating security roles, ceremonies, and artefacts onto the model to improve software security during the development process was not in their scope.

de Vicente Mohino et al. (2019), developed a secure agile software development model using security practices from 16 existing lifecycle models. The three pillars of the model include agility, security, and flexibility. The study presented 23 recommendations for applying security practices. The study adopted a scenario-based validation method in addition to expert evaluation of the model. While the study tried to integrate agile roles, ceremonies, and artefacts to their secure software development lifecycle (S-SDLC) model, none of the used ceremonies were security specific (e.g., protection poker, security design reviews). All the seven ceremonies defined are conventional agile ceremonies which made no reference to security issues.

Bezerra et al. (2020), proposed a set of five security policies consisting of different practices suitable for self-organizing agile teams. The security practices that formed the policies were synthesized from the existing literature and verified through a workshop organized for some agile and security professionals working for a software company. The study proposed a new ceremony for agile teams known as "protection poker", which involves collaboratively scoring each development activity to determine highest risk work item. While the five security policies proposed contained security specific ceremonies like the protection poker and security code reviews as well as artefacts such as security backlog and security coding rules, no security role was defined for any of the five policies.

Rindell, Hyrynsalmi, et al. (2018), developed a framework which mapped common agile processes and artefacts onto a secure development lifecycle model. The security practices used in developing the model were drawn from three heavy-weight linear models which are Microsoft SDL, ISO common criteria and the OWAP SAMM. The practices were integrated with the aim of supporting both agile and secure software development projects. Although the study mapped various agile artifacts and ceremonies onto the developed process model, it did not incorporate security roles and how they could be integrated into the agile teams. By integrating security roles, agile team collaboration would be enhanced during the development process to reduce the incidences of software vulnerabilities (Rindell et al., 2021). While the study included a compliance artifact in its model, it overlooked the integration of security-related agile roles in its secure process model.

2.8 Security Compliance in Agile Software Development

Nowadays applying agile methods are not restricted to developing traditional systems but are adopted for critical systems where security standards and regulations are essential. One of the challenges faced with practitioners is the issue of how to ensure compliance with security standards and regulatory policy. In this section, some of the existing studies relevant to the theme of this thesis are discussed here.

Oueslati et al. (2015), conducted a systematic review on reported challenges of developing secure agile software in the existing literature. The study discovered 20 challenges from 10 publications. Of the 20 challenges, 14 were considered valid while the remaining 6 were found not have been caused by agile values or principles. Among the 20 challenges is the issue of compromising security activities to accelerate early software release leading to non-compliance to regulations and standards.

Mohan and Othmane (2016), conducted a survey of literature from both academia and the industry to identify current trends in the SecDevOps domain. The study identified 5 main aspects of SecDevOps to include process automation, team collaboration, software configuration, compliance, and information secrecy. While the study identified security compliance as a great challenge, none of the software standards were mentioned.

Villamizar et al. (2018), discussed the lack of security requirements and practices integration in agile software projects. The study identified 5 ways of handling security requirements together with limitations of all the approaches. The study found that the identified approaches focused on introducing new introducing new guidelines for handling agile methods security concerns. Thus, nothing was reported about security compliance to the introduced guidelines and standards.

Moyón et al. (2020), conducted a systematic mapping study of security compliance in agile software development. The study established that security compliance remains a concern for both agile practitioners and researchers. The study noted the focus of existing studies on integrating security standards but lacked any empirical evidence on how to assess compliance. While the study made a valuable contribution by increasing awareness within both research and practitioner communities about existing research on security compliance and agile software development, it fell short in offering practical methods for assessing compliance.

Moyón et al. (2021), proposed a tool-supported approach to make it easier for both security experts and software developers to understand security standards. The scope of the study was limited to using BPMN process models to implement IEC 62443-4-1. While the study was structured to be in tune with existing trends of secure agile empirical studies by engaging practitioners, it raises some questions that needs to be answered. These questions include: “Do models represents better ways of getting security norms accepted into daily software activities?”. “Can models provide better collaborative support between security and development teams?” These questions need to be answered to provide convincing evidence to using process models to understand security standards.

Based on the discussions above, compliance to security standards and regulatory policies during the agile software development process is almost non-existent. This thesis has provided some valuable knowledge to the domain of security compliance in agile software development where there is still limited empirical research. Most of the research are secondary studies (mapping studies and literature), which focused mostly on theoretical discussions.

2.8.1 Security and Agile Software Development

Existing studies have highlighted the lack of security activities in agile software development (Adolph et al., 2011; Baca & Carlsson, 2011; Oueslati et al., 2015). Baca and Carlsson (2011), looked at the compatibility of security practices with the characteristics of the agile manifesto. Bansal and Jolly (2014), evaluated security practices with the aim of proposing ways of developing secure software development processes. The limitation of the study in (Baca & Carlsson, 2011), is that the security enhanced agile development process produced has not been empirically evaluated and compared with other existing development processes. For the study done by (Bansal & Jolly, 2014), factors such as cost, time and recurrence which may affect the compatibility between two security activities that can help project managers in decision making has not been considered.

Terpstra et al. (2017), conducted a systematic review on security in agile methods which revealed several methods for integrating security in agile methods. The study further identified three contextual factors important for shaping security in agile projects. These factors are solutions addressing the artefacts, solutions addressing the human factors in agile and solutions addressing the agile process itself. However, the identified solution factors were only focused on people, but nothing was mentioned with regards to tools or other sophisticated methods used. Also, data collected was only based on practitioners' posts on LinkedIn which would not allow for in-depth understanding of the phenomena.

Cruzes et al. (2017), studied security testing practices in agile teams across different organizations. The existing literature on software security testing practices broadly focuses on two areas. The first testing focus area are those done for security services such as confidentiality, integrity, availability, authentication, authorization, and non-repudiation (Felderer et al., 2016). The second focus area is in the aspect of testing for software resilience to withstand attacks (Adolph et al., 2011). In the context of secure agile software development, security testing issues include validating if security requirements have been implemented properly to avoid attacks. Also, there are issues of identifying unintended software system's vulnerabilities.

Bezerra et al. (2020), have grouped the agile security practices based on practitioners' assessment in a particular cyber security organization. Thus, a common finding of agile software development studies in the existing literature is that agile methods do sometimes comply with security requirements, but it is faced with the issues of higher cost and slower development due to inadequate agile security processes (Rindell et al., 2021).

Despite the existing studies highlighting the importance of the secure-by-design concept, several challenges persist. This is primarily because security practices are not integrated into every phase of the agile software development process (Casola et al., 2020). However, there are few studies that have suggested the integration of security practices into agile methodologies like Scrum. For example, an extension to the Scrum methodology, adopting the security backlogs, to be used during the analysis and implementation phases of software development has been proposed (Azham et al., 2011). However, their proposal lacks specific phases dedicated to defining security requirements and conducting risk assessments. This suggests an incremental development process with frequent code changes, potentially compromising security, as discussed in (Oueslati et al., 2016). While several secure-by-design process models exists, to the best of the researcher's knowledge, there is still paucity of empirical evidence on secure agile software models and so this study can add value to the existing body of knowledge.

2.9 Gap in Knowledge

As mentioned earlier, integrating security practices into agile software development process remains a challenge (Aalvik et al., 2023). Many of the existing studies either used practices from linear models or those identified from literature and not empirically explored. Some studies have also used fewer agile concepts to build security models. There are existing studies that adopted one or two of the agile concepts to develop secure models. Baca et al. (2015), developed an enhanced agile model for money transfer, but only security roles were discussed. Bezerra et al. (2020), did not define secure roles within the five security policies they developed; instead, they only referenced security ceremonies and artifacts. However, the study in (Rindell, Hyrynsalmi, et al., 2018) didn't incorporate any security roles in their developed framework but mapped some secure agile ceremonies and artefacts onto its processes. Another study proposed an agile process model, but the ceremonies adopted were not security specific (de Vicente Mohino et al., 2019). Therefore, this research has bridged the identified knowledge gap by proposing a Secure-by-Design process model that integrated security practices across the different phases of the software development process.

Compliance to regulatory and security standards is another concern in secure agile software development. While the theoretical knowledge is vast, there is paucity of empirical evidence especially regarding challenges faced by practitioners. Although, the study in (Usman et al., 2020), provided insights into the practices and challenges to regulatory compliance, it was conducted in the European context, examining regulatory standards such as the GDPR. To the best of my knowledge, there are no existing studies examining the impact of regulatory policy for secure agile implementation in the global south and specifically in Nigeria. Existing studies only considered the NDPR as a legal requirement to which Nigerian companies are mandated to adhere. However, there is a lack of studies within the existing literature that examine the NDPR from the perspective of agile practitioners developing secure software. Thus, this study contributes to the body of empirical evidence on regulatory policies compliance in secure agile software development. Despite the importance of secure regulatory policies in agile environments, limited empirical research exists and so more is needed (Moyón et al., 2020; Nägele et al., 2022a).

2.10 Summary

In this chapter, existing literature related to the research phenomenon was reviewed. The background to agile methods and secure software development models were examined. The existing literature showed a lack of agile security practices integration. There was also lack of empirical evidence on compliance to regulatory policies and security standards which provided further evidence for the need to conduct this study. The reviewed literature provided gaps that motivated this research. Thus, there is need for in-depth studies especially in the global south where research is lacking. The next chapter on research design described the methodological choices adopted to achieve the research aim.

Chapter 3

Research Design

3.1 Introduction

The chapter presents the research methodology adopted for this study. It outlines the research approach, data collection methods and analysis procedure. It presents a detailed explanation of the research methodology applied to achieve the aim and objectives as outlined in chapter one. The study follows the design guidance of the research onion which is discussed in section 3.2 together with the justification for choosing its various elements (Saunders et al., 2007). This is followed by the research purpose and approach in sections 3.3 and 3.4 then, the research sites of this study are presented in 3.5 and 3.6. The data collection and analysis process adopted were then elaborated for the research sites. The process of recruiting participants and interviewing methods are explained in detail. The justification for adopting a multi-methods qualitative methodology is also discussed. Finally, the secure agile model creation and application procedure were explained in sections 3.7 and 3.8 while 3.9 summarizes the chapter.

3.2 Research Model

The research model describes the objects of the study (Palvia et al., 2006), which can be effectively determined through proper interconnection and interaction between and amongst different design components (Maxwell, 2012). Figure 3.1 presents a model comprising of the various layers used in the development of a research project. The aim of the model as presented by its developers (Saunders, Lewis, and Thornhill) is to help researchers better organize their methodology as explained in their book titled “Research Methods for Business Students”. The research onion is a taxonomy for justifying methodological choices (Saunders et al., 2007).

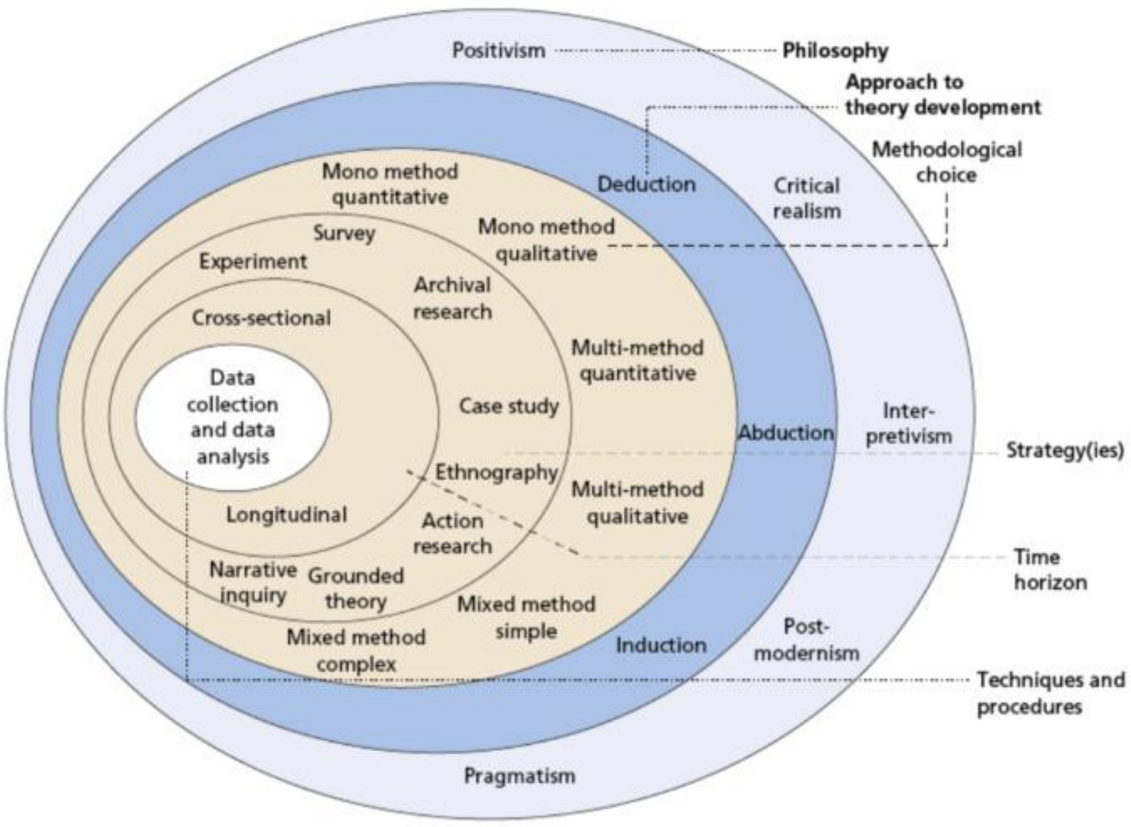


Figure 3.1: Research Onion (Saunders et al., 2007)

3.2.1 Research Philosophy

In carryout any research, assumptions are made either knowingly or unknowingly by the researcher (Burrell & Morgan, 2017). Research philosophies refers to the set of principles that describe a researcher’s worldview about the phenomena been investigated. The four popular research philosophies include positivism, interpretivism (constructivist), critical realism and pragmatism (Easterbrook et al., 2008; Petersen & Gencel, 2013).

Positivists assume the reality under investigation is observed and not constructed. Positivists have an epistemological stand that is objective in knowing or investigating the reality in question and an ontological stand of subjective reality. They view the existence of objects to be independent of the researcher (Myers & Avison, 2002). The paradigm lays emphasis on finding cause and effect of a phenomenon. While positivism has been a dominant research philosophy compared to other paradigms (Davison & Martinsons, 2011), it is not appropriate for answering the research questions in this study. It lacks the ontological depth to explain relationship between variables making it difficult to contribute to explanatory knowledge.

Interpretivist researchers' belief reality is socially constructed. They see things in a subjective manner which also varies from one person to another. The epistemological stand of interpretivist researchers is the belief that the world's existence is dependent upon human knowledge which is subjective (Klein & Myers, 1999). The researcher employs methodologies that help them to understand the occurrence of events based on their individual viewpoints. Thus, the weakness of this philosophy is the possibility of research participants having inadequate knowledge and understanding of the phenomenon under investigation. Also, there is the possibility of false consciousness of peoples' assigned meanings and values.

Critical realism captures the complexity of the real world through its multi-pluralist approach (Carlsson, 2012; Mingers, 2004). The ontological stand of the critical realist relies on two assumptions; reality is independent of the knower (researcher). Such a reality is transcendental (i.e., reality exist in three stratified domains of the empirical, actual, and real domains), hence the existence of stratified ontology (Bhaskar, 2008). Critical realism is not suitable for this research due to its assumption of objective truth, thus, aligning with the positivist paradigm (Sekaran & Bougie, 2016).

Pragmatism uses all possible methods available to understand the research phenomenon. It is not confined to a particular philosophical viewpoint but rather uses the most suitable method for the problem context (Petersen & Gencel, 2013). This paradigm believes the research questions would determine the approach to be adopted to produce meaningful knowledge (Sekaran & Bougie, 2016).

This research adopts the pragmatic philosophy. The viewpoint allows for the adoption of research methods suitable for achieving the study aim. It also allows for the combination of different philosophies in a research (Petersen & Gencel, 2013). This approach views truth as tentative which may change over time (Sekaran & Bougie, 2016).

As earlier mentioned, the aim of this research is to develop a secured process by exploring agile security practices implementation based on practitioners' perceptions. The research further examined security compliance in agile software development. Thus, a pragmatic worldview is more appropriate for investigating such a phenomenon taking into account the socio technical nature of software development.

Phases 1 and 2 of the research were influenced by the interpretive philosophy. The focus then was on understanding the agile security practices in organisations during software development. A method informed by grounded theory was adopted for data analysis. Phases 2 also adopted GT for theory generation rather than verifying existing ones. In Phases 3 and 4, a multi-methods approach was adopted, through the model developed and implemented in an organization with the aim of deliberately influencing current security practices. Thus, this study went beyond just describing or interpreting what was on-going but rather to explore ways of impacting the software security practices of a company.

3.2.2 Research Logic

It describes the direction taken to achieve the aim and objectives of the research (Machamer & Silberstein, 2008b). It can also be described as the method of reasoning adopted by a researcher (Machamer & Silberstein, 2008a). The decision on an appropriate research logic is influenced by the philosophy chosen. The three common logical inferences are inductive, deductive, and abductive approach.

Inductive Approach. starts with a specific observation by the researcher and then patterns are looked for in the data collected. Its drawbacks include the fact that the results are limited to the empirical level and that the inferences drawn are never analytically or empirically certain.

Deductive Approach starts from a general concept and moves to specific. The data collected is used to either confirm or refute a hypothesis. The limitation of this approach is that no new knowledge about reality is discovered other than what is already in the premises and its conclusions are analytical (Gregor, 2006). Therefore, validity of conclusions can be dependent on adherence to certain logical rules.

Abductive Approach starts with unexpected fact then different premises are considered and the most sufficient one is chosen (Saunders et al., 2009). This approach aligns with the pragmatist philosophical perspective.

This research takes an abductive reasoning approach, been the most appropriate to answer the research questions. The concept of security practices integration and compliance remains ambiguous in agile software development literature. The researcher believes beginning with a specific aspect of the concept would limit an in-depth investigation of the phenomenon. This research began with an exploratory case study with the aim of investigating security practices implementation in organisations, then the formulation of hypothesis based on analysis of findings. The abductive mode of reasoning guided the researcher interpret the studied phenomena. The study further developed a practice-based model and evaluated it with a team of agile practitioners and cybersecurity professionals.

3.2.3 Methodological Choice

Existing literature have grouped methodological choices mainly into three; qualitative, quantitative, and mixed methods (Cruzes et al., 2015; Williams, 2007).

Qualitative method is adopted when the aim is to understand a research phenomenon. It explores in-depth participants' perceptions using data collected. Data analysis is usually conducted inductively by starting with a specific concept and moves to a general theme (Patton, 2014).

Quantitative method adopts mathematical techniques to test theories and evaluate the relationship between variables (Cruzes et al., 2015). Quantitative methods are associated with the deductive research approach.

Mixed method combines both qualitative and quantitative methods in a single study. Researchers that had adopted the mixed method belief it gives better understanding of the phenomena been investigated and gives credence to research strategy used (Cruzes et al., 2015).

All the approaches discussed in this section were designed for answering specific type of research questions (Williams, 2007). Having examined the three common methodological choices described above, **this research adopted a qualitative methodological choice**. This method aligns with the pragmatic perspective of this study, where the research questions determine the method to be adopted. A qualitative method has been employed to investigate questions related to the intersection between agile software development process and cybersecurity issues, and the integration of security practices. The adopted approach helped the researcher in understanding and interpreting underlying opinions of the phenomena under investigation (Creswell & Creswell, 2017). The investigation involved collecting data through case studies and analysing it using an approach informed by grounded theory (Williams, 2007). Multi-methods are used to develop and evaluate a secure software development process model.

3.2.4 Research Strategy

The research strategy describes how a researcher intends to carry out the proposed work. There are several strategies for conducting research depending on what is been investigated. Research strategies such as grounded theory (GT), action research, case study research, ethnography, are associated with qualitative inquiry (Williams, 2007). Other strategies like experiments, surveys and archival research are related to quantitative research (Saunders et al., 2009). **For this research, two strategies were employed. These strategies are case study and grounded theory research.**

This study adopted **case study research** with a data analysis method informed by grounded theory, aimed at influencing existing security practices in organisations. Case study research involves the collection of empirical data on real-life occurrence, over a period (Merriam, 1988). It has defined boundaries which is geared towards in-depth understanding of the phenomenon being investigated. It is a widely adopted strategy in the software engineering research (Boehm & Huang, 2003; Iyawa et al., 2016). The guidelines for conducting case study research in the field of software engineering is well documented in literature (Runeson & Höst, 2009). The exploratory case study in phase 1 is an investigation of agile security practices implementation in organisations from practitioners' perspectives.

Grounded theory research is a methodology concerned with constructing theories grounded in data without any preconceived hypothesis (Glaser et al., 1968). It begins with a general idea about a topic and provides in-depth understanding through inductive reasoning. In this research, data analysis involved GT techniques where agile practitioners and cybersecurity professionals were interviewed and the data transcribed, coded, and grouped into concepts then merged into categories. GT has been used in software engineering research to understand human behaviour (Seaman, 1999). Grounded theory research strategy in phase 2 enabled the investigation into multifaceted implications of regulatory policies for building secure agile software from practitioners' perspectives, which led to the development of a novel GT termed policy adherence challenges (PAC).

A secure software development process model has been developed and evaluated in phases 3 and 4 respectively. Phase 4 of this study is more related to case study research, which was conducted with the intention of improving an organisation's secure software development process.

3.2.5 Time Horizon

It defines the time taken to investigate a phenomenon. Ideally, this should be based on the time required to collect and analyse relevant data (Philips et al., 2008). However, adopting a particular time horizon in research is dependent on the resources available and the availability of research sites. Research can be broadly grouped into two categories based on time. They are longitudinal and cross-sectional (Bell et al., 2018).

Longitudinal Study investigates a phenomenon over a longer period. However, this timing approach would not be suitable for this research as it requires repeated investigation to understand the correlation between results which would not be possible within the timescale of completing the PhD research.

Cross-sectional Study describes a research phenomenon been investigated within a specified time period (Olsen & St George, 2004). Participants were selected based on specific variables of interest. This type of studies does not result in cause-and-effect conclusions of the research variables.

In this research, a cross-sectional timing had been chosen with the aim of conducting the research within the 3-4 years PhD study timeline.

3.3 Research Purpose and Outcome

Research purpose explains the rationale behind conducting the research (Robson & McCartan, 2016). It examines the study context and classify it into one of the four approaches. These approaches are exploratory, descriptive, explanatory, and predictive.

Exploratory research was chosen in phase 1 of this study. Choosing an exploratory research approach was appropriate as it examined an area lacking previous investigation. It was also chosen as it helps narrow a broad idea to focus on a specific problem. The study in phase 1 experimented the research methods with a small population before expanding the study in phase 2. Phase 1 developed a taxonomy of agile security practices which laid the foundation for further data collection in other research sites. The taxonomy developed contributed to the empirical evidence on the phenomena for future researchers. The approach provided initial findings to further explore the implications of regulatory policy for building secure agile software in Nigeria. Thereafter, a novel secure agile software development process model was developed and evaluated in phases 3 and 4.

Explanatory research was adopted in phase 2 of this study. Explanatory research is conducted with the aim explaining the relationships between variables. It builds on exploratory research conducted in phase 1 by collecting data to perform an in-depth analysis of secure agile practices implementation and investigate the impact of regulatory policy across different organisations involved in secure agile software development in Nigeria.

The purpose of this research was to develop an explanatory model of secure agile software development process with the hope of influencing security practices and its implementation in software development companies. The research purpose was mainly aimed at solving industry-related problems which aligns with existing literature (Beecham et al., 2013).

The outcome of this research has been applied to improve the software development processes of a particular company. The research outcome is categorised as an applied research (Robson & McCartan, 2016).

3.4 Research Approach

In this study, the different components of the research design were formulated to answer the research questions. The thesis adopted a multi-methods qualitative approach. This approach aligns with the research objectives and four phases which has been briefly described in this section. A detailed description of the activities involved in each of the phases are explained in sections 3.5 to 3.8 and diagrammatically shown in figure 3.2.

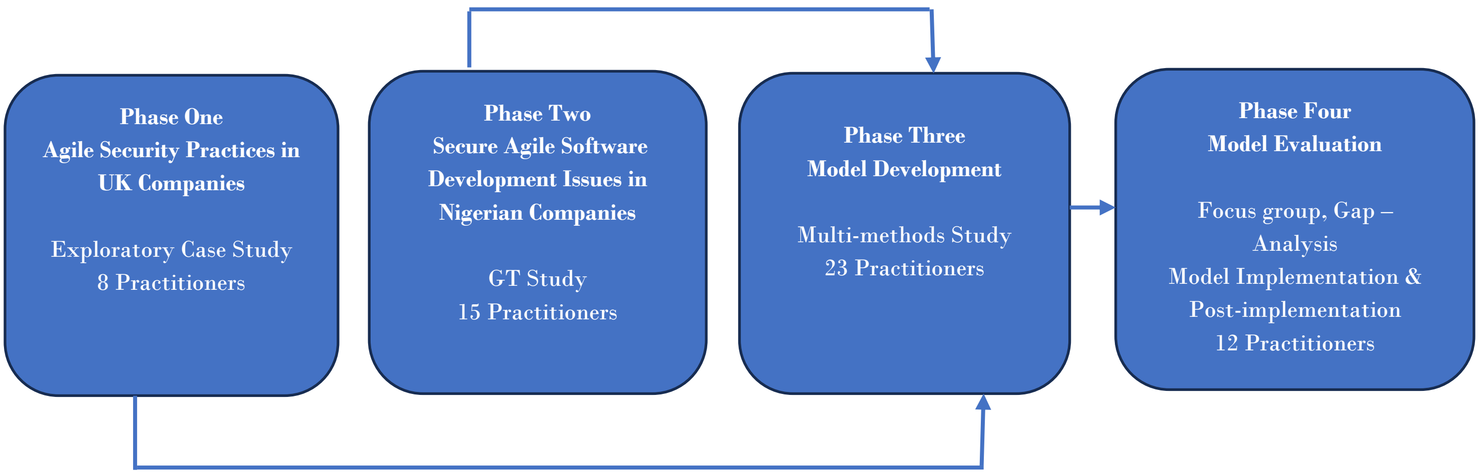


Figure 3.2: Summary of Research Phases

Phase 1

It describes the exploratory case study conducted with agile practitioners and cybersecurity professionals in the UK. Data was collected through semi-structured interviews and analysed by a method informed by Grounded Theory.

Phase 2

It expanded the study conducted in phase 1 through in-depth engagement with Nigerian practitioners. The study involved collecting data on secure agile software development implementation and its challenges in the Nigerian context. Just like phase 1, data collection involved conducting semi-structured interviews and a method informed by grounded theory adopted for data analysis.

Phase 3

It describes the steps taken to develop the secure agile process model. Findings from the two earlier phases of the research served as input for the developed secure agile process model.

Phase 4

This phase conducted a preliminary model validation using a focus group session. Arising from the focus group workshop, a new security ceremony was introduced. The model was further implemented in an organisation. The implementation led to adopting a security role by the company.

3.5 Grounded Theory

Grounded theory is a methodology that involves generating theory from data in an inductive manner. Initially introduced by (Glaser & Strauss, 1967), its focus is on developing new theories rather than validating or expanding existing ones. The grounded theory method, as initially developed by its originators, evolved into two main approaches: Glaserian (Glaser, 1978, 1992) and Straussian (Strauss & Corbin, 1990). They both begin with open coding but deviate in the subsequent stages. Specifically, they differ in terms of research problem formulation, data analysis techniques, and coding methods. The Glaserian approach suggests initiating the research process with a general interest in the phenomenon, while the Straussian method advocates for a more precise definition of the research problem. The literature discusses alternative grounded theory methods known as second-generation approaches, which include Constructivist (Charmaz, 2000, 2006, 2008), Urquhart's GT (Urquhart & Fernández, 2013; Urquhart et al., 2010), and Critical realist approaches (Bunt, 2018; Kempster & Parry, 2011; Oliver, 2012).

In this research, grounded theory was chosen for three primary reasons. Firstly, the research phenomena of secure agile development emphasize individuals and interactions. Grounded theory allows for a detailed examination of practitioners' interactions and behaviours (Gralha et al., 2018; Terpstra et al., 2017). Secondly, grounded theory is well-suited for investigating phenomena such as the intersection of government policies and secure agile software development, where there is limited existing literature. Thirdly, there is growing evidence of the adoption of grounded theory to study the behaviour of agile teams.

For this study, the classic grounded theory (GT) method, following Glaser's approach, has been selected. This decision is primarily based on the ample resources available and the extensive use of Glaser's method in software engineering research (Hoda et al., 2011; Shastri et al., 2021). In this study, Glaserian grounded theory was employed, as the researcher investigated the phenomena without prior knowledge and began with a general understanding of the topic, aligning with previous research (Hoda et al., 2012; Stray et al., 2016). The researcher ensured that codes, concepts, and categories emerged from the collected data, contrasting with the approach advocated by Straussian grounded theory.

The distinctive characteristic of the GT method is its lack of a defined research problem or hypothesis from the start. Instead, the researcher endeavours to uncover the research problem as the primary concern of the participants during the process. Besides GT, another methodology utilized for theory development is action research. Although these two methodologies share similarities, they are distinct in qualitative inquiry. Action research provides an opportunity to generate new understandings through action and can sometimes collect and interpret information more efficiently. Grounded theory, on the other hand, is more explicit about the process of building theory from evidence (Dick, 2007). Therefore, the choice of grounded theory was deemed more suitable for this research because the objective was not to apply an existing theoretical framework but rather to utilize the collected data as the theoretical foundation for the study.

3.6 Phase One - Agile Security Practices in UK Companies

To explore the agile security practices adopted in UK companies, ten empirical interviews were conducted with selected UK practitioners working in seven companies as shown in Table 3.1.

The study adopted a data analysis method informed by grounded theory methodology. Grounded Theory is a systematic methodology which is aimed at theory construction using qualitative data. Due to the paucity of literature on agile security from practitioner perspective, the Grounded Theory methodology allows for the emergence of concepts grounded in data (Glaser & Strauss, 1967). Also, the Grounded Theory is an appropriate methodology in software engineering for constructing theories relevant to practitioners (Adolph et al., 2011).

Table 3.1: Description of participants' & organizations in the study

Code	**Job Title**	**Software Development Experience (Years)**	**Business Type**	**Organization Size**
ITCONCco1_SC1	Security Consultant	11	IT Consulting	Medium-sized
Finco1_SSE2	Senior Software Engineer	17	Financial Services	Large
ITCONCco1_ITS	Manager, IT Security	26	IT Consulting	Medium-sized
CRMco1_FSSD1	Full Stake Software Developer	8	Customer Relationship Management	Large
HEALTHco1_SD1	Software Developer	8	Healthcare Services Company	Small
ITSERVco1_VP-COS1	Vice President, Cyber Operations Security	27	IT Services	Large
LAWENFco1_SC1	Security Consultant	12	Digital Forensics Services	Large
CYBERFOUDco1_ADL1	Cybersecurity Analyst	11	Cybersecurity Solutions	Large

3.5.1 Research Sites - (Phase One)

Appropriate research sites were identified and selected as shown in Table 3.1. Due to the Covid-19 pandemic, interviews were conducted online. The snowballing sampling technique was used to select additional research participants. The selected research sites consisted of agile practitioners working in companies operating in different sectors including IT consulting, CRM Company, Healthcare services and the FinTech industry. Most of them are large companies but there are a couple of medium-sized and one small and medium-sized enterprise (SME). The diversity of the research sites provided richness to the data collected and lends credence to the results.

3.5.2 Data Collection - (Phase One)

In this study, the source of data collection was practitioner interviews. Each of the research participants was sent an information sheet which detailed what the study was about. It further asked for their informed consent to record their responses and indicate their choice of anonymity on a consent form (See – Appendix C). An interview guide containing questions was followed. Each participant was asked the same questions even though the wording and sequencing of the questions were not uniform for all participants. The initial interview guide questions were generated from light literature review and the researcher's experience with investigating agile software development. The initial questions were subjected to several reviews by the researcher. The questions were again modified as data collection progressed following the constant comparison technique of grounded theory. The interview transcripts were manually analysed at the initial stage before moving to the qualitative data analysis software, NVivo. The NVivo software package was chosen because of its ability to facilitate many iterative aspects of the grounded theory process (Hutchison et al., 2010). NVivo can analyse various types of data i.e., text, audio, video, and photos, whereas tools like Leximancer can only be used for text analysis. Also, the qualitative data analysis tool like ATLAS.ti 4.0 seem to work with data files of limited ranges. This means files must be converted to ASCII format before been inputted.

3.5.3 Data Analysis - (Phase One)

After conducting each interview, the researcher listened to the recordings many times to ensure accurate transcription to avoid distortion in meanings (See – Appendix E for sample) (Oates, 2005). Four activities were conducted to analyse the collected data. They are (i) reviewing the data to identify repeated themes, (ii) use keywords to categorize the themes, (iii) code the themes and (iv) categorize the themes through the relationships identified. These activities are summarized in the Glaserian grounded theory as open coding, memoing, constant comparison and theoretical saturation (Glaser & Strauss, 1967; Glaser, 1978).

Open Coding: This stage involved line-by-line reviewing of the interview transcripts with the aim of identifying key themes from the interviewee's responses (Glaser, 1978, 1998). Going through the first interview, 42 codes were identified. A second interview was analysed where 29 codes emerged. Subsequently analysed transcripts had lesser number of codes as many themes were already identified in the initial interviews.

Memoing: Memos were written to capture interviewee's responses and show the relationship between the different concepts and categories. Brief notes were written on different related topics containing verbatim quotes from interviewees pulled together to make-up a memo. Memo writing helped elucidate the researcher's ideas and re-focused further data collection.

Constant Comparison: This research used the constant comparison technique to iterate between data collection and analysis. The data collected was constantly compared within itself as well as other instances of same and similar events. The technique was essentially helpful for refining generated codes and categories.

Theoretical Saturation: It defines the point in the data analysis process when no new categories emerge, and the data categorization is not impacted by adding more interviews (Glaser & Strauss, 1967).

3.6 Phase Two - Secure Agile Software Implementation in Nigerian Companies

Phase two of this research explored secure agile software practices implementation in Nigeria. Practitioners' perceptions on impact of NDPR policy while building secure agile software in Nigeria was investigated. To achieve this, a method informed by grounded theory as has been done in phase 1 was adopted. The method is especially relevant in software engineering research for developing theories relevant to practitioners. The method also enabled deep understanding of a phenomenon in a unique context (Glaser et al., 1968). Since there is lack of literature on the intersection of secure agile software development practices and practitioner's compliance to NDPR in Nigeria. Thus, the use of grounded theory method is relevant, and this makes the study context unique to information systems research.

In exploring the Nigerian software industry, agile practitioners were recruited through personal contacts and professional networks such as LinkedIn as has been done in other existing studies (Sharma & Bawa, 2020; Terpstra et al., 2017). This phase of the research made use of two sampling techniques: snowball (Miles & Huberman, 1994; Patton, 2002) and intensity sampling (Patton, 2002). Snowball technique was used at the initial stage to recruit participants to get a broad perspective on the phenomena been investigated from different stakeholders. Also, the unwillingness of practitioners to talk about sensitive security issues in their organizations especially to strangers due to confidentiality and non-disclosure agreements makes snowball sampling a suitable technique to adopt (Sharma & Bawa, 2020). Intensity sampling was adopted in later stages of the data collection to select some practitioners in managerial roles such as CTOs to provide in-depth knowledge of the phenomenon been investigated. The combination of multiple sampling approaches provided some elements of methodological triangulation. This has offered insights into both the current problems of the phenomenon investigated as well as the motivation for adopting existing practices. The underlying motivations for adopting these practices are difficult to obtain in studies using survey methods (Bass, 2015).

3.6.1 Research Sites - (Phase Two)

Nigeria was chosen because it is unarguably an emerging powerhouse for software development in Africa and considered the largest economy on the continent (Nigeria Bureau of Statistics, 2021; Sowunmi et al., 2016). Most software development companies in Nigeria are located in Lagos and Abuja, hence the decision to limit data collection of this phase of the research to those sites. According to (Ogunyemi et al., 2018; Soriyan et al., 2001), the two cities represent the heart of Nigerian software industry. They have implemented agile methodology and are involved in secure software development using agile methods. The chosen research sites are made up of a mixture of small, medium, and large sized companies as shown in Table 3.2. The research sites include companies operating across key sectors including IT services, Healthcare, Financial services, Manufacturing, Digital solutions, and educational software solutions. Among the firms is also a company registered in the UK where it offers digital innovative solutions with deep financial expertise. Having engaged with diverse research sites, this provided richness to data collected and establishes the legitimacy of the findings.

3.6.2 Data Collection - (Phase Two)

Data collection involved conducting fifteen face-to-face and virtual interviews with Nigerian agile practitioners developing secure software over a period of six months (February – July 2021). Interviews were adopted for data collection in this study because it provided the researchers with in-depth knowledge of the phenomena (Kvale & Brinkmann, 2009). Also, interviewing can provide compelling evidence on individual’s worldview. To protect the anonymity of interviewees and their companies, the abbreviations of their business sectors and job titles were used. The interviewees’ software development experience ranges from 6 to 24 years as detailed in Table 3.2.

Table 3.2: Participants' and organizations description

Participant Code	Job Titles	Experience in Agile (Years)	Interview Date	Interview Location	Business Type	Organization
ESSco1_DE	DevOps Engineer	9	07/02/2021	Abuja	Educational Software Solutions	Medium-sized
ESSco1_PROJ-MGR	Project Manager	16	07/02/2021	Abuja	Educational Software Solutions	Medium-sized
ESSco1_SSE1	Senior Software Engineer	9	14/02/2021	Abuja	Educational Software Solutions	Medium-sized
ESSco1_BEE	Back-end Engineer	9	22/02/2021	Abuja	Educational Software Solutions	Medium-sized
ESSco1_PROD-MGR	Product Manager	13	02/03/2021	Online	Educational Software solutions	Medium-sized
ITSERVco2_SE1	Software Developer	6	05/03/2021	Abuja	IT Services & Consulting	Small
ITSERVco3_QAA	Quality assurance analyst	8	17/04/2021	Lagos	IT Services & Consulting	Small
HSCco1_SSE2	Senior Software Engineer	8	18/04/2021	Lagos	Healthcare Services Company	Large
ITSERVco1_STP-MGR	Security Technical Program Manager	9	11/05/2021	Online	IT Service Management Company	Large
FSSco1_SDE1	Senior DevOps Engineer	11	17/05/2021	Lagos	Financial Services & Solutions	Medium-sized
FSSco1_FLM	Frontline Manager	11	20/05/2021	Online	Financial Services & Solutions	Medium-sized
ESSco1_PROD-MGR	Project Manager	11	13/06/2021	Abuja	Educational Software Solutions	Medium-sized

Table 3.2: Continued

DSco1_SETL1	Software Security Team Lead	10	25/06/2021	Online	Digital Solutions	Medium-sized
ESSco1_CTO1	Chief Technology Officer	24	01/07/2021	Abuja	Educational Software Solutions	Medium-sized
MFGco1_ITA	Manager, IT Security & Operational Risk	17	04/07/2021	Lagos	Manufacturing	Large

The data collection was started by first interviewing six practitioners before the remaining nine were recruited and interviewed subsequently. The rationale behind starting the data collection and pausing it was to allow the researcher to transcribe the early interviews and analyse the data using coding and memo writing. This gave the researcher the chance to collect his thoughts regarding the data collected and helped refine the interview guide. The open-ended semi-structured questioning technique was adopted which afforded the interviewer the opportunity of observing interviewees actions and mannerisms in addition to the verbal information. It also helped the researcher identify issues of real concern to practitioners instead of forcing a topic on them. All the interviews were conducted in English and audio recorded with the permission of the interviewees. Although the Glaserian grounded theory approach advices against recording of interviews, it afforded the researcher the opportunity of capturing all information without losing any details, which was similar to the approach adopted by (Hoda et al., 2012). All the interviews were transcribed, reviewed, and validated. Each of the interviews were begun by explaining the purpose of the session and then assuring the participants of confidentiality. The average duration of the interviews was 45 minutes. Probing questions were used to explore relevant topics not included in the interview guide but that appeared important to the interviewees.

The initial interview guide for this phase of the research consisted of three parts. The first part contained questions on partitioner perceptions on agile methods adoption. The second part explored the security practices adopted by agile practitioners. The third part comprised of questions on interviewees personal details such as educational level and professional background, current job titles and the nature of projects they are currently handling.

As the data collection progressed, the interview guide was modified and refined to focus more on agile security practices, NDPR compliance and challenges confronting practitioners during the development process. A closing question was also added to give interviewees the chance to make comments or add anything else that is relevant which has been missed. Both the initial and consequently adapted interview guides are available in Appendix B.

Another important source of data collection in grounded theory approach used in this research was document reviews. Some participants interviewed in Nigeria provided the researcher access to their company policy documents and guidelines for developing secure agile software. Other documents on secure agile practices were publicly available on some of the participating company's websites. However, it was not possible to access documentation for projects security designs and architectures as all the companies regarded those as highly commercially sensitive.

For direct observation of teams, the approach was not considered due to the cross-sectional time horizon of the PhD research. As participants were engaged for a short period, this technique may be intrusive and alter their behaviour since they were not being observed or are not going to observed over a long time (Stray et al., 2016).

Practitioners interviewed during both phases one and two of the data collection have experience in secure agile software development across various sectors. Thus, the main objective of conducting the data collection was to explore the security practices adopted by practitioners, as well as their impact and challenges in the development process.

3.6.3 Data Analysis - (Phase Two)

Data analysis in the second phase of the research involved the four key techniques of the Glaserian grounded theory approach: coding (open and selective), constant comparison, memoing and theoretical saturation (Glaser, 1978, 1992, 2001).

Coding: This phase of the research adopted the two coding stages of the Glaserian grounded theory approach: open and selective. Open coding is the initial analysis phase of a grounded theory study. The selective coding builds on the initial phase by continuing the analysis process but limited to categories discovered from the core category (Glaser, 1992). The three levels of abstraction in this approach are codes, concepts, and categories. Codes are formed from statements in interview transcripts while groups of codes are known as concepts and the grouping of different concepts form categories.

Open Coding: The open coding process was begun by reviewing each of the interview transcripts line-by-line to allow for codes to emerge from data collected (Glaser, 1978, 1998). Codes (themes) were identified by reviewing interviewee's responses. 36 codes were identified after reviewing the first interview transcript. After analysing the second interview, 13 new codes emerged. At the initial stage, the manual coding procedure was adopted to ensure correctness which also helped reminded the researcher of the social and emotional aspects of the interview (Vaivio, 2012). All transcripts were completely coded as the researcher had no idea at the initial stage which data will be relevant as the analysis progressed. As the dataset grew, subsequently transcribed data was imported to the Nvivo-12 qualitative data analysis tool (Edhlund & McDougall, 2019).

Selective Coding: The moment a core category has been established; the researcher moved to selective coding. It is a coding technique that is focused on selectively coding only variables in interview transcripts related to the core category. While open coding was time consuming, selective coding was quicker because the focus is on categories related to the core. It was also less challenging because the constant comparison method has become more familiar to the researcher. As the selective coding progressed, less categorisation was noticed highlighting that the data collection and analysis is reaching theoretical saturation (Glaser, 1992).

Memoing: This technique represents the written accounts capturing the relationships between codes and concepts as they develop into categories (Glaser & Strauss, 1967). Based on codes identified, statements were written on each topic with embedded quotations as supporting evidence. Adhering to the recommendation by (Glaser, 1978), analysis process was often paused so that the researcher could reflect on the data collected. Memoing have helped to converge research ideas as more transcripts were being analysed. This technique represents an important stage of the theory generation process (Adolph et al., 2011).

Constant Comparison: This stage involves iteratively comparing between and amongst different interviewees, their job titles, and organizations. It starts by forming codes from interview statements, grouping codes to form concepts and combining related concepts to create categories (Glaser, 1992). As the analysis progressed, the constant comparison technique was used across the dataset. Codes from the same interview were continuously compared with others from subsequent transcripts. 584 codes were generated at the end of the open coding process.

Figure 3.3 illustrates the data analysis process that led to development of the study grounded theory. The items in the boxes on the left represent different concepts identified in interview transcripts. The middle boxes represent categories which were derived through the constant comparison technique of GT. Finally, the box on the right labelled “NDPR Implementation Challenges” is the GT core category derived from iteratively grouping of identified categories.

To further illustrate the data analysis process, figure 3.4 provides a snapshot of the emergence of the category “Paucity of Indigenous Hosting Companies.” The category describes some of the challenges confronting the agile practitioner in their efforts to adhere to a new regulatory policy.

The analysis process began by exploring the interview transcripts, then gathering the key points from the data. Two-to-three-word phrases were used to summarize the points known as codes. The key points were iteratively grouped to form higher levels of abstraction known as codes, concepts and finally category.

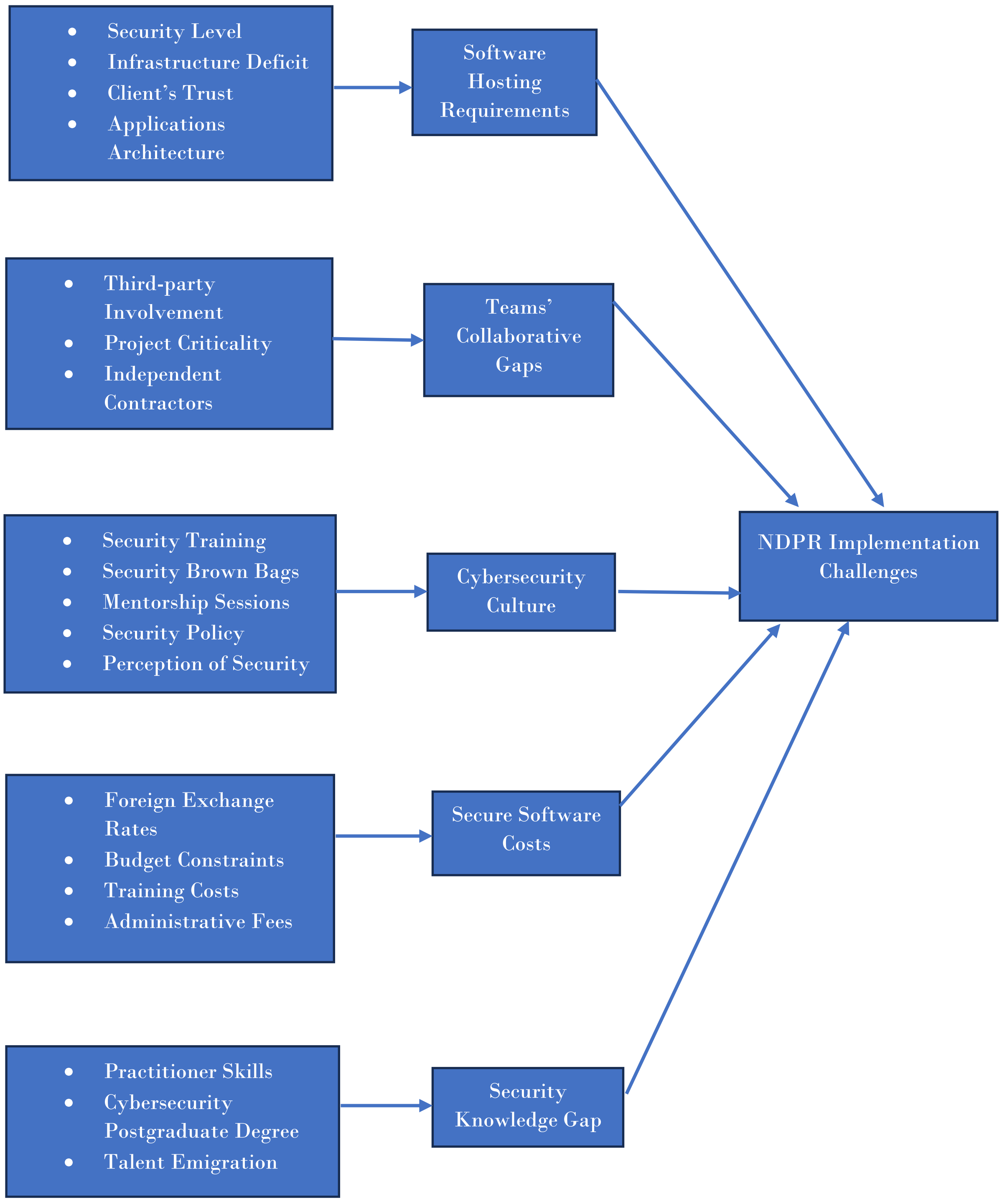
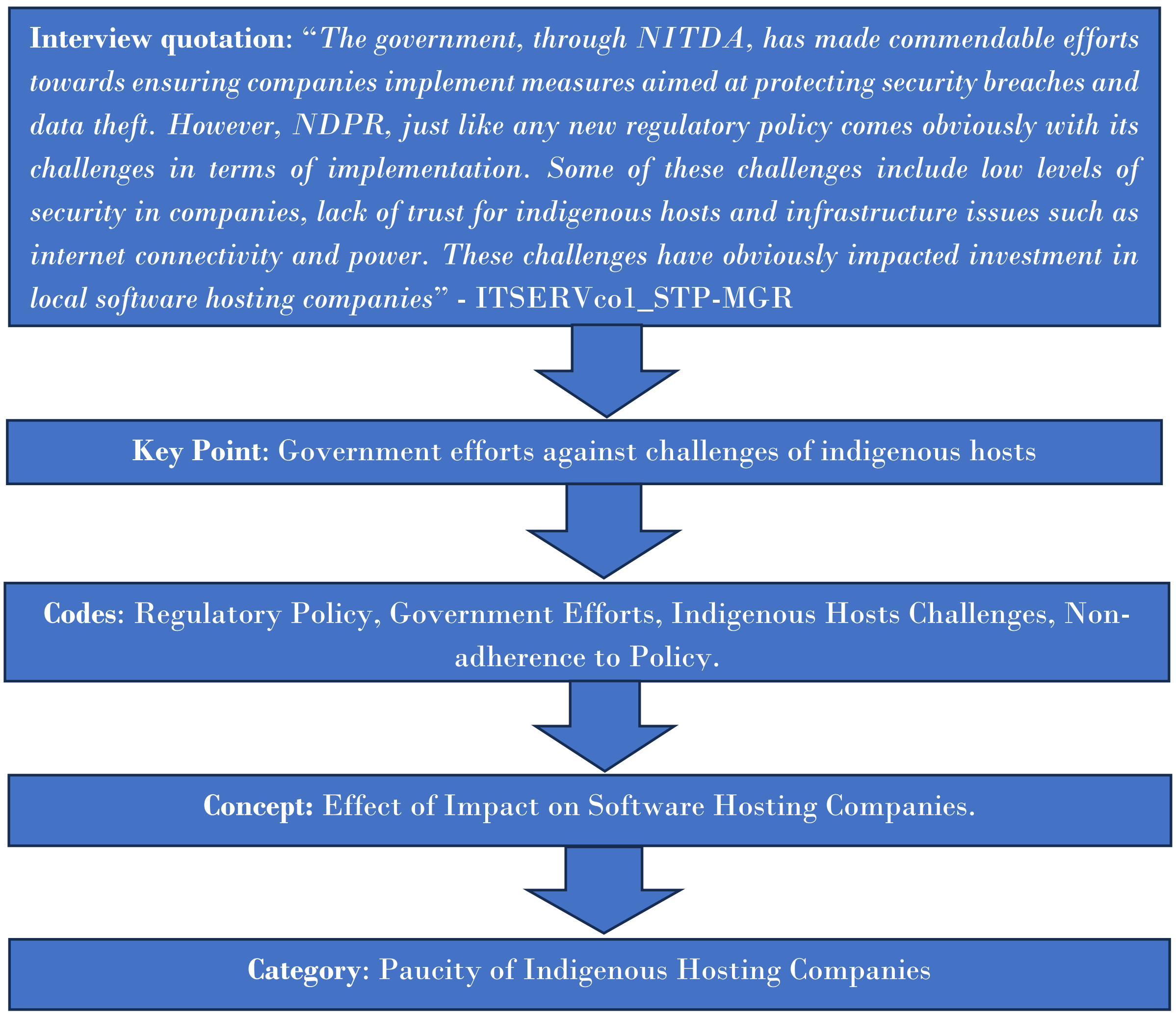


Figure 3.3: GT Data Analysis (Ardo et al., 2023)



Interview quotation: "*The government, through NITDA, has made commendable efforts towards ensuring companies implement measures aimed at protecting security breaches and data theft. However, NDPR, just like any new regulatory policy comes obviously with its challenges in terms of implementation. Some of these challenges include low levels of security in companies, lack of trust for indigenous hosts and infrastructure issues such as internet connectivity and power. These challenges have obviously impacted investment in local software hosting companies*" - ITSERVco1_STP-MGR

Key Point: Government efforts against challenges of indigenous hosts

Codes: Regulatory Policy, Government Efforts, Indigenous Hosts Challenges, Non-adherence to Policy.

Concept: Effect of Impact on Software Hosting Companies.

Category: Paucity of Indigenous Hosting Companies

Figure 3.4: Emergence of GT Category (Ardo et al., 2023)

Theoretical Saturation: In this research, Glaser's idea of theoretical saturation was adopted. It describes a point in the data collection process when new categories no longer emerge. At this point, evidence from the data collection can be said to have converged. Conducting additional interviews was stopped when analysing more transcripts does not lead to the emergence of new categories and no new information seemed to emerge. Thus, conducting and analysing of additional interviews had no impact on categorisation (Glaser & Strauss, 1967).

3.7 Phase Three - Secure Software Development Model Creation Process

The secure software model creation process involved interviewing a cross section of practitioners ranging from technical (software developers, security specialists, systems analyst, business analyst) to non-technical (decision makers/managers, project managers) to understand different security practices adopted in their organisations. The identified practices across the diverse research sites in phases 1 and 2 were collated, analysed, and modelled using swim lane diagram concept. Also, the constant comparison technique was used to include other findings.

Swim lane diagrams were selected to depict the identified practices because they have emerged as the primary modelling tool for planning business process reengineering and enterprise-wide software development (Davenport, 1993; Hammer & Champy, 2009). Furthermore, swim lane diagrams were selected due to their efficiency in conveying information to stakeholders and verifying the efficiency of business processes, as compared to non-swim lane diagrams (Jeyaraj et al., 2014). Considering the business process as a series of sequenced activities that an organization performs to achieve a valuable output, the swim lane diagram allows for the visual representation of activities in business process models, enabling the definition and analysis of business processes (Aldin & de Cesare, 2011; Ramias & Rummler, 2009).

To illustrate the flow of information across various research sites, each horizontal lane on the swim lane diagrams represents an agile role. Then, security activities (agile ceremonies & agile artefacts) were positioned in the corresponding lane. Arrows were used to show the sequence of activities from start to the end point of an iteration (Bera, 2012). Different swim lane diagrams were used to represent the baseline state (current/existing model), proposed state (pre-intervention model), and the final state (post intervention in Phase 4) to show the effect of a secure-by-design model in influencing the agile software development process.

The swim lane diagrams were compared to visually show the differences after each intervention as presented in chapter 7. Thus, swim lane diagrams were adopted in this research to understand the relationships and dependencies between the different security practices. This research identified 24 security practices, organized into the six - software development life-cycle phases: planning, requirements, design, implementation, testing, and deployment.

3.8 Phase Four - Secure Software Process Model Evaluation

The evaluation of the secure software process model was carried out in two stages: First a preliminary evaluation conducted with five agile and cybersecurity experts through a focus group workshop in the UK. Second, the proposed model was evaluated at HealthCo1. HealthCo1 is a Nigerian-based company, specialised in building healthcare software solutions. The company provides patient management solutions, laboratory systems and primary healthcare data record portals. HealthCo1 also provides consultancy services to many States' Primary Healthcare Development Agencies across Nigeria. The company's development team is based in Nigeria; however, they collaborate virtually with some practitioners working in United States of America (USA) and Germany.

3.8.1 Model Evaluation Structure

The evaluation of the proposed secure software process model consisted of eight main activities:

1. Preliminary evaluation with five agile and cybersecurity professionals through a focus group session in the UK.
2. Pre-workshop meeting with HealthCo1 management in Nigeria.
3. Workshop on secure agile software development with HealthCo1 development team in Nigeria.
4. Focus group session involving presenting proposed research model and conducting a gap analysis of HealthCo1 processes conducted in Nigeria.
5. Recommendation of action to improve the security of the software development processes at HealthCo1, based on the findings of the gap analysis
6. Implementation of the agreed security practice
7. Post-Implementation interviews with HealthCo1 practitioners.
8. Updating of the proposed model based on participants' feedback.

3.8.2 Preliminary Model Evaluation – Focus Group Workshop

A focus group workshop involving five agile and cybersecurity professionals in the UK was conducted to evaluate the proposed model. An interview guide containing questions related to participants understandability of the model as well as the relevance of presented practices. The workshop session lasted 45 minutes and had five participants. The workshop was recorded with the participants consent which was later transcribed and analysed. The workshop also involved a discussion of the research findings from phases 1 and 2, as well as a general description of how the proposed model was developed. Prior to the workshop, a participant information sheet was sent to each participant (See – Appendix D). The outcome of the workshop led to introducing the "Compliance Sprint" to the research model.

3.8.3 Secure Software Process Model Implementation at HealthCo1

The model evaluation phase was aimed at influencing existing practices in the selected software development companies. Permission was granted by the management of Healthco1 for the evaluation of the model after an initial engagement meeting with the CTO and the Security Team Lead. The company requested an undertaken from the researcher to delete all interviews after analysing the data. Also, both the company and the engaged practitioners be made anonymous in reporting the research findings.

At the initial meeting, findings from earlier phases of the PhD research were presented to the CTO and the Security Team Lead (Ardo et al., 2021; Ardo et al., 2022). After the initial session, a workshop on agile security practices identified in various research sites was conducted. At the end of the workshop, team members discussed the practices they are familiar with as well as the one's related to their job roles. The workshop was attended by 7 participants – 1 Security Team Lead,1 Project Manager, 2 Senior Software Engineers, 1 Software Security Analyst and 2 Software Developers. Based on the interest generated during the workshop as well as the enthusiasm of the participants, the researcher was able to convince the management to further engage the same team in a focus group meeting.

The initial focus group meeting was held with the same team where the developed practice-based model was presented in detail. The researcher explained how data was collected and analysed, the model development process and discussed the findings of the preliminary model evaluation (Ardo et al., 2022). The Security Team Lead and Project Manager led discussion on existing practices adopted for secure software development at Healthco1. A second focus group meeting was conducted with 4 participants – 1 Project Manager, 1 Senior Software Engineer, 1 Software Security Analyst and 1 Software Developer. The researcher and the team agreed on two practices for implementation to an on-going project and report if any impact on their development process. The security practices implementation was agreed over 3 iterations with a progress/monitoring meeting with 2 – 3 practitioners after each iteration. At the end of the 3rd iteration, 5 interviews were conducted with 3 practitioners over a period of 10 weeks. Post-implementation review was conducted with all the 4 practitioners present during the second focus group meeting when the two implemented practices were chosen for the model application process. The model application process used the principles of research impact evaluation to study the impact created to the company (Reed, 2016). Thus, the proposed practice-based model underwent numerous refinements to reach the final design state.

3.9 Summary

This chapter summarized all the methodological issues of the research. The research adopted a qualitative research design, with a pragmatic worldview. The research strategies included grounded theory and case study. For data collection, semi-structured interviews were conducted with agile practitioners and cybersecurity professionals. An approach informed by Grounded Theory method was used for data analysis. A secure software development process model would be developed and evaluated.

Chapter 4

Taxonomy of Secure Agile Software Development Practices

4.1 Introduction

This chapter presents findings from exploratory the case study (phase 1) conducted to explore security practices adopted by agile software development practitioners **(Objective 1)**. The chapter presents an in-depth description of secure agile implementation in practice, as described by practitioners.

This chapter presents a novel taxonomy of secure agile practices derived from study findings. The security practices have been categorised into roles, ceremonies, and artefacts. Thereafter, secure practices were examined in regulated environments – FinTech and healthcare sectors. This chapter answered the first question of the PhD research which sought to explore the security practices adopted by agile practitioners during software development.

4.2 Secure Agile Software Practices

This section presents the security practices adopted by practitioners. The practices have been categorised according to agile use in software development organisations. Thus, these categories are artefacts, roles, and ceremonies.

4.2.1 Secure Agile Artefacts

The findings of this study shows that organisations adopt more artefacts than roles and ceremonies. From the analysis of interview transcripts, the use of automated test tools is common among interviewed practitioners. The reliance on these tools indicates a somewhat straightforward absence of other testing methods as reported by this study's general findings.

The identified tools were either used for vulnerability testing or standard software testing. There are also practitioners that combine manual and automated testing at different stages of software development. Thus, the ten artefacts identified in this study are vulnerability assessment tools, code verification tools, API testing tools and Git-Hub test platform. Other artefacts include security backlog, creating misuse cases, risk assessment, security baseline standards, security test plan templates, and security audit checklists.

Vulnerability assessment tools

The specialized testing tools mentioned by practitioners in this study were mainly used for vulnerability assessment, scanning and management. According to ITCONCco1_SC1: "*We use a tool known as OpenVAS and what it does is to give you a free vulnerability assessment test…*" (Security Consultant, Company E). Basically, what the user needs to do is to supply the port numbers and other details and the tool does the testing. ITCONCco1_SC1 have highlighted the advantage of using the OpenVAS tool when he said, "*OpenVAS is pretty easy and freely available to use as well…*" (Security Consultant, Company E). Apart from the OpenVAS tool, other software companies use software tools such as Qualys, Rapid7 and Nessus to conduct vulnerability assessment. Among the reason's software practitioners mentioned for using these tools are providing specialized functions such as automating network auditing, identifying threat actors through cloud-based solutions, and providing other penetration testing services such as website scanning to identify potential vulnerable spots, confidential data searches and compliance checks as well. According to ITSERVco1_VP-COS1: "*vulnerability testing using Qualys or Rapid7 or Nessus takes place at all points, so we know that we are not introducing any vulnerabilities …*" (VP, Cyber Operational Security, Company G). The adoption of automated testing tools in other companies is informed by the diverse capabilities of open-source frameworks such as Kali Linux. According to CYBERFOUDco1_ADL1:

> "*We use an array of different tools for security vulnerability checks, most of which can be found in Unix … Kali Linux is one which has a whole platter of tools in that to analyse and exploit software security issues…*" (Cybersecurity Analyst, Company A).

Another alternative to the use of vulnerability assessment tools is security patching. It defines a process of identifying vulnerabilities and improving software functionalities by proactively guarding against attacks. It is an important theme in company policy documentation as explained by ITSERVco1_VP-COS1, "*part of our company security policy document are* *guidelines on security patch management to handle the ever-changing security vulnerabilities during software development.*" (VP, Cyber Operational Security, Company G).

Code verification tools

In software development, the security of codes is very important. In the phase 1 study conducted with agile software practitioners, security of codes relies on either security documentation (artefact) or using tools in review session (ceremony/meeting). In companies D, E & G, they use different tools for verification of code security and performance. Some of these tools include Black Duck, Syno Cube and Snyk. ITSERVco1_VP-COS1 stated: "*We use code nursing tools like Black Duck or Syno cube tool alot to actually go through and validate the code from a security perspective...*" (VP, Operational Security, Company G). The tools been open source is among the reasons for their adoption by practitioners.

API testing tools

There are certain practitioners that favour combining the use of certain testing tools for security checks on parts of their application like APIs with manual software testing. Postman is an API testing tool which enables the automating of various forms of tests such as functional tests, regression tests and end-to-end tests, among others. ITCONCco1_ITS stated: "*We use Postman to do certain checks on APIs ... we use that to set up automated tests to prevent human errors…*" (Manager, IT Security, Company E). Moving to the implementation phase, ITCONCco1_ITS prefers to perform software test manually. "*I don't use any tools for security tests on codes because coding analysis needs to be done manually by going through the code and working that in…*" (Manager, IT Security, Company E).

Git-Hub test platform

In standard software testing, GitHub is a common collaborative code hosting platform that allows co-located practitioners to work together on projects. Some practitioners prefer using specialized plug-in on the GitHub platform to perform vulnerability checks. According to CRMco1_FSSD1: "*We use a GitHub plug-in called Snyk for vulnerability checks. Previously, we used Greekeeper but recently moved to Snyk because we felt it is a bit better…*" (Full-Stack Software Developer, Company I). The GitHub platform has various features for things like code verification and modelling of threat actors. This is explained by Finco1_SSE2 who said, "*Basically some tools are used for code verification from the security perspective and others for internal tasks to check and identify imposters …*" (Senior Software Engineer, Company D).

Security backlog

The security backlog documentation describes all security-related activities contained in the product backlog. According to ITSERVco1_VP-COS1: "*All our security activities are prioritized in a sort of security backlog document … it is normally updated based on a project's peculiar requirements*" (VP, Cyber Operational Security, Company G).

One of the challenges of secure agile projects is the dynamism of requirements changes to the backlog throughout the SDLC. The requirement changes sometimes leads to increased pressure on practitioners to meet project deadlines. Practitioners describe it as a major drawback in their quest to deliver secure software on schedule. CYBERFOUDco1_ADL1 explained: "*I have had instances where new requirements were added to the backlog which we thought we could get it done but couldn't which affect the delivery date*" (Cybersecurity Analyst, Company A). There are other unforeseen problems which may affect software delivery timelines. ITSERVco1_VP-COS1 described a case where a penetration test failed two days to delivery date: "*I can remember I was involved in a project that should go live on Christmas day, however, on the 23rd of December the Pen test failed which caused one month delay*" (VP, Cyber Operational Security, Company G).

Creating misuse cases

Misuse cases are sometimes referred to as evil user stories. They are used to understand how software may be exploited by attackers. It is often a good practice to review threat scenarios after each iteration so malicious behaviours are identified. To define evil user stories, CYBERFOUDco1_ADL1 described it:

> "*I would liaise with the business to determine what they want out of the project first and then it would be dependent on it that threat scenarios of attacking the system can be determined*" (Cybersecurity Analyst, Company A).

Creating misuse cases are generally considered a team activity as described by practitioners. The first task is defining the user stories for each requested feature. The user stories get sent to clients for their feedback before any development process is initiated. After receiving feedback from clients, practitioners need to analyse the requested features by thinking of all possible attack scenarios. As explained by CYBERFOUDco1_ADL1:

> "*We have to put on our attacker hats to think of all possible ways that the software maybe attacked. Sometimes, there might be delays especially when dealing with SMEs as external security specialist are consulted before development starts and they keep been consulted until the finished product is delivered…*" (Cybersecurity Analyst, Company A).

Creating misuse cases as a security practice can sometimes be undertaken through a workshop session, involving different roles such as: self-organizing team, security specialist or security team according to ITCONCco1_SC1 when he said: "…*security requirements gathering workshop stakeholders include engineers, security specialist or at-times security team members. We share ideas of envisaged unauthorized access and attacks possible*…" (Security Consultant, Company E).

Risk assessment & mitigation

Practitioners involved in security testing who participated in this study discussed very little about the techniques they use for mitigating or managing risks during software testing. According to CRMco1_FSSD1, "*We use slow increments and the maker checker approach to develop our projects… You are not allowed to develop, push, approve and merge your code all by yourself…*" (Full-Stack Software Developer, Company I). Apart from CRMco1_FSSD1, Finco1_SSE2 also described managing risk by relaying on the iterative feature of agile software methods. "*The short lifecycle and iterative way of development minimizes the risk of project delay by been able to provide exactly what the customers requested*" (Senior Software Engineer, Company D). The second agile risk management technique described by practitioners in this study is a method where a company uses the developed software internally before pushing it out known as "Dogfooding".

Security standards

Security baseline standards are another important artefact discovered from analysing empirical interviews. ITCONCco1_SC1 explained: "*We adhere strictly to the government-regulated standard requirements like the GDPR … Additionally, we have our company baseline standards like basic stuff such as encryption and using strong passwords…*" (Security Consultant, Company E). These baseline security standards help organizations protect their critical resources such as servers and workstation from cyberattacks.

Security test plan

Security test plan templates are categorized under the technical group, while baseline standards are sometimes classified as either High-level or Technical. ITCONCco1_SC1 explained: "*We adopt test plans that will ensure our software operates securely…*" (Security Consultant, Company E). The objective of adopting a template for the testing phase is to ensure a process of identifying security threats and the elimination of issues specifically on the safety and integrity of the software.

Security checklist

In ITSERVco1, ITCONCco1, and CRMco1, experienced practitioner having job titles such as chief technology officers (CTO) and IT security manager are mostly responsible for issuing the checklist used in their company for information security and audit checks. According to ITSERVco1_VP-COS1: "*It is part of my duties to issue security audit checklist to the security and development teams*…" (VP, Cyber Operational Security, Company G).

4.2.2 Secure Agile Ceremonies

In the phase 1 study conducted with practitioners, four secure agile ceremonies were identified. They include threat modelling sessions, secure code reviews, brainstorming sessions, and conducting security trainings.

Threat modelling session

Integrating the threat modelling activity by thinking like an attacker during the design of secure software helps to identify points of threats. For example, in company D: "*We think evil sometimes thinking of any gaps or possible hole when we try to integrate our systems to third parties like Klarna, Payon and WorldPay*…" (Senior Software Engineer, Company D). Threat modelling has not been widely adopted into the secure agile development process. However, there is still evidence of the different methods of the practice in some of the research sites of this study. According to Finco1_SSE2: "*We have dedicated sessions which are attended by our security experts, developers where the main activity is to come-up with a list of assets at risk of been attacked*." (Senior Software Engineer, Company D). Apart from manually generating the list, tools such as STRIDE has been used to simplify the process. Finco1_SSE2 added, "*We* *now use STRIDE for generating our threats list which makes it much more easier now than before*." (Senior Software Engineer, Company D).

Conducting secure code reviews

Agile teams involved in this study have also used security sessions such as code review meetings to implement security. For example, senior engineers or security team members maybe tasked to handle security in software codes. Finco1_SSE2 explained: "*We have peer reviews for the code where you develop the code, and another colleague verifies it checking for security aspects."* Similarly in company I: "*I work with a group of very experienced people, 10 years and above that checks for security vulnerabilities as we develop software products & they also holds sessions to reflect on security issues*" (Full-Stack Software Developer, Company I).

Brainstorming sessions

This is where the self-organizing team collaboratively work with the security roles to identify and deliver features defined for different user roles. "*We work as a team to deliver the product as different security features are defined for different users like administrator, manager, and the others…*" (IT Security Manager, Company E). However, in Company C, the security specialist only works with the CTO to identify and build security critical features. According to a software developer, *"…usually the CTO and security specialist handles all the security critical features during the development process*" (Software Developer, Company C).

In brainstorming sessions, various security-critical decisions are undertaken by the security roles in collaboration with agile teams. Such decisions made include adopting a security framework.

> "*In adhering to secure-by-design principles of software engineering, we compare different peer-reviewed frameworks… we review how actively maintained it is, how many times it has been breached and how quick was it to be fixed*?" (Cybersecurity Analyst, Company A).

Choosing an appropriate architectural design to implement in projects is another important decision undertaken during brainstorming sessions.

> "*…we look at the security requirements and then we look at the architectural designs that are available… maybe a website that has to do with payments, we go for MV-6 and for real-time system we adopt event driven architectural design…*" (Manager, IT Security, Company E).

Conducting security trainings

Evidence from data collected have shown the widespread usage of both formal and informal meetings for raising security awareness to develop a cybersecurity culture. Agile teams develop cybersecurity culture through the norms and habits exhibited by practitioners during the secure development process. Therefore, practitioner experience, education, creativity, and understanding are prioritized over documented artefacts. For instance, ITSEVco1-VP-COS1 explained: "*I will say our culture evolves around our understanding of security tasks as a team. So, individual security competency, attitude and behaviour are more important to us…*" (VP, Cyber Operational Security, Company G). Apart from the usual methods of knowledge sharing in the agile process such as pair programming, scrum meetings and retrospective, security training is an important practice.

Company G conducts various security trainings for its staff. There are specific trainings mainly targeted at new recruits joining the company. ITSEVco1-VP-COS1 said: "*Security training is a very regular thing especially for staff coming on-board. So, it's part of the induction process*" (VP, Cyber Operational Security, Company G). In addition to the new recruits training, there are yearly refresher programs or those required for handling different projects. ITSEVco1-VP-COS1 further explained the various security training available for staff, "*We have standard sort of yearly mandatory security trainings. There are also trainings for projects as there might be different security considerations between your previous project and the current one*" (VP, Cyber Operational Security, Company G). There might be further trainings designed due to lessons learnt from security incidents, "*We also do conduct security training because something happen couple of times and therefore people need to be trained so the mistake is not continued forward…*" (VP, Cyber Operational Security, Company G). However, in company C security trainings are mostly provided to practitioners with security roles. For instance, HCSco1_SE1 described the sorts of generic trainings he had attended in the past: "*I have attended trainings on good practices in setting passwords. I was told not to share passwords and confidential information about patients but not security-specific training…*" (Software Engineer, Healthcare Services Company C).

4.2.3 Secure Agile Roles

Typically, agile teams are composed of three roles which are product owner, scrum master and self-organizing team members. In this study, the following security roles were identified.

Security specialist

This study discovered another role known as the security specialist. As explained by CRMco1_FSSD1: "*In my organization we have an expert that handles issues related to security design and architecture and also responsible for implementing it…*" (Full-Stack Software Developer, Company I). The security specialist is responsible for handling all security related tasks, but their function sometimes overlaps with that of the product owner to ensure that the software development process does not impose any security risks. CYBERFco1_ADL1 mentioned that their security officer does more than just handling security but performs other tasks for the company: "*So, the security guy is also involved in software development ... the person does PR and some other assigned tasks for us…*" (Cybersecurity Analyst, Company A).

Penetration tester

There are situations where a company has individuals with specialist knowledge dedicated for handling all cybersecurity related issues of the software development process. According to Finco1_SSE2:

> "*We have a group of experts when it comes to security to act like internal hackers to try to expose any holes in the system by simulating cyberattacks & report these flaws…*" (Senior Software Engineer, Company D).

Thus, security roles vary across different projects and organizations. This study found that a lot of decisions about security are mostly handled by those having security tasks assigned to them. According to CYBERFco1_ADL1: "*Security is handled by …… practitioners having job titles like security specialist, penetration testers or security analyst…*" (Cybersecurity Analyst, Company A). While having a security role within the software development team is one way of doing it, in some other organizations all stakeholders are involved in security decision as explained by ITSERVco1-COS1, "*All stakeholders are required, you got the Business, the Development Lead, Legal team to handle legal requirements that might need to be met…*" (VP, Cyber Operational Security, Company G). Security decisions are mostly considered as business decisions because there are some financial impacts to them.

At the security requirements gathering phase, a better way of developing secure application will be to involve all stakeholders. Therefore, all the stakeholders need to know the threats and vulnerabilities in their domain. ITSEVco1-VP-COS1 indicated that when he said:

> *"So, it's engaging the right stakeholders at the right time and it's not just a security talking to a techy and saying these are the requirements. It needs to be understood across the board…*" (VP, Cyber Operational Security, Company G).

The findings of this study do not point to a specific security role for each of the SDLC phases. However, the software security specialist, software developer and security teams traverse from the requirements to the release phases of the development lifecycle.

4.3 Secure Agile Software in Regulated Environments

Developing secure agile software in regulated environments such as the Financial Technology (FinTech) and Healthcare industry are regarded very sensitive. Software companies are faced with many regulations that must be strictly adhered to avoid losing money in damages and litigations. In UK as well as other European countries, practitioners handling sensitive client's data are mandated to adhere to legislations such as the GDPR. Thus, practitioners in critical sectors such as FinTech and Healthcare sectors should be guided by the principle of privacy, safety and security, and quality in secure software development.

4.3.1 Regulations in FinTech Industry

Software security is central to the smooth operation of the fintech industry. This has made it imperative for practitioners to consider and adhere to many standards and regulations to reduce incidences of cyberattacks. As explained by Finco1_SSE2: "*There are consequences of not complying to standards such as PCI DSS, PCI PTS, and PCI PA-DSS*" (Senior Software Engineer, Company D). In addition to these standards, there are other regulations that prevents service providers saving information longer than necessary.

> *“As a company, we can’t expose or save credit card information at our own side for a long time. For example, right to be forgotten, you can’t just save the credit card CVV information such as name, address and so on unless it is needed for the system”* (Senior Software Engineer, Company D).

Therefore, security is one aspect that must be considered to prevent hackers gaining access to cause serious destructions.

4.3.2 Regulations in the Healthcare Sector

In the healthcare sector, the most important asset held by companies are patient data. The health record of all patients especially those with critical ailments needs to be well managed. For instance, in Company C which handles the records of cancer patients that needs to be surveillance, security of the patient management system is of top-most priority. As described by HCSco1_SD1:

> “*Imagine if an attacker comes into our system and changes the appointment dates for patients waiting for Chemotherapy, Radiotherapy treatment, these will be a disaster as lives might be lost and we will also be sued*” (Software Developer, Company C).

Considering the criticality of healthcare software systems, agile practitioners consider security from the basics of setting passwords to securing network servers. According to HCSco1_SD1:

> “*We consider even basic stuff like encrypting passwords. If we are retrieving passwords, you should know that you don’t return it to the end user unencrypted. Then things about security of our designs, architecture and securing servers are for the security specialist…*” (Software Developer, Company C).

The importance of securing healthcare records can never be overemphasized. Just like in FinTech companies, regulations must be adhered to ensure secure software development. HCSco1_FED further explained: “*Ofcourse we follow the standards proscribed in the industry in order remain as consultants for the NHS...*” (Front-End Developer, Company O).

4.4 Secure Agile Software Development Challenges

This section presents the challenges of developing secure agile software as described by practitioners. The challenges discussed in this section do not fit onto the taxonomy of security practices presented in figure 4.1 as they are not process activities of the SDLC.

4.4.1 Challenges in Conducting Security Trainings

Companies do face a lot of challenges in their efforts to provide security trainings for staff. One of the most common issues are having junior engineers not showing interest in the trainings. For example, according to ITCONCco1_ITS: "*One thing is some junior practitioners are not quit passionate security trainings*" (Manager, IT Security, Company E). Similarly, there are issues around experienced practitioner not mentoring junior engineers to acquire the right skill. ITCONCco1_ITS further added, "*Another problem is having managers in the company who are not necessarily training or teaching their staff the same competences they have got with regards to security"* (Manager, IT Security, Company E).

4.4.2 Lack of Teams' Collaboration

Team collaboration describes the methods adopted by practitioners to communicate and work together to achieve project goals. In this study, the concept of "security collaboration in agile practices" refers to how security is involved in secure agile ceremonies such as protection poker, threat modelling, and secure code review. However, there is little collaboration on security among team members as most times security related issues are handled by the specialists. According to ITCONco1_SC1, for SMEs, a single individual within the software team maybe assigned the responsibility of handling all security issues "*we have one of the team members who is a security specialist and so focuses more on such things in the company…*" (Security Consultant, Company E). While in other organizations, a security team will be in charge as explained by Finco1_SSE2: "*we have some experts when it comes to the security who act like…internal hackers to try to explore any vulnerabilities in the system*" (Senior Software Engineer, Company D). There are still organizations like Company I where developers are involved in secure practices even if they are not really contributing much to security discussions: "*We are involved at some point although the initial identification of threat assets is being done by the security guys*" (Full-Stack Software Developer, Company I). The involvement of developers in security activities have proven to improve collaboration between teams as well as raise awareness on security.

Company size is another determinant for considering who handles security within an organization. LAWENFCO1_SC1 explained: *"I think company size play a huge role in how many people will be dealing security exclusively or dealing with it as part of their role."* (Security Consultant, Company J).

4.4.3 Lack of Internal Team Involvement

Collaboration between teams and clients can take different forms. In company C, all issues that has to do with security of software projects are handled by a partner organization, "*My company recently signed a contract with XXX for the maintenance of our security critical software systems although our security specialist participates in their meetings both during development and retrospective to discuss security exclusively …*" (Software Developer, Company C). The reason for most security choices and decisions are unknown to internal team. Thus, not all security issues can be answered by internal team in customer demo session and so the external team must be always involved.

Finco1_SSE2 corroborated the assertion in company C with regards collaboration with contracted external security teams.

There are situations where contracted third parties take part in all demo session with clients. According to Finco1_SSE2:

> *"XXX have a professional team of software and cybersecurity engineers, so our company pay them to handle security issues. Our internal team here mostly don't have much inputs especially towards the end of the development."* (Senior Software Engineer, Company D).

Thus, there are reported challenges of external teams getting more involved in secure software development.

4.5 Taxonomy of Agile Security Practices

This section provides a taxonomy of the security practices identified from analysis of interview transcripts. In this thesis, the term "taxonomy" refers to the systematic categorization of secure agile practices identified across different research sites. Figure 1 illustrates the diverse security practices that have been presented as a result. The taxonomy aims to enhance the comprehension of security practices adopted in organisations during the software development process. It is derived from interviews conducted with practitioners. The taxonomy developed in this research holds significance, as existing taxonomies of agile practices have been created based on overviews from the literature (Diebold & Dahlem, 2014; Jalali & Wohlin, 2010; Neumann, 2021; Williams, 2010). While diverse security practices exist in agile methods (Rindell et al., 2021; Rindell, Ruohonen, et al., 2018), the taxonomy created in this thesis offers a systematic framework for practitioner activities. We employed a qualitative method for creating the taxonomy since there are presently no numerical scales available for classifying agile methods and their components (Usman, Britto, et al., 2017). Thus, the taxonomy describes security practices adopted by organisations in secure agile software development process.

Figure 4.1 shows the grouping of the identified security practices according to agile use in organisations - roles, ceremonies, and artefacts. The horizontal lanes, oval and square rounded shapes represent roles, artefacts, and ceremonies respectively. Figure 4.1 shows the dominance of the security specialist throughout the software development process. Out of the 11 activities undertaken to develop a secure software, the security specialist is involved in 9 either exclusively or in collaboration with the senior or penetration tester. The secure development process begins with conducting security training for the different roles and ends with security checklist documentation, where the security specialist and senior developer collaborate to ensure all requested work items have been integrated. The borderline activities such as risk assessment and secure code reviews illustrates work items that are collaboratively undertaken by different roles. While activities within a specific horizontal lane such as security backlog and threat modelling refer to work items exclusively performed by a single role.

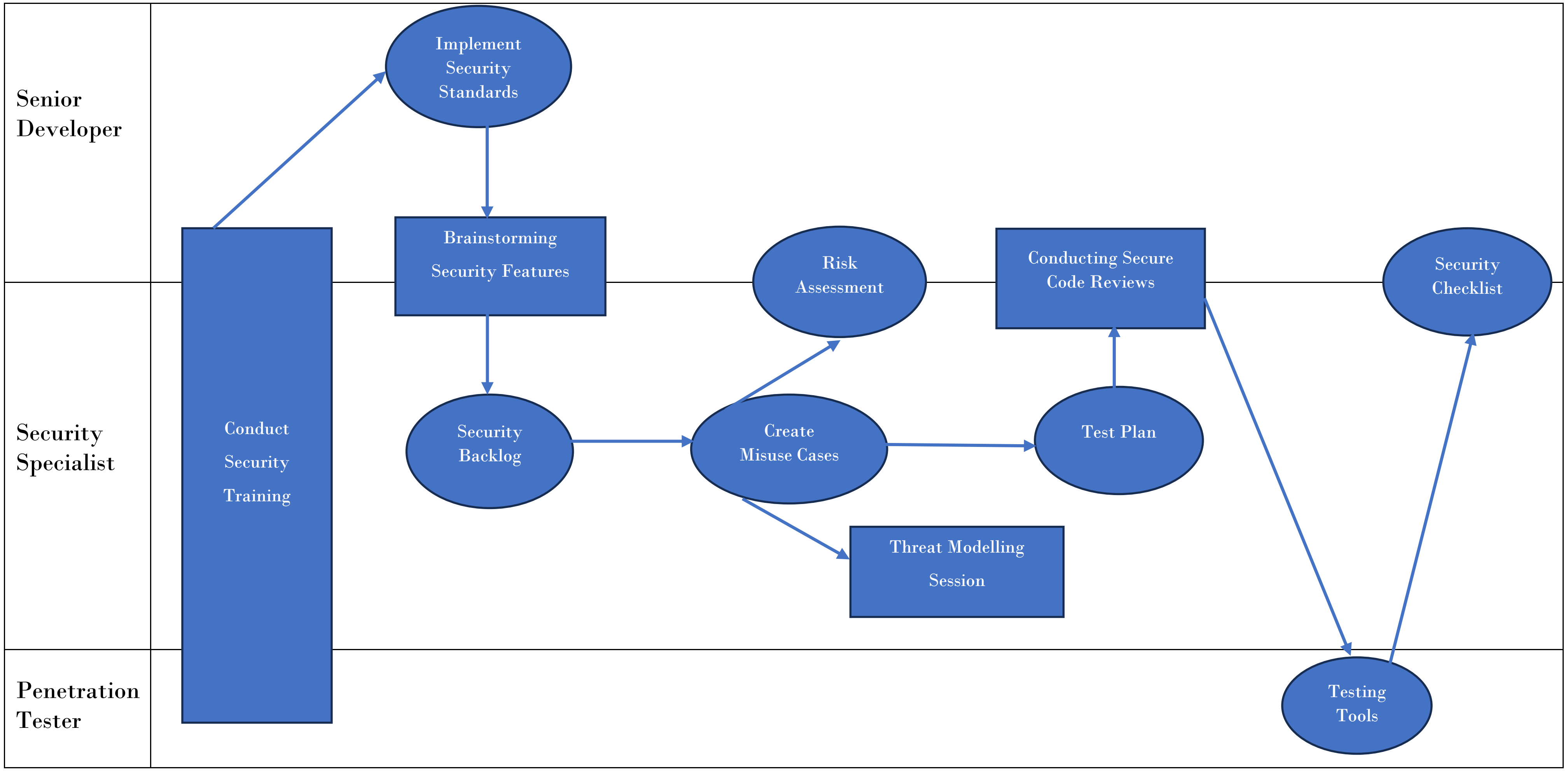


Figure 4.1: Taxonomy of Agile Security Practices (Ardo et al., 2021)

Security Practices: Oval Shapes – Artefacts. Square-rounded Shapes – Ceremonies. Horizontal Swimlanes Titles – Roles.

4.6 Summary

This chapter reported the findings of the exploratory case study conducted to investigate the current state of agile security practices implementation by organisations. The exploratory case study engaged practitioners through ten semi-structured interviews which were recorded and transcribed. The interview transcripts were analysed using a method informed by grounded theory. The study presented an extended version of the taxonomy of agile security practices which were derived from practitioner interviews (Ardo et al., 2021). The study identified more artefacts than roles and ceremonies. The security practices identified included ten artefacts, four ceremonies and three roles. The chapter also presented some of the challenges confronting agile practitioners during secure software development. The identified practices would be employed for creating a secure software development process model in chapter 6 of this thesis.

Chapter 5

Secure Agile Software Development Practices in Selected Nigerian Companies

5.1 Introduction

This chapter presents an in-depth analysis of the data collected across different research sites in Nigeria. Just as has been done for the development of the taxonomy of agile security practices, data was collected and analysed using a method informed by grounded theory as described in Chapter 3. This chapter provides an in-depth investigation into the implications of regulatory policy for building secure agile software in Nigeria which was an issue of critical importance as described by practitioners **(Objective 2)**. This chapter specifically discusses some of the factors militating against the adherence to the Nigeria Data Protection Regulation (NDPR) policy for secure agile software development.

Study findings revealed tension between practitioners' compliance and the Nigerian regulatory environment. This is followed by the development of a grounded theory of the security challenges confronting agile practitioners. The novel GT is termed policy adherence challenges (PAC) model which has already been published (Ardo et al., 2023). Thereafter, the implementation of security practices as described by practitioners in the agile software development process is examined.

5.2 Tension within a Changing Regulatory Landscape

This study discovered the existing tension between software development companies' adherence to some sections of the government's NDPR Act 2019 and the guidelines for Nigeria's content development in ICT. One of the senior software engineers from an educational software solutions company (ESSco1_SSE1) explained the impracticability of hosting their applications locally.

> "Honestly, *it would be very difficult to find a suitable company that can host our software applications here in Nigeria with all the challenges, and even if there are, it must be very expensive…*" (Senior Software Engineer, Company B).

While practitioners find that it is almost impossible to host their applications locally, the government claim it has made significant progress towards the adoption and utilization of ICTs. The government further claimed that data and information hosting should be inevitable in both the public and private sectors due to the adoption of ICT in Nigeria. The Former Minister of Communications and Digital Economy was reported in the Vanguard Newspaper of December 5, 2017, to have said, "*We condemn the current practices by both public and private organizations hosting data offshore, despite having highly reliable Tier III & IV Data centres, certified by various international organizations*".

According to section 14.1 of the Nigerian content development for ICT guidelines, the government has made it mandatory for all Information Technology companies to host data locally.

> "*It is mandatory for data and information management firms to host government data locally within the country and shall not for any reason host any government data outside the country without an express approval from NITDA.*"

While the government has claimed that existing data centres guarantee almost 100% availability with multiple levels of security, practitioners in this study do not agree with the assertion. One of the Chief Technology Officers (CTO) interviewed cited the infrastructure deficit in terms of electricity and internet connectivity among the impediments to adhering to government policies.

> "*We have [an] infrastructure deficit and so the system is going to be subject to those challenges. So, … you need constant power supply, physical security, you have to provide connectivity and so on and you know these are serious costs…*" (Chief Technology Officer, Company B).

Apart from the widely recognised infrastructure deficit in Nigeria, there are instances where clients prefer their applications to be hosted offshore. Clients working in certain sectors, such as healthcare, tend to trust offshore companies to manage their software applications. They identified issues of downtime which can be disastrous to their business. According ITSERVco2_SE1, "*Clients doing business in the safety critical sectors prefer to host offshore as they don't trust the local companies, we have in Nigeria …*" (Software Developer, Company B). While adhering to government policies is good, securing client trust is also important. Thus, if clients continue to mistrust local hosting companies, they will still prefer their data and software applications to be hosted offshore.

The practitioners described the challenges they face in their attempt to adhere to the NDPR policy. ESSco1_SSE1 explained:

> "*When they first released [the] NDPR and we were reviewing the document, my CTO said this policy is not realistic. Looking at the architecture of the applications we build, Nigeria lacks the indigenous hosting capabilities that we need. Power and internet interconnectivity remains big challenges to date.*" (Senior Software Engineer, Company B).

From the above quotation, which was echoed by others on a similar theme, the concept of "adherence" to the NDPR was coined as the major concern for agile practitioners. This is due to the challenges of implementing the policy. Categories representing challenges to adherence include unawareness, distrust, compromise, and culture. These categories represent the building blocks of the grounded theory developed in this thesis which explains the difficulties confronting agile practitioners. The categories and relationship between them are explained in this section.

5.2.1 Practitioners' Unawareness

It emerged that many practitioners are unaware of the existence and capabilities of the few tier IV software hosting companies in Nigeria. These companies are in the major cities of Lagos, Abuja, Port-Harcourt, and Kano. In most situations, because of infrastructure challenges bedevilling the nation, practitioners automatically choose foreign hosts without exploring the capabilities of indigenous companies. This inclination towards foreign companies suggests unawareness amongst practitioners of what exists in the country. While very few of them may be aware of local hosts, practitioners still prefer offshore companies which is partly due to concerns regarding distrust. Hence, this creates tension between the government and practitioners which may also be attributed to the limited number of hosting companies.

Apart from a lack of awareness of the existence of indigenous hosts, some of the interviewed practitioners were not particularly conversant with the provisions of the NDPR Act. ITSERVco2_SE1 stated: *"Yes, I have heard of it but do not know the details of the regulation. Although I know the part about software and data hosting but sincerely, we have always hosted offshore…"* (Software Developer, Company L).

5.2.2 Availability of Different Choices

Policy adherence has a positive relationship with the choices available to practitioners. The same was found with client trust. When different choices were available to practitioners, they chose the one they trusted the most. In this study, the agile practitioners chose foreign hosting companies because they trusted their capabilities. FSSco1_SDE1 explained:

> "*Clients will not be willing to listen to excuses when they get hacked or can't transact due to downtime. Most of them don't care how you do it, so we make decisions based on what we trust and are comfortable with…*" (Senior DevOps Engineer, Company K).

5.2.3 Distrust of Indigenous Software Companies

This study discovered that both the agile practitioners and their clients do not trust the capabilities of indigenous software hosting companies which is referred to as "Distrust". HSCco1_FED1 explained: "*Medical software are safety critical and so clients prefer we host it offshore which we are also more comfortable with due to downtime issues of local hosts.*" (Front-End Developer, Company O).

5.2.4 Compromising Software Security

The category emerged because of practitioners compromising software security due to cost. Most companies - especially SMEs - compromise software security and data privacy by hosting applications offshore because it is cheaper and saves on administrative fees. This poses a choice, namely, to adhere to the NDPR regulatory policy or prioritise cost considerations. Adherence to the NDPR, which is a requirement for software development stipulated by the Nigerian government is then compromised by practitioners due to budget constraints. MFGco1_ITA explained: "*The truth is the few Nigerian hosting companies' charges are too* *high since their service charges are based on dollar-to-naira exchange which is very unstable as you know…*" (Manager, IT Security & Operational Risk, Company H). Other concepts impacting the "compromise" category include company policy, available skills and knowledge, security maturity and cost.

5.2.5 Organizational Culture

This research revealed a desire by agile practitioners to adhere to the NDPR by building an organizational culture. In certain organizations, the concept of culture is propagated through company policy documents. MFGco1_ITA stated: "*Employees are guided by our security policy document during software development…*" (Manager, IT Security & Operational Risk, Company H). It emerged further that non-adherence to the NDPR can be attributable to the concepts of "unawareness" of the policy and "compromise" meaning compromising essential skills over the cost of security training.

5.2.6 Sources of Cost

Cost is a big determinant that shapes organizational culture when developing secure software. Sources of cost include the security training for team members, the purchase of software and hardware tools and the payment of fees to regulatory agencies. While adherence to regulatory policies is important and necessary, budget constraints represent a big barrier. ESSco1_CTO1 explained: "*We pay for services and purchasing hardware and software in dollars which can be very expensive you know*" (Chief Technology Officer, Company B).

5.2.7 Policy Adherence Challenges

This study's social process sought to understand practitioner behaviour towards a regulatory policy. The social process is termed "policy adherence challenges", which explains the reasons agile practitioners are refusing to comply with the NDPR during the software development process. This study introduced the policy adherence challenges (PAC) model. The constructs and relationships in the emergent theory, PAC, were derived using a method informed by grounded theory. Figure 5.1 shows the different stages of the model. The model starts with the current state of non-adherence at stage 1. Investigating the reasons why practitioners are not compliant moves the policy adherence process to stage 2. Through practitioner engagement, challenges to non-adherence were identified. The adherence process either moves to stage 3 (tension phase) or the pursuit state (NDPR adherence achieved). When the challenges identified are not resolved, then the process moves to stage 3 where tension is observed, and the circle backtracks to stage 1 (state of non-adherence) of the NDPR. However, when there is positive collaboration between the government and agile practitioners, practitioner resistance can be overcome, and adherence achieved to terminate the process.

5.2.8 Overcoming Adherence Challenges

While seeking to promote adherence to NDRP, the problem of practitioner resistance (seen in stage 1), means that some of the challenges observed include unawareness, distrust, compromise, and culture. The recommended solution to the identified challenges is for the government to make practitioners aware of and encourage trust in the capabilities of local hosting companies. This will also be dependent on government investment to close the infrastructure gap and support SMEs and other start-ups to reduce the risk of making compromises. An increase in government sensitization of practitioners on the significance of NDPR compliance will steer the non-adherence process in a positive direction. Thus, the more practitioners are aware and trust indigenous hosts, the more likely adherence will be achieved.

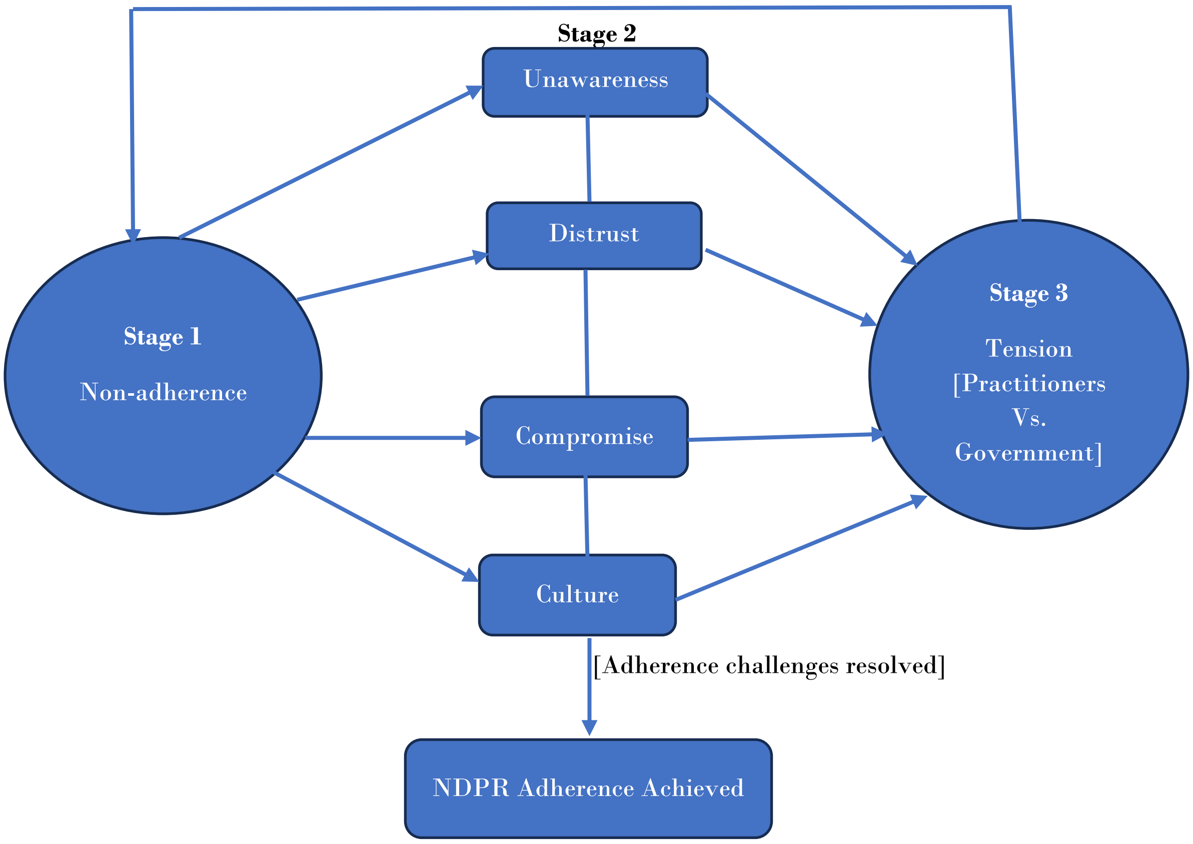


Figure 5.1: Model of Policy Adherence Challenges (Ardo et al., 2023)

5.3 Identification of Security Practices for Agile Implementation

This section describes the identified practices used by organisations in this study for building secure agile software. Essentially, this section classifies the practices as has been done for the taxonomy in chapter 4. The identified security practices have been categorised into roles, ceremonies, artefacts.

5.3.1 Secure Agile Artefacts

Five artefacts have been identified in this study that are used by different organisations across their software development lifecycle: baseline security standards, industry regulatory standards, security audit plan, security regression tests, and security patch management. These artefacts help agile team develop secure software by mitigating cybersecurity attacks across the development pipeline.

Baseline security standards

In this study, it was discovered that company documentations serve as the starting point of building organisational cybersecurity culture. They are shared as company security policy or code of conduct. MFGco1_ITA described their security policy by saying, "*We got our software development documentation which guides developers on how to secure their code and implement security in the design of their applications…*" (Manager, IT Security & Operational Risk, Company H).

In contrast, some organizations have policies or guides on software development in general with some sections devoted to building secure systems. As part of the company policy documentation, security best practices are defined to guide organizations secure their code and implement security in the design. As described by ESSco1_CTO1, some security policies are only introduced after experiencing breaches,

> "*We don't have security specific documentation but ensure our policy include issues of building secure software. Some policies are introduced after we had a nasty experience and then realised, we needed this and that...*" (Chief Technology Officer, Company B).

Industry regulatory standards

Security compliance in agile software development refers to the adherence level of companies to security norms. This study found a clear dichotomy between developers and security practitioners' compliance to software security standards. There are organizations especially those developing secure software in the FinTech industry where understanding security standards by the security team is a priority. They are mandated as part of the company policy to ensure their security team are constantly aware of updates to ISO standards. In company K, for example, security team members are required to be up to date with regards to changes in software security standards.

> "*We keep tap with periodic reviews to ISO standards like ISO 27001, ISO 9001 and PCI DSS or others and it's always easy for security team members to understand them since they are mostly little updates here and there to what we already know…*" (Senior DevOps Engineer, Company K).

Apart from the FinTech industry, some of the interviewed agile practitioners working for software companies involved in developing secure applications for the healthcare sector have displayed a very good understanding of security standards.

> "*There is what we call HIPAA compliance security standards in the healthcare industry which me and others in my team that are involved in the security site of things have a good understanding of and ensure compliance always*" (Front-End Developer, Company O).

However, some of the interviewed software developers in this study mentioned the problems of understanding and ensuring compliance to software security standards. "*I leave everything about compliance to our security specialist as it can be complex or ... even ambiguous a times you know…*" (Software Developer, Company C). Similarly, in Company N, for example ITSERVco3_QAA said, "*I agree our developers invest less effort to understand compliance & security standards… maybe we would look into that…*" (Quality Assurance Analyst, Company N).

These findings show that security practitioners and software developers exhibit different levels of understanding of compliance to security standards during the agile software development process.

Security audit plan

For the continuous success of a deployed software product, an audit plan needs to be in place to identify any potential points of weaknesses. Industry standards and best practices are common ways of identifying areas of risks to companies. In company H, for example:

> "*We rely on industry best practice, CIS benchmark which tells you about the benchmark for software development and secure coding and the rest to design our audit plan which is based on highest risk areas…*" (Manager, IT Security & Operational Risk, Company H)."

The plan describes the schedule, types of audits and personnel to be deployed. For audits schedule, "*Before starting a new project, there is a risk assessment and out of that risk assessment ... audit timetable is presented based on highest risk areas…*" (Manager, IT Security & Operational Risk, Company H). In terms of audit types, *"...security audits can be* *code analysis, penetration testing and then it can be a general security reviews especially if you are working with cyber frameworks…"* (Senior Software Engineer, Company B).

Security regression tests

These tests are performed to ensure changes made to a particular part of the system does not create security threats to unchanged parts. As explained by ESSco1_CTO1:

> "*…we have different tests that are part of quality assurance to ensure it does not break anything, to ensure there is what we call continuous integration. It does not mean if you build a new feature, it should break what is existing…*" (Chief Technology Officer, Company B).

Although conducting security regression tests are important in the development of any software system, techniques such as test cases selection can be time consuming. "*Selecting test cases for the different parts where changes have been implemented takes hours or even a whole day...*" (Security Consultant, Company J). Due to the time-consuming nature of conducting security regression tests, Company B had adopted the test suite minimization technique. "*…our strategy for reducing time is by eliminating redundant test cases…*" (Chief Technology Officer, Company B).

Security patch management

Security patching is the process of identifying vulnerabilities and improving software functionalities to proactively avoid against attacks. It is an important theme in company policy documentation as explained by an interviewed practitioner, "*part of our company security policy document are guidelines on security patch management to handle the ever-changing security vulnerabilities during software development.*" (Chief Technology Officer, Company B). The importance of security patching and adoption of regulations during software development cannot be overemphasized especially in the healthcare industry. In building health software, "*security patching improves software quality by helping us discover vulnerabilities & fix them quickly… Adhering to HIPAA compliance & IEC 82304-1 regulation standards also helps*" (Front-End Developer, Company O). Furthermore, fixing issues through security patching reduces software maintenance time, "*modifying our software through the security assembly of the part having issues saves us a lot of time*" (Front-End Developer, Company O).

5.3.2 Secure Agile Ceremonies

One of the four key values of agile methodology is *customer collaboration over negotiation*. Thus, the important of engaging clients can never be overemphasized in agile software development.

In the phase 2 study conducted with Nigerian practitioners, five secure agile ceremonies were identified. They include conducting security trainings, demoing security features, penetration testing, secure CI/CD pipeline activities, and secure agile retrospective.

Security trainings

Practitioners in this study discussed the concept of security training in their organisations. According to ESSco1_CTO1, "*We don't actually do security specific training; however, occasionally we invite experts to talk about basic security like handling passwords, data encryption and other stuff to raise security awareness…*" (Chief Technology Officer, Company B). The lack of security trainings for practitioners have been corroborated by ITSERVco2_SE1, "*We have not really been involved in security training...*" (Software Developer, Company L). However, depending on a project's security requirements, he explained that some senior engineers maybe trained. The CTO said, "*If we are building a mission critical application, certain senior engineers maybe trained...*" (Chief Technology Officer, Company B). In contrast, other companies offer lots of security training; FSSco1_SDE1 said, "*We implement security through lots of trainings, we call it train and on-train or learn and on-learn. It's just a way to increase awareness which happens every quarter…*" (Senior DevOps Engineer, Company K).

Apart from formal training, practitioners in this study explained several useful strategies they use internally to train other colleagues such as brown bag sessions and mentorship. For example, ESSco1_SSE1, "*We do engage in brown bag session to share knowledge with other colleagues about security or any development although it's a rare practice here…*" (Senior Software Engineer, Company B).

Security practices that sometimes get implemented through workshop sessions, involving different roles such as: self-organizing team, security specialist or security team as described by ESSco1_PROD-MGR1: "*…security requirements gathering workshop stakeholders include engineers, security specialist or at-times security team members. We share ideas of* *envisaged unauthorized access and attacks possible…*" (Product Manager, Company B). ESSco1_PROD-MGR1 also explained the concept of mentorship by saying, "*In company X, Senior engineers do mentor junior and mid-level engineers on strategies of building secure applications based on the experience garnered over the years…*" (Product Manager, Company B). The security mindset remains an issue because the priority for practitioners is always to deliver working software. MFGco1_ITA described the developer mindset by saying, "*The main concern for most practitioners is getting working software. They don't think much about users' protection because they mostly don't have the right security training to understand the implications…*" (Manager, IT Security & Operational Risk, Company H).

Demoing security features

Demoing sessions are important to ensure requested software products are delivered with all necessary features. According to ESSco1_PROJ-MGR:

> "*In the process of our development, we carry our clients along so whatever security feature we add we show them to get their feedback since we want to build what they are happy to use…*" (Project Manager, Company B).

These sessions help agile teams get feedback of their clients. ESSco1_BED1 explained:

> "*It becomes a big problem for you whenever a customer gives negative feedback on a feature which you were assigned to develop as the XXX can sometimes shout at you during meetings not minding you have put up your best to come up with the feature and sometimes lack the necessary skills…*" (Back-End Developer, Company B).

ESSco1_BED1 further explained how negative customer feedback have pushed employees to leave the company: "*Just last week a DevOps engineers resigned after a very bad encounter in one of our session due to some features the client complained about.*" However, in company H, MFGco1_ITA explained that client's feedback has helped them improve their processes:

> "*We look at customer feedback in two ways: first we use it to draw an action plan to improve the product and second we identify any gaps on our side maybe competence that needs to be built into the team*" (Manager, IT Security & Operational Risk, Company H).

Therefore, this study has revealed that companies react in different ways to the feedback they receive from customer demos of security features.

Penetration testing

Penetration tests are setup by a third-party to mimic an actual attack. The advantage of penetration test over security scans is that scanners might not identify all threats. "*I go beyond the mere code scans and conduct pen test because scanners may not give you the result of all vulnerabilities*" (Security Technical Program Manager, Company F). In Company B, experienced engineers are involved in security specific tasks and responsibilities as explained in the following quote:

> "*Senior Engineers internally hack and try to see if they can actually break the system… if it's possible then, more algorithms or more features might essentially need to be written to ensure it is more effectively secured*" (Senior Software Engineer, Company B).

Secure CI/CD pipeline activities

The main responsibility of DevOps is to act as an intermediary between development and operations teams. In secure software development, they create the pipelines and continuously review them to ensure continuous integration and continuous delivery. The DevOps team are tasked with different activities of the software development processes. For example, in company K, the DevOps team collaborate with senior developers to automate and monitor pipeline processes:

> "*…The DevOps team and the senior developers usually work to automate CI/CD pipeline activities… through our usual forum we got some automated bots that notify developers about new releases, and they pick it up from there, push it to a staging environment and test that everything is ok before it is released…*" (Senior DevOps Engineer, Company K).

In ESSco1, the DevOps engineer works with the CTO and other senior engineers to use automated tools such as Kubernetes for creating the pipelines: "*We work as a team with the CTO and other senior engineers to create the pipeline using Kubernetes, peer-review it and deploy designed solutions*." (DevOps Engineer, Company B).

Security retrospective

In agile software development, security retrospectives are meetings held after each iteration to reflect on what happened, and brainstorm ways of improvement. In ESSco1, security retrospective is held to reflect on security mechanisms implemented. CTO_ESSco1 said: "*…we do hold meetings to reflect on security mechanisms implemented and discuss ways of improvement to avoid attacks* …" (Chief Technology Officer, Company B). It is an important activity in the development process as it enables teams identify lessons learnt as well as plan for the next iteration. The ninth principle of the Agile manifesto emphasizes the need for continuous improvement at regular intervals. ITSERVco1_STP-MGR described security retrospective practice in his organisation which takes place usually every two weeks. "*All stakeholders meet like fortnightly to discuss where we are failing... in terms of like misconfigurations, vulnerability, and patch management. So, ... there is a lot of visibility into security issues*…" (Security Technical Program Manager, Company F)

5.3.3 Secure Agile Roles

Agile teams constitute a cross-functional group of people collaboratively working together to deliver secure software products. Thus, team collaboration is an essential principle of the agile values which align with the major aim of secure software development and SecDevOps. In this study, the three roles discovered includes security specialist, penetration tester and DevOps team.

Security specialist / Senior Developers

In this study, we discovered a gap in the collaborative nature between agile teams and the security specialist, for example DSco1_SETL1 explained:

> "*Currently at the company where I am working, it is mainly the role of the two security specialists to handle the security aspect of things although as the software team lead, I do get called sometimes but the rest of my team members are not involved*..." (Security Team Lead, Company M).

In companies without security teams, senior engineers collaborate with contracted third parties; however, this is dependent on the project's sensitivity and available budget as described by ESSco1_CTO1:

> "*We don't have like a security team, but the senior engineers base their experience on understanding the security needs of the system and if there is a need we get some security experts, and they handle it together depending on the nature of the application and our budget*" (Chief Technology Officer, Company B).

Similarly, in SMEs with smaller security teams, experts are contracted to work with those responsible for handling security issues, for instance, FSSco1_FLM explained:

> "*At XXX since the size of our security team is small, we have partners that are experts in cyber security, cyber investigation and osint intelligence whom our security team works with through the whole software development process"* (Frontline Manager, Company K).

DevOps Team

In other companies, the development and DevOps teams work collaboratively while an information security team function independently, as explained by FSSco1_SDE1, "*The Developers and DevOps team actually work together on all issues while the information security team test for security alone*" (Senior DevOps Engineer, Company K). Consequently, practitioners highlighted existing gaps between the agile teams and security specialists.

Penetration testers

Company H has a separate Information Security Team composed of penetration testers who perform tests to identify security weaknesses. "*In my organization, we do have a security team with pen testers separate from the development team, but they all work together*" (Manager, IT Security & Operational Risk, Company H).

5.4 Secure Agile Software in Regulated Environments

Regulated environments are businesses developing critical software that must comply to certain regulations, and standards. They also consider software quality, security, and traceability as important concepts (Fitzgerald et al., 2013). In this study, practitioners developing secure agile software in the healthcare sector and fintech have discussed some regulations and standards they have to comply with to avoid sanctions from their regulators.

5.4.1 Regulations in FinTech Industry

The Nigerian financial industry is a highly regulated environment and so software development in the sector needs to confirm to many standards. FSSco1_SDE1 explained: *"We have so many regulators like ISO 32000 and PCID which we have to adhere to"* (Senior DevOps Engineer, Company K). The financial industry especially Nigerian banks have grown to a level that security of software products; be it Mobile Banking Apps or Internet Banking websites are a major concern. ITSERVco1_STP-MGR gave an example: "*I used to work as a security consultant for Banks in Nigeria and I can say almost 80% of them have grown such that they are proactive in terms of adhering to security standards in software development …*" (Security Technical Program Manager, Company F).

5.4.2 Regulations in Healthcare Sector

Another highly regulated industry within the Nigerian economy is the healthcare sector. Just like how sensitive money is to the financial sector, patient's records are highly sensitive and confidential. Thus, even the practitioners in-charge of software projects do not always have access to live data due to laws governing the industry. HCSco1_FED1 explained:

> *"Security is key especially when you work in the healthcare as there are Laws that says that not everyone working in the development side should have access to patient data because of the sensitivity involved in the healthcare industry*." (Front-End Developer, Company O)

5.5 Secure Agile Software Development Challenges

This section explores the challenges of developing secure agile software from practitioners' perspectives.

5.5.1 Secure Software knowledge Gap

Developing secure software largely depends on the knowledge and skills of practitioners. In this study, ITSERVco1_STP-MGR highlighted the paucity of Nigerian Universities offering cybersecurity courses, "*We don't have cybersecurity as a course in most Universities… I don't think we got any school offering it at M Sc level…*" (Security Technical Program Manager, Company F). ITSERVco1_STP-MGR considers education a key component that needs to be acquired before addressing industry-based issues. FSSco1_SDE1 explained that knowledge gap also manifests in many companies when they acquire new security tools; "*Apart from cost constraints, [a] knowledge gap comes when we acquire some security tools and don't really have anybody that can deploy it ...*" (Senior DevOps Engineer, Company K). While the issues around cost are a major impediment to software development projects in Nigeria, ITSERVco2_SE1 considers skillset a greater challenge to building secure applications. "*Cloud services could actually be expensive and other resources problem are there, however, resources are not the major thing, it's the problem of skills…*" (Software Developer, Company L).

Thus, the practitioners interviewed assert there is a knowledge gap in secure software development.

5.5.2 Talent Emigration

Neglecting security in the development of software applications has also created emigration challenges among the very few skilled practitioners available in the context. Since skilled practitioners are sought globally due to the increasing cases of cyberattacks, the number of professionals leaving Nigeria continues to increase. As explained by ITSERVco1_STP-MGR, "*Security is one space where there are lots of job opportunities and the skillset is rare. So, there is a lot of migration of talents from Nigeria to different parts of the world…*" (Security Technical Program Manager, Company F).

5.5.3 Tight Project Schedule

Pressure on teams to deliver software products within very tight timeframe is another big challenge. There are many instances when managers do not provide realistic deadlines in a bit to secure contracts. DSco1_SETL1 explained:

> *"Project managers while following-up contracts don't communicate achievable datelines to the clients. A project that should take 4 months, they will tell the customer we will do it in 1 month, 6 days"* (Security Team Lead, Company M).

Due to managers giving unrealistic schedules, the challenge is passed on to developers who in most cases will not have the time to ensure security is well integrated. DSco1_SETL1 further stated: *"The pressure that project managers put on developers don't even give them a second to think about security. It also means having short sprints as all they want is results."* (Software Security Team Lead, Company M).

There are situations when the pressure comes from clients rather than project managers. Security managers may decide to hold an impromptu to review security concerns. They might not be creating a list of security specific items as normally done during the development process due to time constraint. According to HCSco1_FED1 explained: "*We sometimes sit together with the IT-Security Manager and members of his team to allow everyone ask questions and we brainstorm about security concerns*" (Front-End Developer, Company O).

Consequently, neglecting security due to tight project deadlines impacts the software development process.

5.6 Summary

This chapter presented an in-depth discussion of the impediments to regulatory policy adherence for building secure software using agile methods. A published grounded theory of the security challenges confronting agile practitioners termed policy adherence challenges (PAC) model was developed (Ardo et al., 2023). The construct of the emergent theory includes unawareness, choice, distrust, compromise organisational culture and costs. The four challenges to secure agile software development identified were (a) a lack of collaboration between security and agile teams; (b) the tendency to use foreign software hosting companies; (c) a poor cybersecurity culture; and (d) the high cost of building secure agile software. The chapter also explored the practices adopted by practitioners for secure software development.

The identified practices were categorised into roles, ceremonies, and artefacts. The practices would be employed in the creation of a model for secure agile software development process in the next chapter.

Chapter 6

Cross Case Analysis Findings & Secure Agile Software Development Process Model Creation

6.1 Introduction

This chapter further examines agile implementation using the findings presented in the two preceding chapters (4 and 5). The chapter demonstrates the creation of the proposed model. Findings from the various research sites were used to show the chain of evidence from memos to model elements. Table 6.1 links appropriate sections of chapters 4 and 5 to show where the elements on figure 6.1 originate.

Based on the findings from the research sites in phases 1 and 2, along with the cross-case findings, a model for secure agile software development process is created. The model creation followed an iterative process using swim lane diagram notation. The constant comparison technique was used to refine the model following the accumulation of additional information from memos. The model development began with the baseline state which represents the collation of security practices adopted across different research sites. The security practices represent the elements of the secure software development process model, which are presented and discussed in this chapter. Finally, the secure process model to be evaluated in the chosen research site is presented.

6.2 Cross-Case Analysis Findings

This section discusses how the data collected in the UK and Nigeria synchronize and differ from each other. Since the study conducted in the UK was preliminary, it did not consider all the influences identified in the more comprehensive research conducted in Nigeria. Table 6.1

illustrates the security practices identified in the UK, comparing them with those observed at research sites in Nigeria.

Approaches to Conducting Security Trainings

A notable distinction was identified in the approach to security training between the surveyed UK companies and the Nigerian sites. In the UK, there is a prevalent practice of conducting both formal and informal training sessions to raise security awareness and foster a culture of cybersecurity. Regular periodic and yearly security trainings are also widely advocated and utilized. Furthermore, organizations in the UK often distribute guidelines on secure software development among their employees.
On the other hand, findings from Nigerian sites indicate a correlation between organizational context and cybersecurity culture arising from a lack of training. Practitioners reported lack of participation in security trainings. For instance, at ESSco1, trainings are prioritized only when developing mission-critical applications, leading to the training of select senior engineers. Consequently, this approach has widened the security knowledge gap within the company.

Compliance to Regulatory Policies and Security Standards

Compliance with regulatory policies and security standards varies among the companies studied. UK companies demonstrate better adherence to regulatory policies like GDPR, which is well-established and comprehended by agile software practitioners. However, practitioners' primary concern lies in adhering to Article 17 of the GDPR, which pertains to individuals' rights to have some of their data erased by organizations. On the contrary, companies in the global south, particularly Nigerian firms examined in this study, underscored the tension between the government and agile practitioners concerning compliance with regulations. Practitioners disclosed encountering numerous challenges in finding a suitable local host for their software applications, and even if one is found, the costs are prohibitively high. Consequently, practitioners highlighted the difficulty of complying with the NDPR (Nigeria Data Protection Regulation).

6.3 Mapping of Security Practices

This section presents Table 6.1 showing the security practices identified in various research sites. The practices have been categorised into roles, ceremonies, and artefacts and mapped onto appropriate chapters and sections of the thesis.

Table 6.1: Security Practices Mapping

Chapters	**Agile Concepts**		
	Artefacts	**Ceremonies**	**Roles**
4	-Vulnerability assessment tools -Code verification tools -API testing tools -Git-Hub test platform $\left.\right\}$ Testing tools Security backlog Creating misuse cases Risk assessment & mitigation Security standards Security test plan Security checklist **(Section 4.2.1)**	-Threat modelling session -Conducting secure code reviews -Brainstorming sessions -Conducting security trainings **(Section 4.2.2)**	-Security specialist -Penetration tester -Senior Developers **(Section 4.2.3)**
5	-Baseline security standards -Industry regulatory standards -Security audit plan -Security regression tests -Security patch management **(Section 5.3.1)**	-Demoing security features -Penetration testing -Security trainings -Secure CI/CD pipeline review -Security retrospectives **(Section 5.3.2)**	-Security specialist -Senior Developers -DevOps Team -Penetration testers **(Section 5.3.3)**

Bringing together agile practices from various research sites can be beneficial in enhancing the software development process (Rahy & Bass, 2021). Existing literature has also presented an empirical taxonomy of DevOps implementation in practice (Macarthy & Bass, 2020). The taxonomy offers an innovative mapping of the identified approaches to on-premises and cloud-based deployments, as well as the facilitators of DevOps practices within these distinct approaches. Although Table 6.1 does not aim to present a complete list of agile security practices in the industry, the researcher emphasized these practices because the analysis of collected data highlighted their importance. For example, practices like agile sprints and retrospectives were mentioned by different practitioners without any evident connection to security and compliance issues. Although combining security practices is beneficial, it does pose some challenges. The process of identifying security practices across different research sites can be highly time-consuming and labour-intensive. The researcher had to manage multiple interview transcripts to identify the practices. At times, gathering data from various research sites may not provide a clear picture of current practices. The data might be inconsistent or contradictory. However, in this research, after meticulous analysis of the five hundred and fifteen pages of transcript data, the researcher did not encounter any conflicts among the identified practices.

6.4 Secure Agile Software Development Process Model Creation

Following an in-depth investigation of agile implementation in organisations, security practices adopted across different research sites were identified as detailed in Table 6.1. The security practices served as inputs for the model creation process. A critical examination of agile implementation in participating research sites shows lack of security integration in software development processes. While this study does not claim on providing an exhaustive list of techniques adopted in practice, those mentioned are based on the analysis of data collected. This means the elements of the proposed model represents the current practices adopted for secure agile software development as described by practitioners. Figure 6.1 presents the baseline state of the proposed model using a swim lane diagram notation, portraying identified practices, and indicating information flow. The diagram illustrates how the agile and security team roles collaborate or work independently to secure the development process.

Some of the security practices described by interviewees in this research are similar to those of existing secure software development models. They are all aimed at implementing practices to secure the software development process. However, figure 6.1 have categorised identified practices into agile use in organisations – roles, ceremonies, & artefacts. This categorisation provide foundation for the design of a model for secure agile software development process. Just as software development methods are not applied in textbook manner in organisations, the practices described in the proposed model may not be implemented in the same way in companies outside this research.

6.4.1 Baseline State

Figure 6.1 shows the baseline state of the process model. Each of the horizontal lanes represent an agile role, while the oval shapes and square rounded boxes positioned across the diagram represent artefacts and ceremonies respectively. The swim lane notation shows the collection of activities performed to achieve desired outcome (Bera, 2012). Thus, the sequencing of activities using arrows is helpful for tracing an entire business process from a start to finish.

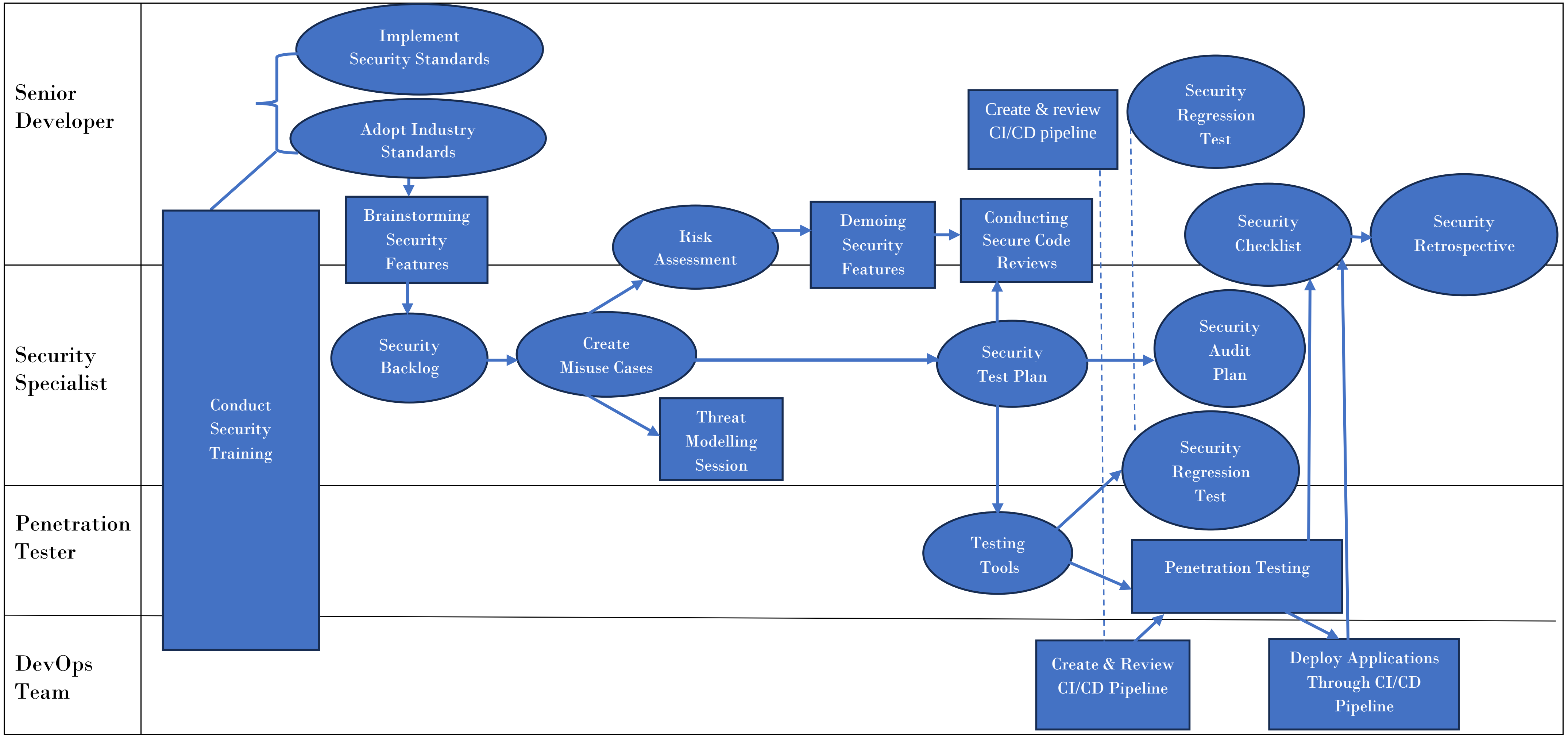


Figure 6.1: Model Baseline Process Flow

Security Practices: Oval Shapes – Artefacts. Square-rounded Shapes – Ceremonies. Horizontal Swimlanes Titles – Roles.

Figure 6.1 presents a swim lane diagram showing identified security practices and the activities performed by different roles as well as those collaboratively executed. The diagram uses four swim lanes to show the roles involved in the software development process. The diagram is used as a representation for the various research sites. The software development process starts with conducting security trainings for the different roles and ends with security retrospective collaboratively performed by the development and security roles. The borderline practices on the swim lane diagram such as conducting security training and brainstorming session indicate activities collaboratively undertaken by two roles (actors). The diagram shows the high influence of the security role as evidence by the number of activities assigned. The model shows how the other roles (penetration tester & DevOps) collaborate to handle security throughout the development process. All the practices were arranged in sequence to show the security activities undertaken throughout the SDLC. The model shows a lack of collaborative ceremonies to disseminate awareness and compliance to security standards and regulatory policies.

6.5 Summary

This chapter describes the similarities and the differences between the findings from the UK and Nigeria research sites. It further presents the steps taken to create a novel practise-based software development process model based on the analysis of interview transcripts. The model creation process began with a baseline state presented in figure 6.1 which was based on the findings from various research sites. The practised-based model is intended to guide organisations adopt secure practices across the SDLC to reduce incidences of cyberattacks. The model elements were categorised based on agile use in organisations – roles, ceremonies, and artefacts.

Chapter 7

Secure Agile Software Development Process Model Evaluation

7.1 Introduction

This chapter describes the steps taken to evaluate the developed secure agile software process model. A preliminary evaluation was undertaken by presenting the model to a team of five agile cybersecurity experts based working in UK companies during a focus group workshop, as summarized in (Ardo et al., 2022). Quotes from the workshop are used to relay practitioners' perceptions of the model as well as ways of improving it. The evaluation led to introducing an agile ceremony (compliance sprint) created in response to an observed lack of collaborative ceremonies, to disseminate awareness on security standards and policy legislations. In this thesis, "compliance sprint" is introduced as a dedicated session to enhance the understanding of security standards and regulatory policies. This aims to ensure that software development aligns with industry regulations. It is described as a collaborative agile ceremony for conducting a compliance check, leveraging on the diverse knowledge of team members on security standards. Dedicating entire sprints mainly focusing on compliance could enhance organisational cybersecurity culture, consequently reducing the occurrence of security software breaches. It is similar to protection poker but focuses on regulatory policy adherence. In a compliance sprint, the team actively disseminates knowledge regarding security standards and regulatory policies. It is crucial to ensure a thorough understanding and adherence to regulatory standards before software products are released and deployed in the customer environment. The growing frequency and severity of software vulnerabilities create a knowledge gap in security for agile teams. Thus, the significance of practitioners comprehending the various standards and policies applicable to the specific type of software they are developing cannot be overstated, despite the multitude of regulations and policies that software must adhere to.

After the preliminary validation of the refined research model, it was implemented in an organisation involved in secure software development for the healthcare industry. To begin the evaluation process in the chosen company, a workshop session on secure agile process was conducted for team members to raise security awareness and interest in the research findings. The workshop elicited reactions about the security process involved in the company. The evaluation involved comparing the secure agile practices contained in the model with those adopted in the chosen company. Based on the gap analysis findings, a call to action was initiated. The proposed action involved the implementation of a ceremony (security retrospective) from the developed process model in 3 – iterations of a project undertaken by the participating company. The chapter concludes with proposing an agile role (security champion) in the participating company with the aim of reducing the work burden on the Chief Technology Officer (CTO). The evaluation process was not only conducted to assess the proposed model but seeking to improve the security of the software development processes at the participating company. This chapter represents phase 4 of the research **(RQ4)**.

7.2 Secure Process Model Validation – Focus Group Workshop

As the secure practice-based process model developed evolves, a focus group session was conducted with a team of five agile cybersecurity professional working across diverse UK software companies. Following a detailed explanation of how the model was created, the practitioners were asked to reflect on some specific themes. These include their understandability of the model, relevance of the presented practices to their software development activities and if the model contains any redundant practices. The participants were also asked of any security practices they would have included when creating such kind of model. The focus group participants are detailed in Table 7.1.

Examining the model elements presented in figure 6.1, practitioner’s feedbacks were grouped into two themes: comments supporting the model and suggestions on improving it. First, results from the focus group session have shown that many of the practices identified are used in secure agile software development. According to ITSERVco1_VP-SSA&E, “*Yeah, … I agree with XXX here, you have got most of the practices that we are using here to make our software secure…*” (VP, Security Strategy, Architecture & Engineering). Also, other members engaged in the session generally supported the developed model. According to ITSERVco4_ITS, "*Yeah, I agree must of … the vast majority of the boxes are there … maybe you could look at improving collaboration between practitioners through your model, more meetings, more collaboration is always good …*" (Manager, IT Security). Second, it was suggested by one of the participants that different forms of penetration tests should be included onto the model. "*A split between the manual penetration test by experts and the automated ones could be something to put in there. Other than that, it seems you got everything that I could think of*" (Security Solution Architect, Company P). ITSERVco1_VP-SSA&E suggested adopting a single view for creating the model. "*…we all agreed that you got the majority of items in that, just that a view needs to change; whether you want to take a product development view or a security process view*" (VP, Security Strategy, Architecture & Engineering). Third, CYBERFOUDco1_ADL1 further suggested increasing awareness about security standards and compliance to regulatory policies which remains a challenge in practice.

> "*I think the issue of compliance to security standards is a big issue. Your models need to have more sessions where these issues are discussed…*" (Cybersecurity Analyst, Company A).

Introducing more collaborative ceremonies onto the model was supported by ITSERVco4_ITS another interviewed practitioner when he said: "*You can introduce a ceremony where issues of compliance to policies and security standards are discussed. This is especially important for developers to know as security people already understand these things*" (Manager, IT Security). Finally, there were suggestions on removing unnecessary practices and including missing secured techniques to the model. "*You shouldn't have a separate box for secure API meeting because that is part of your security requirement. It is the same as saying what is your data security requirement?*" (VP, Security Strategy, Architecture & Engineering). What was highlighted as missing in the model as highlighted by CYBERFOUDco1_SE was logins to figure out how a system was attacked and how it can be fixed. "*The only thing that I can say is potentially missing from the diagram is log-ins*" (Security Engineer, Company A). It was argued that most companies do not have that and even the few that have it defined in their baseline standards do not fully check to ensure compliance.

Figure 7.1 represents the secure process model to be implemented and evaluated in the chosen Nigerian research site. The processes where change is planned are indicated with call-outs with the agile concept to be implemented. To overcome the observed lack of collaborative ceremonies, compliance sprint will be introduced. Also, penetration testing will be split into different testing approaches. These two suggested actions will enhance the security of the development process.

Figure 7.2 presents the intervention state of the process model arising from the conduct of the focus group workshop. It shows how security collaboration has been improved in agile software development through the introduction of the compliance sprint ceremony. This has also filled the literature gap on the non-existence of security compliance in agile software development. The diagram also shows that the penetrating testing has been split into manual and automated. While automated testing is important in detecting vulnerabilities, manual testing can detect cleverer threats which automated test may miss.

Table 7.1: Preliminary Model Validation Participants

Practitioners		**Cybersecurity Experience (Years)**
Participant Code	**Roles**	
ITSERVco1_VP-SSA&E	VP, Security Strategy, Architecture & Engineering	29
ITSERVco4_ITS	Manager, IT Security	26
CYBERFOUDco1_ADL1	Cybersecurity Analyst	11
ITSERVco4_SSA	Security Solution Architect	7
CYBERFOUDco1_SE	Security Engineer	13

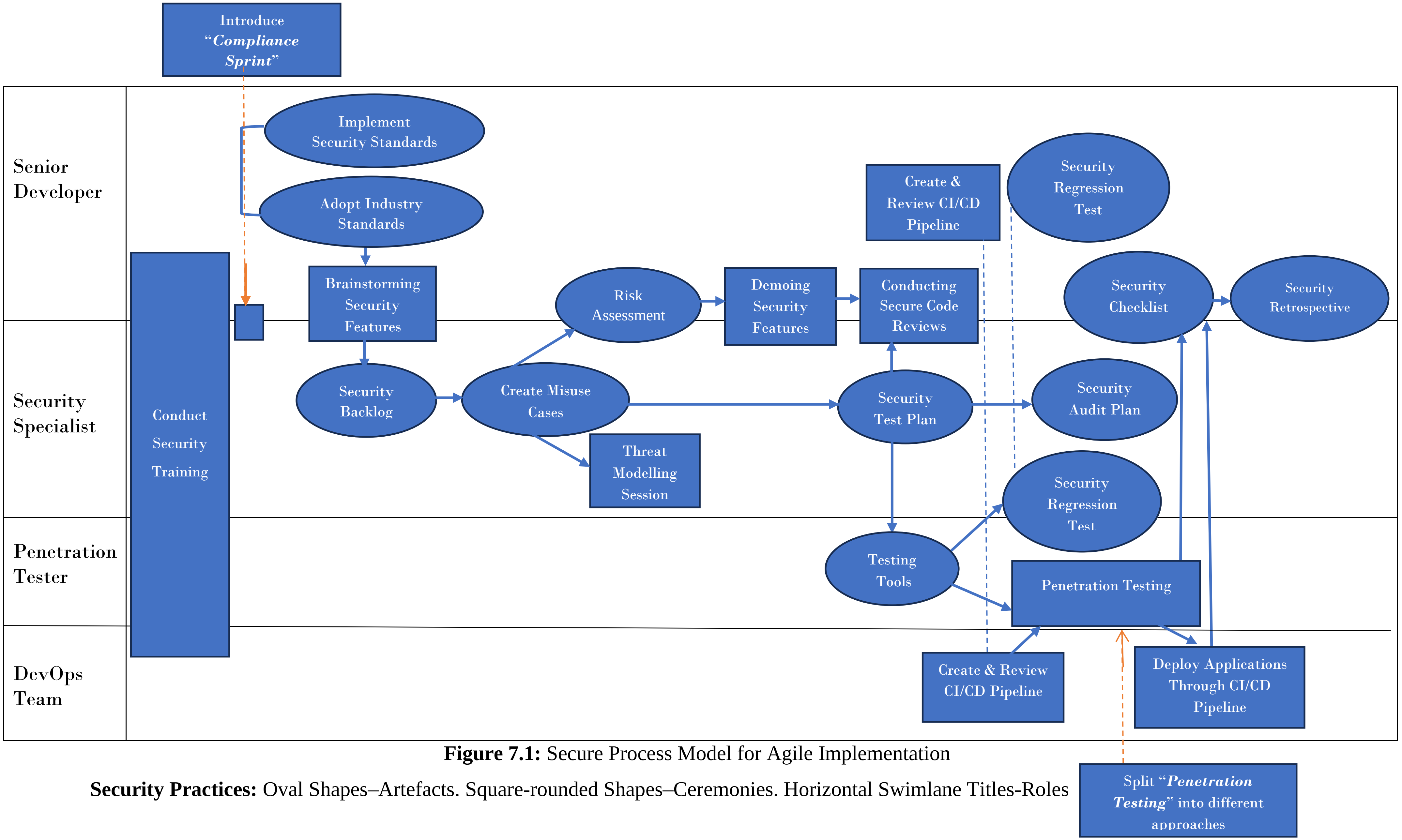

Introduce "*Compliance Sprint*"
Senior Developer
Security Specialist
Penetration Tester
DevOps Team
Implement Security Standards
Adopt Industry Standards
Conduct Security Training
Brainstorming Security Features
Security Backlog
Create Misuse Cases
Risk Assessment
Threat Modelling Session
Demoing Security Features
Conducting Secure Code Reviews
Create & Review CI/CD Pipeline
Security Regression Test
Security Checklist
Security Retrospective
Security Test Plan
Security Audit Plan
Security Regression Test
Testing Tools
Penetration Testing
Create & Review CI/CD Pipeline
Deploy Applications Through CI/CD Pipeline
Split "*Penetration Testing*" into different approaches

Figure 7.1: Secure Process Model for Agile Implementation

Security Practices: Oval Shapes–Artefacts. Square-rounded Shapes–Ceremonies. Horizontal Swimlane Titles-Roles

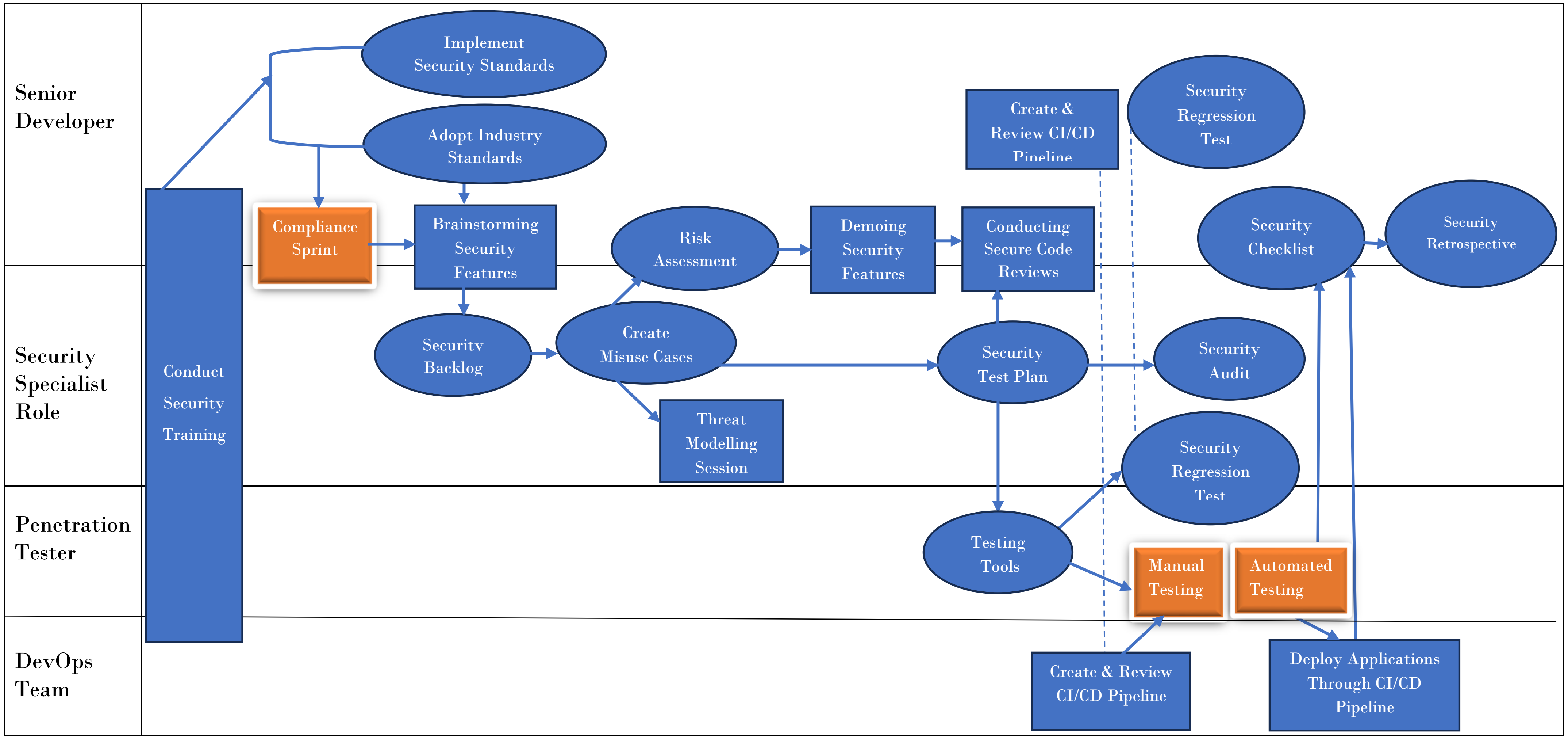


Figure 7.2: Process Flow – Intervention state.

Security Practices: Oval Shapes–Artefacts. Square-rounded Shapes–Ceremonies. Horizontal Swimlane Titles-Roles

7.3 Model Evaluation – Focus Group Session with HealthCo1

The evaluation of the model at HealthCo1, a software development company in Nigeria, involved conducting two focus group sessions and five post-implementation semi-structured interviews. First, an initial engagement with the Security Team Lead and a Project Manager was held to explain the aim of the research. After gaining permission, a workshop on secure agile software practices was conducted. The aim of the workshop was mainly to introduce participants to secure practices adopted in organisations for software development and to further raise awareness of secure development process. The discussions during the session motivated the company to agree on another workshop. The PhD research findings were presented at the workshop. This included the taxonomy of secure agile security practices categorised into roles, ceremonies and artefacts, and implications of regulatory policy for building secure agile software. Before introducing the secure software process model, several questions were asked to explore the current state of agile security implementation. Some of the questions include: 1. What practices do you adopt to ensure security in the development process? 2. How do you integrate security practices into development processes? 3. How do you decide on designs and security frameworks to adopt? 4. What security practices are most relevant in your development activities? These questions were explained clearly to participants to avoid any ambiguity.

The second objective of the workshop was to conduct a gap analysis between the secure process model and existing practices at HealthCo1 with the aim of positively influencing company processes. The model evaluation process was iteratively conducted starting with describing the outcome of integrating security practices collated from research sites (baseline state) as presented in figure 6.1, then the pre-intervention state in figure 7.1, and the preliminary validation is presented in figure 7.2. Figure 7.3 summarised the model evaluation engagement at HealthCo1 while the final evaluated model is presented in figure 7.4. The model elements as well as it's implemented processes were clearly explained. The session noted some gaps between the processes at HealthCo1 and the designed secure software process model. Table 7.2 shows an inventory of all the security practices which were deduced from the swim lane diagrams presented. All cells marked X represents the practices adopted at HealthCo1. Finally, the researcher and participants agreed for the implementation of the security retrospective in a project been executed at the company and report on any impact noticed through individual semi-structured interviews. The model evaluation had 7 participants – 1 CTO,1 Project Manager, 2 Senior Software Engineers, 1 Security Analyst and 2 Software Developers.

7.4 Model Evaluation Findings – Gap Analysis

This section presents the findings of the gap analysis of HealthCo1's secure software development processes with the practised-based model developed.

Table 7.2: Security Practices

Security Practices	**Proposed Model**	**HealthCo1**
Baseline Security Standards	X	X
Industry Regulatory Standards	X	X
Compliance Sprint	X	---
Conduct Security Training	X	X
Brainstorm Security Features	X	---
Security Backlog	X	---
Create Misuse Cases	X	X
Threat modelling session	X	X
Security Risk Assessment	X	X
Demoing Security Features	X	X
Secure Code Review Session	X	X
Security Test Plan	X	X
Testing Tools	X	X
Security Regression Tests	X	X
Penetration Testing	X	X
Security Retrospective	X	---
Secure CI/CD Pipeline Review	X	X
Deployment through CI/CD Pipeline	X	X
Security Audit Plan	X	X
Security Patching	X	X
Security Checklist	X	---

X – Adopted Security Practices; NX – Security Practices Not Adopted

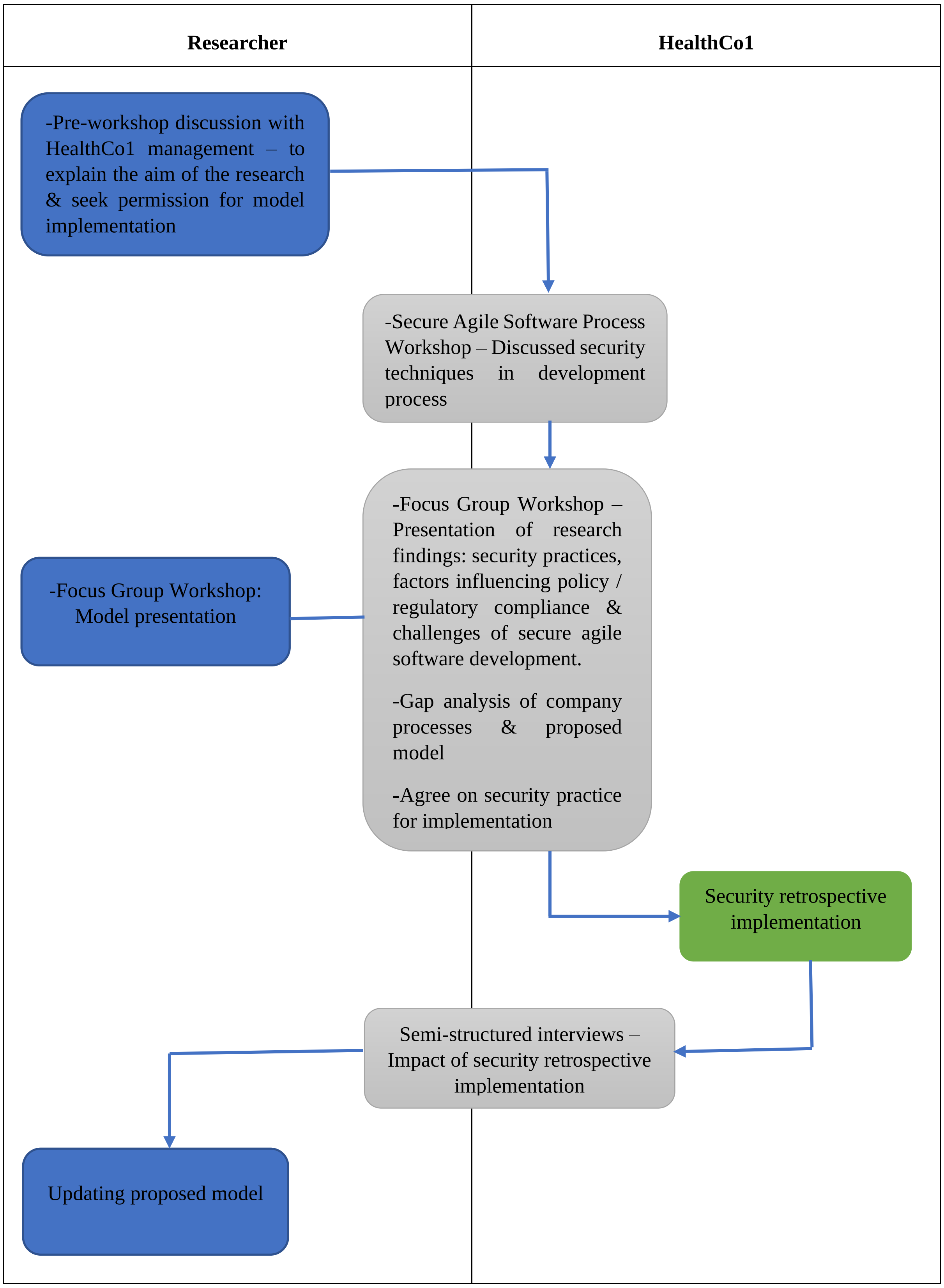

Researcher
HealthCo1
-Pre-workshop discussion with HealthCo1 management – to explain the aim of the research & seek permission for model implementation
-Secure Agile Software Process Workshop – Discussed security techniques in development process
-Focus Group Workshop: Model presentation
-Focus Group Workshop – Presentation of research findings: security practices, factors influencing policy / regulatory compliance & challenges of secure agile software development.
-Gap analysis of company processes & proposed model
-Agree on security practice for implementation
Security retrospective implementation
Semi-structured interviews – Impact of security retrospective implementation
Updating proposed model

Figure 7.3: Secure Software Process Model Evaluation at HealthCo1

7.4.1 Security Techniques in Practice

The taxonomy of agile security practices presented during the focus group contained many practices currently been adopted at HealthCo1. However, the company also has several other practices and techniques they use to ensure their software are secure and can withstand cyberattacks. Some of the security practices adopted at HealthCo1 includes quality gates, application security settings definition and documentation of security solutions. Although most of the practitioners are very knowledgeable about agile software development, only those with many years of experience and those with security job roles are conversant with secure development processes. Team members seem to be self-motivated and showed enthusiasm to learn about secure agile software practices to proactively guard against cyberattacks as evidence through discussions during the workshop. HealthCo1_CTO1 have reported adopting informal meetings such as brown bag sessions to share security knowledge within the organisation.

> "*So, … generally I coordinate the entire security activities of the development process in addition to my other responsibilities as CTO. The team generally does the development while I guide them to ensure they are incorporating all the essential activities that I feel will make the software secured. I ensure things like threat modelling is properly done to ensure our software are not vulnerable to attacks*" (Chief Technology Officer, Company Q).

To adopt security practices during the development process, it depends on a project's requirements as explained by HealthCo1_SSE1:

> "*Adopting security practices in our software development process is dependent on the needs of the products requested by clients. If we think adding a job role or conducting certain meetings will make it more secure, then we go ahead and adopt. Apart from this, there are instances the CTO request us to perform certain activities...*" (Senior Software Engineer, Company Q).

7.4.2 Absence of Security Retrospective

Apart from the compliance sprint earlier proposed to reduce collaborative gap between teams, adopting security retrospective is another way of improving team collaboration. Interviewed practitioners highlighted the absence of formal security specific meetings to reflect on what happened in preceding iterations. According to HealthCo1_SSA: "*We do not have kind of structured meetings after each iteration that are entirely dedicated for reviewing security issues*" (Software Security Analyst, Company Q). While some practitioners understand what a retrospective is, the session is conducted to discussion the entire software development process and so not exploring security issues in-depth. This research discovered that retrospective is usually conducted based on the client's operating sector, public or private. HealthCo1_SSA further explained:

> "*Usually if the software been developed is for a government agency, it is not in all cases that the CTO calls for a meeting after the product has been developed but for private-sector clients he always does*" (Software Security Analyst, Company Q).

Also, HealthCo1_SSE2 added: "*When developing software for companies, the Security Lead, Security Analyst and myself take charge of the security aspect of things*" (Senior Software Engineer, Company Q).

The rationale for the different reasons for conducting retrospective in this study was explained by the HealthCo1_STL. For the Public sector, the HealthCo1_STL explained: "*I and the other senior engineers sometimes chat over any vulnerabilities we observe because ordinarily government officials do not care much about these things so why waste our time on it*" (Security Team Lead, Company Q). Also, sometimes the CTO explained that there are instances when government clients request so many features in an application that they don't even need just because they have seen something similar in developed countries. For example, the HealthCo1_SSA explained:

> "*This government people just ask you to develop a system with so many features that they don't get to use and so won't even know if there is a flaw in what has been delivered"* (Software Security Analyst, Company Q).

On the other hand, the HealthCo1_SSA explained that private companies are business-oriented which requires software companies to hold more meetings and ensure all project requirements are met. According to the HealthCo1_SSA, "*We are very careful when developing software for private* *sector players to hold more meetings to ensure we get everything right*" (HealthCo1_SSA). The HealthCo1_SSA mentioned that one mistake might cause a company to lose a big client when dealing with the private sector. The Security Analyst further explained that they lost a big client who is involved in projects for the digitization project of the primary healthcare record in Nigeria for mistakenly adding a feature that was not requested. HealthCo1_SSA explained:

> *"I remember in 2020 when we were denied a contract by XXX simply because we mistakenly added a feature that was not requested and failed to incorporate two features that they requested. XXX IT Manager said they deal with critical patient data that doesn't give room for mistakes or not understanding requested requirements*" (Software Security Analyst, Company Q).

Interviewed practitioners have displayed fair knowledge of conducting retrospective meeting but no in-depth discussion on security. Also, the rationale for conducting more meetings when handling private sector contracts and less when handling government projects were explained.

Adopting Security Retrospectives

Depending on the client whose project is been handled, practitioners may feel motivated or otherwise about conducting a security retrospective. As most of the projects handled by HealthCo1 are for government agencies, practitioners feel it was a waste of time conducting security retrospectives for government projects.

The security retrospective conducted at HealthCo1 was tailored towards the Nigerian software sector. Participants have two options: either to voice their concerns during meetings or scribble it on a piece of paper for the security team lead to read it to the hearing of all. The participants were asked to ponder over the following four questions relating their responses to security concerns: "What went well?", "What didn't go so well?", "What have I learned?", & "What still puzzles me?". The adopted method of feedback mechanism has ensured that all participants were involved in the development process. HealthCo1_SD1 explained: "*The methods ensured there is a way for everyone. If you are shy or fear what the management might victimize you, writing it down makes you anonymous*" (Software Developer, Company Q). Thus, it was discovered that practitioners were more willing to express their views freely.

7.4.3 Lack of Security Backlog

Backlog in HealthCo1 mainly contains the prioritized list of work items requested by clients without much emphasis on security tasks. There is no separate security tasks list as adopted by many agile teams. The items related to software security are mostly discussed but not necessarily documented. HealthCo1_CTO1 mentioned that the security activities to be considered in a project are discussed in each iteration but not documented.

> "*So, this is one thing I think we need to improve. . .certainly, having separate product backlog and security backlog will increase our team's visibility of security issues. However, sometimes the honest truth is the level of competence to do all these security tasks without my inputs, it's hard*" (Chief Technology Officer, Company Q).

7.4.4 Security Compliance Challenge

Compliance to security standards in HealthCo1 is almost non-existent. Although HealthCo1 is an organization operating in the healthcare sector where there is much emphasis on safeguarding patient data, company size and maturity is a factor impeding compliance. For instance, HealthCo1_CTO1 stated: "*Looking at our level of maturity as a company, understanding all these industry standards is a challenge for the team and the issue of compliance is also a burden*." (Chief Technology Officer, Company Q). Balancing between compliance to security standards and policies and delivering secure software that meets client's needs remains a challenge. HealthCo1_SD2 have expressed ignorance of the security standards existing in the healthcare industry.

> "*I have heard people talking about HIPAA, I have forgotten the full meaning, but I know it has to do with patients' information, but I can't explain its details. Project managers sometimes try to explain these things, but the fact is there is no time as clients are always on your neck to deliver their software which is what management are also more concerned about*." (Software Developer, Company Q).

7.4.5 Top Management Challenge

Handling top management supremacy on practitioner's job responsibility is a challenge faced by junior-level engineers. According to HealthCo1_SD1: "*There are many instances when the top management do not support your ideas and assert their dominance over all issues*" (Software Developer, Company Q). The HealthCo1_SD1 further explained scenarios when management disregards inputs from employees even on issues that posses' great security risks to the organization and its products. HealthCo1_SD1 explained: "*The CTO mostly takes unilateral decisions detrimental to the software been developed and company reputation*" (Software Developer, Company Q).

The top management mostly base their decision on costs rather than the quality of the software. HealthCo1_SSE2 explained: "*I sometimes get shut-down by the CTO during our sprints if what I am suggesting is going to cost the company a lot without allowing me explain its benefits*" (Senior Software Engineer, Company Q). HealthCo1_SSA corroborated a similar experience when he suggested the adoption of PractiTest software tool. According to HealthCo1_SSA, "*When I suggested we adopt the PractiTest tool, the CTO simply said he thinks that will be a waste of money*" (Software Security Analyst, Company Q). Thus, interviewed practitioners complained about top management been more interested about cost and making profit rather than the security of software.

7.5 Model Discussion: Practitioners' Views

This section discusses how practitioners' view the practice-based process model developed. The practitioners were asked about the relatability, suitability, and clarity of the proposed model in relation to their job roles and software development processes.

7.5.1 Relatability

The software development processes at HealthCo1 are not very well structured when compared to the proposed research model. HealthCo1_SSE2 agrees that there are elements of the model useful and relatable to their company processes.

> *"Many of the elements here are useful that even me working dint think about it. . . Another good thing I see here is having the security specialist and penetration tester as different roles. My CTO and SA always try to have another person who could take up a role if the engineer assigned is not available. However, it is not always easy getting the right fit of competence especially with security tasks*" (Senior Software Engineer, Company Q).

Similarly, HealthCo1_SSA analysed the relatability of the developed process model in relation to his current job role. HealthCo1_SSA explained:

> *"Currently, I am in the security specialist line. . . secure code review sessions, secure coding template are what I am involved in as a Software Security Analyst. In my organization we don't have a security backlog or create misuse case. So, I think your model is really defined properly with all these components*" (HealthCo1_SSA, Company Q).

7.5.2 Suitability

Practitioners were unanimous that the proposed model contained elements capable of improving the security of their software development process as explained by HealthCo1_PM:

> "*Yeah, I think following the development lifecycle view you adopted here and baking many of your model's security practices would be helpful for us as a company. . . Although I know everything on the model is important, for me, I have not really seen much of security regression testing both here and in other places. I think it's important because most of the breaches you see are due to flawed security in development process*" (Project Manager, Company Q).

7.5.3 Clarity

Explaining what different shapes represents to practitioners attributed to the ease of understanding the proposed model. The researcher knew that not all interviewees were grounded in secure software development; thus, all the model components were explicitly explained again at every interview or workshop. HealthCo1_PM explained:

> "*To be honest I liked the way you took me through the model processes and so I dint need to figure things out myself from this diagram. The vast majority of the boxes there are clear, and the layering is also straightforward*" (Project Manager, Company Q).

7.6 Call to Action: Model Post-Implementation Evaluation

This section discusses the impact of the process model implementation at HealthCo1. Five interviews were conducted with three practitioners involved in the model evaluation process. The average duration of the interviews was 35 minutes, and they were audio-recorded, transcribed, and analysed using techniques explained in section 3.6.3. The interview quotes were used to validate the research data. Finally, the developed process model was updated to illustrate the introduced security role as shown in figure 7.4.

The findings from the model post-implementation have demonstrated the need to introduce a middle-level role to assist the CTO handle security tasks. Before the model implementation, the CTO was responsible for leading all security discussions of the development process. Significant impact has been reported by interviewed practitioners. By incorporating security retrospective in their development process, junior engineers were given the opportunity of expressing their views in contrast to been dominated by top and middle-level managers during meetings. HealthCo1_CTO1 explained:

> "*Your model has made us reflect on our processes and we now use the start, stop, continue retrospective technique. This has helped us better understand our security strengths as well as strategize on ways of overcoming weaknesses*" (Chief Technology Officer, Company Q).

The model application at HealthCo1 has also improved the participation of junior engineers in the company in contrast to what was reported in the past. For example, HealthCo1_SD1 explained the improvement he noticed in his company:

> "*I noticed a slight flexibility in the conduct of the security retrospective introduced due to our engagement with you… In the 2nd and 3rd iterations of the XXX project, the CTO asked each of us to list 3 security-specific issues we encountered*" (Software Developer, Company Q).

The model implementation has also motivated the organisation to seek further collaborations aimed at improving security. For example, HealthCo1_SSA explained:

> "*I can invite the security architect of XXX to share his experience with us in a brown bag session as I know they have adopted the security retrospective in their agile processes for some time now*" (Software Security Analyst, Company Q).

7.6.1 Overcoming Collaborative Gap

To further overcome the collaborative gap identified between teams as noted in practice (the model) as well as in literature as discussed at the beginning of this thesis, the security champion has been introduced to the secure agile process at HealthCo1. The company management did a skills analysis of their current mid-level engineers to identify who can better take up the responsibilities of a security champion. As explained by HealthCo1_CTO1: "*I assessed all our mid-level engineers looking at their years of experience and certifications they have and selected XXX who I think can take up the role of a security champion.*" (Chief Technology Officer, Company Q).

The introduction of the security champion role although initially misunderstood in the company has been reported as a success. The introduction of the new role to the model was aimed at spreading security knowledge in teams and reducing collaborative gap. Interviewed practitioners highlighted its impact on company secure practices at Healthco1. The security champion's responsibilities assigned to HealthCo1_SSA has moved some major duties off the CTO for him to focus more on management issues and other tasks. As HealthCo1_CTO1 explained, "*I can now focus on other general management issues while XXX takes up more security tasks including coordinating the security retrospective sessions*" (Chief Technology Officer, Company Q).

In addition to introducing the security champion role to the model, the security retrospective implemented at Healthco1 has helped to overcome the challenge of lack in secure agile ceremonies. Practitioners have also expressed more freedom engaging with a mid-level engineer than when a management personnel is chairing a meeting. HealthCo1_SSA explained: "*XXX and XXX told me they feel freer to discuss things if their immediate boss [me] ask them about things than if it was* *the CTO*" (Software Security Analyst, Company Q). In addition, HealthCo1_SD1 said: "*Since we get to discuss issues in-depth during security retrospective, I do hear new things mentioned by others which I can look-up later or ask for trainings on them from the company*" (Software Developer, Company Q). Thus, with the implementation of more secure agile roles and ceremonies, security knowledge across the organization was enhanced.

Figure 7.4 represents the final secure-by-design model after the preliminary validation and in-depth implementation at HealthCo1. The points of change are indicated with call-outs where security practices were implemented. These include introducing the security champion role, conducting a compliance sprint meeting, and performing retrospective after each iteration.

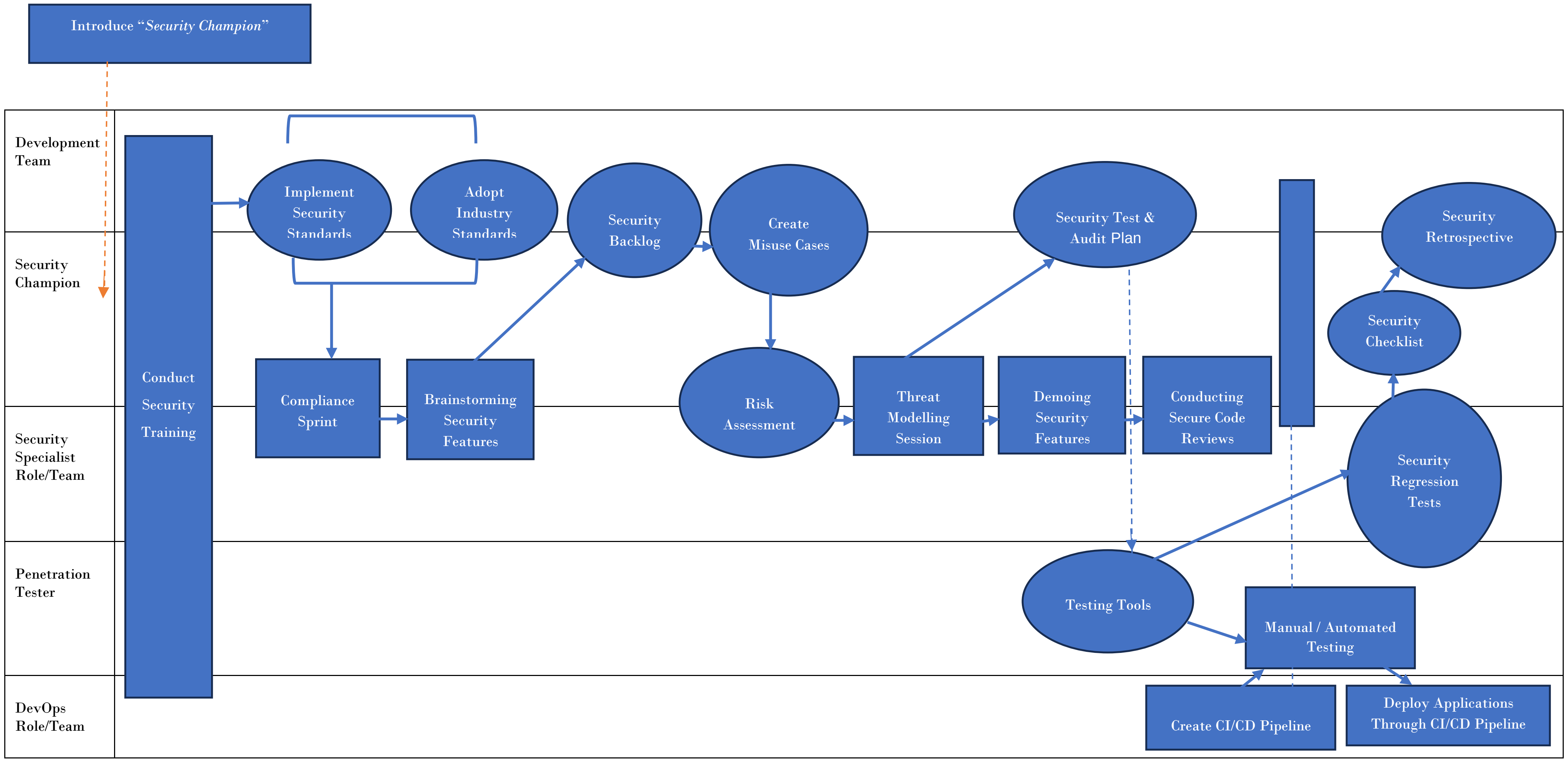

Introduce "*Security Champion*"
Development Team
Security Champion
Security Specialist Role/Team
Penetration Tester
DevOps Role/Team
Conduct Security Training
Implement Security Standards
Adopt Industry Standards
Security Backlog
Create Misuse Cases
Security Test & Audit Plan
Security Retrospective
Security Checklist
Compliance Sprint
Brainstorming Security Features
Risk Assessment
Threat Modelling Session
Demoing Security Features
Conducting Secure Code Reviews
Security Regression Tests
Testing Tools
Manual / Automated Testing
Create CI/CD Pipeline
Deploy Applications Through CI/CD Pipeline

Figure 7.4: Secure Agile Software Development Process Model

Security Practices: Oval Shapes – Artefacts. Square-rounded Shapes – Ceremonies. Horizontal Swimlanes Titles – Roles.

7.6 Summary

This chapter presents findings of the preliminary secure agile model validation as well as the evaluation conducted at HealthCo1. It also explained how the implemented security practices positively impacted the software development process. Findings of the post-model implementation evaluation were also presented.

Chapter 8

Discussion

8.1 Introduction

This chapter answers the research questions. The analysis of the findings in relation to pertinent literature were also presented. The research contribution to theory and practice as well as the study limitations are discussed.

8.2 Answering the Research Questions (RQs)

As discussed in chapter one, this research aims to explore practitioner perceptions of security practices and develop a secured agile process. This goal propelled conducting an in-depth investigation of the security activities undertaken by agile practitioners in various companies during software development. To achieve this aim, this chapter answers the four research questions formulated in section 1.3.3.

Research Question 1: How do selected practitioners describe the current state of agile security practices implementation for software development?

To answer the above question, a novel taxonomy of security practices based on interviewees' description was developed. The taxonomy comprised of security practices adopted by organisation in agile software development as shown in figure 4.1. The taxonomy is described based on agile use in organisations – roles, ceremonies, and artefacts. As described in chapter 4, the taxonomy consists of ten artefacts, four ceremonies, and four roles. The identification of more artefacts than roles and ceremonies point to lack of collaboration within agile teams. The ten artefacts identified are: vulnerability assessment tools, code verification tools, API testing tools and Git-Hub test platform. Other artefacts include security backlog, creating misuse cases, risk assessment, security baseline standards, security test plan templates, and security audit checklists. The four identified secure agile ceremonies include threat modelling sessions, secure code reviews, brainstorming sessions, and conducting security trainings. Finally, the three roles involved in the development process includes security specialist, penetration tester, senior developers, and DevOps. The practices identified in the taxonomy are similar to those in other existing studies (Mihelič et al., 2023; Rindell, Ruohonen, et al., 2018).

Developing taxonomies to systematically describe components, including their connections to each other, is a well-established method in the field of software engineering (SE) (Usman, Britto, et al., 2017). However, in the agile software development, there has been limited investigation, specifically regarding the categorization of security practices into roles, ceremonies, and artifacts. Although there are few studies that address taxonomies in other contexts, such as requirements engineering (Saher et al., 2017), and effort estimation (Usman, Börstler, et al., 2017), within agile methods in software development. Likewise, a taxonomy for large-scale agile software development, categorizing projects into small, large, and very large scales has been proposed (Dingsøyr et al., 2014). The classification is based on the number of agile software development teams and the coordination approaches adopted by these teams in a project. Besides the software engineering domain, developing taxonomies are important in other research fields, including information systems. Despite the existence of taxonomies, as explored in this section (Neumann, 2021), there is still paucity of research that have integrated practices from the three agile components – roles, ceremonies, and artifacts (Ardo et al., 2022).

Rindell, Ruohonen, et al. (2018), conducted a survey in selected Finnish software development companies to explore their security practices. This research however differs from their study as the surveyed practices were drawn from literature and linear models such as BSIMM, Microsoft SDL and the Finnish government framework, VAHTI. While the study by Mihelič et al. (2023) identified security elements (roles, ceremonies, and artefacts) adopted in secure agile software development, again the elements were derived from literature. Another study by (Rindell et al., 2021), conducted a practitioner survey to investigate the state of the art of security practices used during agile software development. The study extracted forty security practices from various models which were grouped into SDLC lifecycle. With increasing security concerns, the study suggested examining the effects of security practices as regulatory policies are constantly growing in response to cyberattacks. My research investigated the impacts of regulatory policy and standards for building secure agile software in RQ2.

Research Question 2: What are the impacts of regulatory policies and standards for developing secure software using agile methods in the case study companies investigated?

This study found that implementing the Nigeria Data Protection Regulation (NDPR) impacts the security practices of agile teams. The study developed a grounded theory (GT) of security challenges which was termed policy adherence challenges (PAC) model. Study findings also revealed tensions between the regulatory environment in Nigeria and agile software developers' compliance. Thus, the four challenges identified include the following: (a) a lack of collaboration between security and agile teams; (b) the tendency to use foreign software hosting companies; (c) a poor cybersecurity culture; and (d) the high cost of building secure agile software.

The existing literature have described many factors influencing the non-adherence of practitioners to regulatory policies in software development (Nägele et al., 2022b; Sebega & Mnkandla, 2017; Tøndel et al., 2022; Venson et al., 2019). First, findings of this study corroborated the research by Tøndel et al. (2022), which described the collaborative nature between agile and security teams as often sub-optimal. Conflict was reported in balancing team autonomy and security governance when collaborating between different teams. In Nägele et al. (2022b), two new roles were proposed to improve collaboration between teams, however, that will only feasible in large-scale agile development (LSAD). While the roles appear helpful, it might not be suited for this study context as most software companies in Nigeria are SMEs. Even in the LSAD context, these new roles are just starting to emerge with the capacity required to improve security competence around cross-teams.

Second, this study discovered that the cybersecurity culture in participating organizations is self-taught where practitioners learn and teach their colleagues. This finding aligns with an earlier research by Bodin and Golberg (2021), whereby interested persons learn through reading blogs and watching videos. This study further revealed that practitioners rely on two elements of cybersecurity culture to ensure adherence to regulatory policies. These elements include artifacts (awareness training and employee behaviour or mindset) and values (security code of conduct or guidelines). There were practitioners in this study with very few opportunities to engage in security training who relied on senior engineers for security knowledge. They stated this was due to the cost which most companies - especially SMEs - in Nigeria cannot afford. A study by Alshaikh (2020), advocated for practitioners to move beyond just security education, training, and awareness (SETA). The study implemented the SETA approach in three Australian firms to show the transformation from compliance to building a cybersecurity culture. In contrast to the cybersecurity culture in global north companies, this research reveals that within the ICT4D landscape, organizational culture significantly influences how security concerns are addressed. This research highlights the presence of coercive pressure, characterized by external authoritative influences like regulatory policies such as NDPR. However, there is a lack of normative and mimetic institutional pressures, which typically play a role in organizations gaining legitimacy from external institutions. Legitimacy, in this context, refers to the accepted actions of an entity within a socially constructed system of norms, values, and beliefs. This study emphasizes the importance of institutional legitimacy, defined as organizations conforming to pressures that shape their behaviour (Lui et al., 2016), particularly in adopting software security practices. Despite coercive pressure, the research underscores the need for varied sources of influence to ensure organizations obtain legitimacy and avoid potential sanctions. For instance, the research findings indicate that some practitioners are unaware of the NDPR legislation, and even those who are aware do not feel compelled by to comply.

Third, this study identified three major sources of cost when building secure agile software in Nigeria. These include the additional administrative costs of maintaining data centres, the cost of outsourcing security experts, and issues around the exchange rate of purchasing infrastructure. Some aspects of the results corroborate the findings of Sebega and Mnkandla (2017), which highlighted that non-functional requirements (i.e. security and safety) in secure agile software development are not always considered due to costs. This leads to the potential compromise of poor software quality. However, in contrast to this study findings, Venson et al. (2019), identified that conducting security reviews, applying threat modelling, and performing security testing are the three most significant sources of cost associated with building secure software. The study reported that ‘security-by-design’ paradigm was the lowest source of cost, with only one study in the literature.

While the findings of this study are comparable to some of the existing literature discussed in this section, the PAC model presents valuable insights into practitioner challenges in a developing country context where existing empirical studies are lacking. The developed theory explains how the challenges to adherence were discovered based on emergent empirical evidence.

Research Question 3: How can the identified security practices in the case study companies be integrated to create an agile process model that suits software development needs and organisational context?

This question was predicated on the lack of security practices integration in agile software development process (Khaim et al., 2016; Rindell et al., 2021). The existing studies highlighted the need to integrate security practices into the development process to mitigate cybersecurity attacks (Rindell et al., 2021). Analysing identified security practices from research sites led to the development of a novel practice-based process model as described in chapter 6. The model consists of security practices which were mapped onto a swim lane diagram.

The developed secure agile process model shares similar practices to some of the existing studies (Baca et al., 2015; Bezerra et al., 2020; de Vicente Mohino et al., 2019; Rindell, Hyrynsalmi, et al., 2018). However, this study found the lack of compliance to regulatory policy and security standards during the agile software development process as discussions around this issue is rarely held. The process model supports the findings in (Moyón et al., 2020), which discussed the almost non-existent compliance to security standards in the agile software development literature but didn't suggest ways of improving it. While the study acknowledged security testing as a critical aspect of secure agile software development for ensuring compliance with testing requirements, it did not delve into the specifics of how and when to create testing plans within the process. The secure agile process model in (Baca & Carlsson, 2011), was developed based on security activities from three well-known engineering models: Microsoft SDL, Cigatel touchpoints and Common Criteria (Baca & Carlsson, 2011). The model has also not been practically implemented in an organisational setting. There are other existing studies such as (Mohan & Othmane, 2016; Oueslati et al., 2015; Villamizar et al., 2018), which are focused on theoretical discussions of how to integrate security standards into agile methods. However, there is still limited empirical evidence on regulatory compliance in agile software development (Usman et al., 2020). Thus, this study provided a practise-based model and proposed security practices to improve collaboration as given in chapter 7, sections 7.2 and 7.6.

Research Question 4: What are the selected practitioners' perceptions of the effect of implementing a secure agile process model on improved software development process?

My model was evaluated in two-phases. First, a focus group workshop was held to validate the proposed model in the UK. To improve collaboration and compliance to security standards and regulatory policy, the model evaluation proposed a new ceremony termed "compliance sprint" to improve collaboration between agile and security. The lack of collaboration on security promotes non-adherence to regulations. Previous literature had discovered the complexity and ambiguity of security standards which poses great challenge to development team members (Moyón et al., 2021). Such a scenario negates the principle of increased velocity in software development. Therefore, the proposed ceremony would improve collaboration and raise awareness on compliance to security standards in agile software development teams. While Dännart et al. (2019), developed a process model for security compliance in large scale agile environment, the model was only designed to assess compliance to IEC 62443-4-1 (4-1) standard. Another process model developed by (Moyon et al., 2018), for compliance in agile software development at scale. However, the model was also specifically designed to ensure organisational compliance to IEC 62443-4-1 for agile practitioners following SAFe. The practitioners also suggested certain modifications to the model which included splitting penetration testing into manual and automated tests.

The second phase of the model evaluation involved implementing it at a software development company in Nigeria. Adopting security retrospective has been reported to enhance security knowledge of agile practitioners. The existing collaborative gap in HealthCo1 was also reported to have reduced through the introduction of a security champion role. With the implementation of parts of the secure agile process model at Healthco1, security awareness and knowledge were enhanced. These was shown through the secure agile process model, from the baseline process flow in figure 6.1, it was revised through several iterations to reach the secured process model in figure 7.3.

The highlights of the model evaluation implementation phase are as follows:

1. Security compliance sprint, a new ceremony was proposed to reduce the collaborative gap between agile and security teams.
2. The role of a security champion was introduced to handle security issues during the development process. Table 8.1 shows some of the responsibilities of the security champion in comparison to other security specific roles
3. Security retrospectives was tailored so that junior level engineers can learn and enhanced their security knowledge by been engaged in the process.
4. The CTO handed most routine day-to-day security task to a middle level manager to have more time to concentrate on other issues such as compliance, and other innovative technological concerns.

Table 8.1: Comparison of security champion responsibilities to other roles

Security Champion	Security Guru/Security Developer/Security Master
• Identify organisational security needs • Increase security awareness • Review & escalate security concerns • Serves as a security advocate in the development team	• Design security test cases • Analyse potential threats • Conduct attack surface analysis • Handle all technical tasks related to organizational security

Based on the existing literature gap on the strategic importance of the security champion role and the limited knowledge about it (Aalvik et al., 2023), this study introduced the role to improve team security culture.

8.3 Research Contribution to Theory

This thesis contributes to the studies of regulatory compliance in secure agile software development through the Theory of Explaining, and the Theory of Design and Action following the structure proposed in (Gregor, 2006). The Theory of Explanation seeks to what is, how and why a phenomenon occurs. This thesis presents a Grounded Theory of Explanation on practitioners' challenges to adhering to regulatory policies. The emergent theory was termed Policy Adherence Challenges (PAC) model. The social process termed "policy adherence challenges", explains the reasons agile practitioners are refusing to comply with NDPR during software development. The developed GT followed the approach adopted in other previous studies in the domain of agile software development (Bass, 2016; Hoda et al., 2012; Masood et al., 2020; Shastri et al., 2021).

This thesis provides in-depth understanding on the security practices adopted by agile practitioners during software development. Results show correlation between practitioners' security knowledge, company maturity and regulatory landscape. The Theory of Design and Action in this thesis gives prescription for the development of a secure agile process. It has been further demonstrated through the evaluation of the secure agile process model seeking to positively influence company practices.

The novel findings of this thesis focus on regulatory compliance of practitioners and the security practices adopted in companies through the instrumentality of the developed model. The findings of this thesis have contributed to filling the research gaps identified in the security practices integration and regulatory compliance specifically in the Global South as highlighted in chapter two. To the best of the researcher's knowledge, and after an extensive literature on the investigated phenomenon, no existing study has presented findings as described in this thesis. This research is unique as it provides empirical evidence on regulatory compliance on secure agile software development which is lacking in literature. The creation of a secure agile process model offers the opportunity for other researchers to implement other practices with real software teams to ascertain their impacts.

8.4 Research Contribution to Practice

The secure agile software process model developed and presented in this thesis has implications for practitioners. While the model in its current form is not a standard recognise by regulatory bodies, adopting the practices contained therein will have positive impacts on company processes as shown in the validation phase. Thus, this section presents the model implications based on research findings and existing literature, for senior managers (CTOs, IT security manager), security practitioners (security specialist, penetration testers, cybersecurity analyst), software development practitioners (software developers, DevOps engineers, systems analyst, business analyst) and customers.

For senior managers like the CTOs and IT security managers, the research findings provide a strategy for reducing collaborative gap through the introduction of additional security roles such as security champion to take-up security tasks and responsibilities. This means senior management can concentrate on the business decision making instead of been involved in day-to-day security tasks of software projects. The secure agile model provides a clear understanding of the impacts of security practices in the software development process (Rindell et al., 2021). As organisations face increased threats of cyberattacks, the model would help senior managers decide which practices are suitable to adopt based on their company maturity.

The taxonomy of agile security practices developed in this study (Ardo et al., 2021), provide benefits to software development practitioners in many ways. First, it categorised security practices into roles, ceremonies, and artefacts and mapped them onto the SDLC. The identified practices can be used by practitioners to improve their development processes with regards to software security. Second, integrating different practices (roles, ceremonies, & artefacts) improves companies security activities as opposed to relying on single agile concept such as security roles in a team (Baca et al., 2015). Third, highlighted the security testing approaches used in practice to proactively guard against software cyberattacks. Fourth, in the absence of security dedicated role, implementing the taxonomy would help practitioners adopt suitable practices capable of improving team cybersecurity culture.

Not only practitioners, many customers lack security knowledge and awareness (Ionita et al., 2019; Oueslati et al., 2015). To improve software security and reduce the incidence of cyberattacks, software companies may consider engaging clients, like what was done in this research at the beginning of the model validation phase. The researcher conducted workshops on secure agile software development with team members to raise security awareness and introduce the model. The workshop motivated further engagements which resulted in the model validation at HealthCo1. Collaborating with customers will improve security awareness. Also, the description of the intervention in this thesis should help security champions understand how to disseminate security knowledge within agile teams. During the model validation, the security champion was introduced to take-up more security responsibilities. The security champion facilitated the retrospective session and ensured its success.

The creation of a practice-based process model (Ardo et al., 2022), offers better understanding of practical techniques to improve organisation's cybersecurity landscape. Practitioners can implement suitable practices of the secure agile model, to observe it's impacts on software development processes. This thesis aids better understanding of the security compliance sprint ceremony. Research findings shows the lack of compliance to security standards (Usman et al., 2020), which necessitated proposing a new ceremony. This study corroborates other existing literature (Moyón et al., 2020), which lacked empirical evidence on security compliance in agile software development.

8.5 Limitations of the Study

The assessment of rigour and trustworthiness in this study has adopted the methodological trinity of validity, reliability, and generalization (Tobin & Begley, 2004). The threats have been thought of from the viewpoint of the 3-phased multi-methods research approach adopted.

- **Validity:** It describes how to ensure the "trustworthiness" of the data collected (Grossoehme, 2014). In this study, validity ensured data richness is not lost during the model abstraction process. The secure-by-design software process model created from the grounded theory studies were checked for accuracy through a focus group sessions conducted both in Nigeria and the UK. Thus, the aim of the focus group was to check if the final model bears relationship to reality or not.

- **Reliability:** The essence of reliability in qualitative research is to ensure result consistency [25]. Documenting research decisions taken ensures that another researcher understands what was done and can be systematically repeated. The data collection, analysis and interpretation methods adopted were explained in the methodology chapter of this thesis. Also, the interview guides for this research are attached in the Appendix B.
- **Generalization:** Most qualitative research focus on studying a specific phenomenon within a certain population or context, hence generalization not usually expected (Leung, 2015). However, this concept is increasingly becoming pertinent in qualitative research. In this research, findings were generalized by collecting data from different participants working in diverse business sectors, ranging from financial services and healthcare, to manufacturing, IT services and educational software solutions. Using theoretical sampling technique of grounded theory, participants with different job roles such as security engineer, security solution architect, cybersecurity analyst, back-end engineer, senior DevOps engineer, and senior software engineer were interviewed. Various management roles such as vice president, security strategy, architecture & engineering, chief technology officer, security manager and project managers were also interviewed to provide multiple perspective on the research phenomena. While data collected through interviews maybe subjective, interviewing wide range of participants reduces bias (Diefenbach, 2009). Also, the focus group sessions conducted in the UK as part of the model validation process contained a mixture of both earlier interviewed practitioners as well as individuals not involved at the earlier stages of the study.

To further evaluate the trustworthiness of the research findings, the naturalistic approach of Lincoln and Guba was adopted (Lincoln & Guba, 1985). The four techniques used include confirmability, dependability, internal consistency, and transferability.

- **Confirmability:** This defines the researcher's ability to provide chain of evidence depicting the actual responses of participants without introducing bias (Cope, 1969; Glaser, 1978). The snowball sampling technique was adopted to avoid selection bias as only agile practitioners were involved in the research. To ensure confirmability, explanation of how data was interpreted, and conclusions were arrived has been detailed in the methodology chapter of thesis. The chapter illustrated how the related interview codes were merged to form concepts then categories and memos written before developing a grounded theory.
- **Dependability:** This defines the act of repeating the study systematically (Cope, 1969; Glaser, 1978). In this thesis, the same data collection; data analysis and interpretation methods were

used which has been clearly explained in the methodology chapter. This will allow for the replication of the study by other researchers. Also, the research interview guides comprised of open-ended questions rather than directed questions since the research aim was to explore practitioner perceptions in detail. The interview guides are attached in Appendix B.

- **Internal Consistency:** It indicates the credibility and consistency of the research findings (Cope, 1969; Glaser, 1978). Participants responses were described as clearly as possible. Verbatim quotes from participant interview transcripts were included chapters four and five of this thesis for readers to derive meaning from interviewees perspective.
- **Transferability** studies the applicability of the research findings to another context (Lincoln & Guba, 1985). While an in-depth study was conducted in Nigeria, the results may be transferable to similar contexts in the Sub-Saharan Africa region in countries like Ghana, Rwanda, and Tanzania. Nevertheless, the unit of analysis in this research are the agile practitioners and their perceptions of security in the development process, the challenges confronting them and not the research sites which contributes to the generalizability of our study.

8.6 Summary

This chapter presented the research findings and analysed it in relation to existing literature. The research contributions to theory and practice were discussed. Finally, limitations of the study were highlighted.

Chapter 9

Conclusion

9.1 Research Summary

Increased reliance on software, magnified cyber-security threats and accelerating adoption of agile development methods highlight the need for enhanced secure software practices. With the increase in software development and use, cyberattacks has also increased (Nägele et al., 2022b). In the aftermath of the Covid-19 pandemic, software and by extension the Nigerian digital economy sector has recorded over $4.4 billion investments between 2019 and 2023 highlighting the increased reliance on software systems (Adepetun, 2023). As Nigeria is regarded a software hub and unarguably the largest economy on the African continent, cybercrimes are prevalent in organisations. Thus, agile practitioners and security professionals continue to face a lot of challenges throughout the development process.

This thesis conducted an extensive review of relevant literature on the integration of security practices in agile software development process. This study found evidence of practitioners adopting security practices, however, integrating those into the entire development process to create a more secured software remains a challenge (Valdés-Rodríguez et al., 2023). Also, this thesis explored security compliance in agile software development. Compliance to regulatory policies and standards were found to be lacking due to the paucity of empirical evidence (Moyón et al., 2020; Usman et al., 2020). Therefore, these issues motivated an in-depth investigation as presented in this thesis.

The widespread adoption of agile software developments methods can be attributed to its flexibility. According to the "State of Agile Report", there has been an exponential rise in agile methods adoption from 37% in 2020 to 86% in 2021. However, agile principles (Fowler & Highsmith, 2001), conflicts with security. Generally, security has been considered as part of the non-functional requirements in the development process rather than integrating it in all the SDLC phases (Futcher & von Solms, 2012). Other studies have considered security towards the end of the software development process (de Vicente Mohino et al., 2019). Furthermore, security compliance to regulatory policies and standards are getting attention due stricter regulations by various Governments and growing fines and sanctions (Breaux & Antón, 2008; Nägele et al., 2022b).

This research was aimed at improving the security of agile software development process through the development of a secured agile software process. To achieve the aim of this thesis, four questions were formulated as follows: **Research Question 1:** How do selected practitioners describe the current state of agile security practices implementation for software development? **Research Question 2:** What are the impacts of regulatory policies and standards for developing secure software using agile methods in the case study companies investigated? **Research Question 3:** How can the identified security practices in the case study companies be integrated to create an agile process model that suits software development needs and organisational context? **Research Question 4:** What are the selected practitioners' perceptions of the effect of implementing a secure agile process model on improved software development process?

This study adopted a multi-methods research design approach. All methodological choices were extensively discussed in chapter 3 together with their rationale. The research was divided into four phases.

Phase 1 was an exploratory case study which explored the practices adopted by organizations for secure software development. The study developed a novel taxonomy of agile security practices categorized into roles, ceremonies, and artefacts. The study findings revealed the existence of more artefacts than ceremonies and roles which invariably shows the lack of collaboration between practitioners.

Phase 2 conducted an in-depth investigation of secure agile software implementation in Nigeria. The study noted the existing tension between the Nigerian regulatory environment and practitioners' compliance. Based on the analysis of interview transcripts, a GT was developed termed Policy Adherence Challenges (PAC) model. The GT identified four challenges confronting Nigerian agile practitioners: a lack of collaboration between security and agile teams; the tendency to use foreign software hosting companies; a poor cybersecurity culture; and the high cost of building secure agile software. Although practitioners acknowledged the government's efforts, the practicality of implementing such legislation remains a challenge. These findings led to the conclusion that there is a lack of indigenous software hosting companies in Nigeria. This thesis recommended increased government action by raising public awareness of the capabilities of the few local software hosting companies', and closer collaboration between agile and security teams.

To achieve these, there must be a strong political will at all levels of government when introducing any new development initiative (Ezenwa & Brooks, 2014).

Phase 3 collated security practices from research sites in the two earlier phases to develop a secure agile software process model. The practice-based model developed was preliminarily validated using an expert focus group with 5 practitioners in Nigeria. The model validation led to proposing a new practice due to an observed lack of collaborative ceremonies to disseminate awareness of security standards and legislations and hence non-adherence.

Phase 4 evaluated the secure agile software development process model by applying it in an organisation in Nigeria using workshops, interviews, and focus group sessions. The organisation adopted the security retrospective and implemented it in three iterations of a software product been developed. The implementation of the security retrospective led to the introduction of the security champion role at HealthCo1 to further improve the security of their software development process.

The answers to the RQs formulated have been answered in chapter 8, reflecting on findings from the 4 phases of this thesis. The 3-main contributions of this thesis are: a novel taxonomy of agile security practices adopted by organisation in secure software development process, the developed GT of security challenges confronting agile practitioners in Nigeria, and the development of a novel process model for secure agile software development. This study also evaluated the proposed model in an organization, providing practitioners with insights about security practices implementation and its impact on software development process.

This thesis concludes that despite the growing rate of cybersecurity attacks on software applications, the model developed in this study can guide organisations adopt practices to secure their development process.

9.2 Conclusions

In recent years, cybersecurity has been an important issue irrespective of the software development method been adopted. Security practices are being adopted by agile practitioners, however, there are still paucity of empirical evidence on its integration into the development process. This research developed a practised-based process model for secure agile software development based on practitioner interviews. It concludes that despite the growing rate of cybersecurity attacks on software applications, the proposed model has guided the research participating organisation improve their cybersecurity culture. Thus, this research which investigated the intersection between security issues and agile methods is timely. Research findings showed the need for companies to better integrate security practices during normal agile software development processes.

In conclusions, there are lessons learnt arising from the conduct of this study for researchers to consider in the future. First, contacting practitioners through snowball sampling seem to have worked well to a certain extent, however, considering the research phenomena of secure software development, many were sceptical to provide in-dept information of their business processes. In the future, researchers are advised to focus more efforts at getting the buy-in of company managements on the value their studies offer them which would make them interested to engage. For instance, in this research, getting the buy-in of the CTO at ESSco1 unlocked several interview participants. The CTO ensured seven practitioners from his company participated fully in the data collection process. This gave the researcher an in-dept understanding of their company processes. Whereas, at average the researcher manages to get only 2-3 practitioners from the same company who agree to participate in the research when contacting them individually. Second, the qualitative approach chosen for this research has proven to be the most appropriate strategy. This is because writing a good survey questionnaire at the beginning of this study would have been very difficult due to the researcher's lack of knowledge of the phenomenon been investigated.

9.3 PhD Journey Reflection

This section reflects on different aspects of the PhD research journey including some personal challenges encountered. Working in a University, acquiring a doctorate degree was something I had looked forward to since completing my M Sc. I had applied for many scholarships in countries like Malaysia and China, where I was accepted but most of them only covered tuition fees and not stipend. I had wanted a scholarship that will cover both tuition and stipend so I can focus on my research. When the Petroleum Technology Development Fund (PTDF) overseas scholarship was advertised for 2018/2019, I applied and was shortlisted but unfortunately my application wasn't successful. In the following year, 2019/2020, I applied again and this time I was among the two candidates selected from Adamawa State.

At the initial stage of starting the PhD, identifying a research gap was a very herculean task. Coming from an Information Systems background, learning about agile software development was something completely new. Taking a complete shift from the initial PhD proposal submitted for admission to University of Salford and moving to a new research area was indeed difficult. With hard work through reading many papers on agile methods, dedication towards the PhD research, constant support from the supervisory team and Ofcourse prayers, a study on secure agile software development process was formulated.

Data collection was started at a very difficult time during the covid-19 disruptions when face-to-face meetings were not an option. All the interviews for the exploratory case study were done under unprecedented circumstances as practitioners either had caring responsibilities, fell ill, or faced lots of challenges with the new normal way of doing things. Thus, it was challenging setting-up meetings online as some practitioners never turned-up at the agreed time or cancelled it at the last minute. Also, learning how to conduct qualitative interviews and data analysis was an important learning experience as I had no prior knowledge of.

To keep the data collection process going, many agile practitioners and cybersecurity professionals were contacted since not all of them will agree or have the chance to participant in the research. After conducting some interviews with a few selected agile practitioners from diverse business sectors in the UK, it was discovered that research on agile methods adoption has been well-studied. However, the paucity of empirical evidence on the intersection between agile methods and cybersecurity especially from global south context was noted which resulted in the current research.

Expanding the research to practitioners in Nigeria, the interviews were a mixture of both online and face-to-face as many of the Covid-19 restrictions then have started to be relaxed. Although, some practitioners participated, others mentioned work pressure and other commitments as reasons for non-participation. Contacting participants through LinkedIn also helped as some individuals responded promptly. I also used LinkedIn to contact company CEOs, thankfully some showed interest in the research and directed their staff to participate. I had the privilege of attending some webinars and workshops organised by the Cyber Resilience Centre for the North-West England. I used the opportunity to network with some industry leaders and practitioners who eventually became my research interviewees. Also, I used two sampling techniques: snowballing at the initial stage and intensity sampling adopted at the later stages of the data collection.

The data analysis process at the initial stage was tedious and time consuming since manual methods was used at the beginning. After collecting some quality data and meticulously analysing it, the issue of the research contribution was a big challenge. My first two attempts at publishing conference papers were rejected owing to not highting clear research contribution(s). I had to pause data collection and analysis and do a critical comparison of the data collected with the existing literature. Only then, on the third attempt the research findings were accepted at EMCIS 2021 conference.

The writing-up phase was also not easy as I thought since I have published a couple of papers. However, even with having published some parts of the thesis, updating the literature and rearranging sections copied from published work was time-consuming. In other cases, I had to extend some sections as contents of published papers alone were too short for thesis chapters. However, having already some good materials from my interim assessment, internal evaluation meant I didn’t have to start the writing-up phase with a blank page. Thus, I advise future PhD researchers to consider the PhD writing up as an incremental process which should start from the beginning of the research with the preliminary literature and building up with all other written works and published papers.

Finally, in the course of the PhD journey, there were also a lot of personal family struggles that I had to contend with. The loneliness of been away from family and loved ones added a mental pressure on me. The covid-19 lockdown compounded the problems as physical socialization was almost impossible.

9.4 Future Work

For future research, it would be interesting to extend the study on secure agile software development by collecting data across Sub-Saharan African countries. The study would build on the very few earlier works which empirically investigated secure agile software development implementation in organisations. The proposed future study aligns with the United Nations sustainable development goal 9: Build resilient infrastructure, promote inclusive sustainable industrialization, and foster innovations.

Further focus on the implementation of security compliance in agile software development may also lead to new insights for agile practitioners and cybersecurity professionals. Regulatory policies such as NDPR which still poses a lot of challenges in secure agile software development activities is another avenue for future investigation. The emergent theory presented in this thesis provides a foundation to study regulatory policy adherence in Nigeria. There is a need for further research to investigate whether any tension exists between the government and agile practitioners in other Sub-Saharan African countries.

Despite this study achieving its objectives, there are some limitations that should be examined by researchers in future studies. To begin with, this study did not explore the implementation of regulatory policies in other sub-Saharan countries apart from Nigeria. It would have been more interesting to investigate security compliance to regulatory policies in other countries and do a comparative analysis to determine if there are differences in terms of practitioners’ perceptions.

This investigation might provide insightful understanding of security practices implementation which can enable governments to create a more conducive environment for agile practitioners to develop better secure software systems. Interviewing a limited number of practitioners across the various software firms presented another challenge due to the small sample size. This is primarily due to practitioners being hesitant to disclose sensitive security information about their company processes to people they perceive as strangers. Additionally, the limited number of participants interviewed per company presents a significant limitation when it comes to generalizing findings to specific business sectors such as healthcare or financial services. Future research efforts could address this limitation by exploring the potential to interview a larger number of participants within a particular industrial sector, thereby facilitating the generalization of findings within that specific context.

Evaluating the proposed agile process model developed in this study at different organisations is also an area of future research. The model needs to be further validated to establish its suitability for use by agile practitioners for developing secure software. This research envisages that an in-depth validation of the model by many companies will positively influence current practices and ultimately help especially SMEs reduce incidences of cyberattacks in the development process. Future work intends to use newer approaches to product development such as Dual Track.

References

Aalvik, H., Nguyen-Duc, A., Cruzes, D. S., & Iovan, M. (2023). *Establishing a Security Champion in Agile Software Teams: A Systematic Literature Review* Future of Information and Communication Conference, (pp. 796-810). Springer, Cham.

Abdelnour-Nocera, J., & Sharp, H. (2012). Understanding conflicts in agile adoption through technological frames. *International Journal of Sociotechnology and Knowledge Development (IJSKD)*, *4*(2), 29-45.

Abdulrauf, L. A. (2021). Giving 'teeth'to the African Union towards advancing compliance with data privacy norms. *Information & Communications Technology Law*, *30*(2), 87-107.

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439.*

Abubakar, M. M., Armaya'u, Z. U., & Abubakar, M. (2022). *Personal Data and Privacy Protection Regulations: State of compliance with Nigeria Data Protection Regulations (NDPR) in Ministries, Departments, and Agencies (MDAs)*. 2022 5th Information Technology for Education and Development (ITED), (pp. 1-6). IEEE.

Adepetun, A. (2023). *Nigeria's digital economy sees $4.4 billion investment in four years*. The Guardian. Retrieved 14 June 2023 from https://guardian.ng/business-services/nigerias-digital-economy-sees-4-4-billion-investment-in-four-years/

Adolph, S., Hall, W., & Kruchten, P. (2011). Using grounded theory to study the experience of software development. *Empirical Software Engineering*, *16*(4), 487-513.

Agbali, M., Dahiru, A. A., Olufemi, G. D., Kashifu, I. A., & Vincent, O. (2020). *Data privacy and protection: The role of regulation and implications for data controllers in developing countries.* Information and Communication Technologies for Development: 16th IFIP WG 9.4 International Conference on Social Implications of Computers in Developing Countries, (pp. 205-216). Cham: Springer International Publishing.

Akhigbe, O., Amyot, D., & Richards, G. (2019). A systematic literature mapping of goal and non-goal modelling methods for legal and regulatory compliance. *Requirements engineering*, *24*, 459-481.

Akinnuwesi, B. A., Uzoka, F. M., Olabiyisi, S. O., Omidiora, E. O., & Fiddi, P. (2013). An empirical analysis of end-user participation in software development projects in a developing country context. *The Electronic Journal of Information Systems in Developing Countries*, *58*(1), 1-25.

Aldin, L., & de Cesare, S. (2011). A literature review on business process modelling: new frontiers of reusability. *Enterprise Information Systems*, *5*(3), 359-383.

Alenezi, M., Basit, H. A., Beg, M. A., & Shaukat, M. S. (2022). Synthesizing secure software development activities for linear and agile lifecycle models. *Software: Practice and Experience*. 52(6), 1426-1453.

Alshaikh, M. (2020). Developing cybersecurity culture to influence employee behavior: A practice perspective. *Computers & Security*, *98*, 102003.

Amoroso, E. (2018). Recent progress in software security. *IEEE Software*, *35*(2), 11-13.

Ardo, A., Bass, J., & Gaber, T. (2021). *An empirical investigation of agile information systems development for cybersecurity.* 18th European, Mediterranean, and Middle Eastern Conference on Information Systems (EMCIS), Dubai. (pp.567-581), Cham: Springer International Publishing.

Ardo, A., Bass, J., & Gaber, T. (2022). *Towards secure agile software development process: A practice-based model.* 48th Euromicro Conference Series on Software Engineering and Advanced Applications (SEAA), Maspalomas, Gran Canaria, Spain. (pp. 149-156). IEEE

Ardo, A. A., Bass, J. M., & Gaber, T. (2023). Implications of regulatory policy for building secure agile software in Nigeria: A grounded theory. *The Electronic Journal of Information Systems in Developing Countries*, *89*(3), e12285. https://doi.org/https://doi.org/10.1002/isd2.12285

Arkin, B., Stender, S., & McGraw, G. (2005). Software penetration testing. *IEEE Security & Privacy*, *3*(1), 84-87.

Azham, Z., Ghani, I., & Ithnin, N. (2011). *Security backlog in scrum security practices* 2011 Malaysian Conference in Software Engineering, (pp. 414-417). IEEE

Baca, D., Boldt, M., Carlsson, B., & Jacobsson, A. (2015). *A novel security-enhanced agile software development process applied in an industrial setting.* 2015 10th International Conference on Availability, Reliability and Security, (pp. 11-19). IEEE

Baca, D., & Carlsson, B. (2011). *Agile development with security engineering activities* Proceedings of the 2011 International Conference on Software and Systems Process, (pp. 149-158).

Bansal, S. K., & Jolly, A. (2014). *An encyclopedic approach for realization of security activities with agile methodologies* 2014 5th International Conference-Confluence The Next Generation Information Technology Summit (Confluence), (pp. 767-772). IEEE

Barbosa, D. A., & Sampaio, S. (2015). *Guide to the support for the enhancement of security measures in agile projects* 2015 6th Brazilian Workshop on Agile Methods (WBMA), (pp. 25-31). IEEE

Bass, J. M. (2015). How product owner teams scale agile methods to large distributed enterprises. *Empirical Software Engineering*, *20*, 1525-1557.

Bass, J. M. (2016). Artefacts and agile method tailoring in large-scale offshore software development programmes. *Information and software technology*, *75*, 1-16. https://doi.org/https://doi.org/10.1016/j.infsof.2016.03.001

Beck, K. (1999). Embracing change with extreme programming. *Computer*, *32*(10), 70-77.

Beck, K. (2000). *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional.

Beecham, S., OLeary, P., Richardson, I., Baker, S., & Noll, J. (2013). *Who are we doing global software engineering research for?* 2013 ieee 8th international conference on global software engineering, (pp. 41-50). IEEE

Bell, E., Bryman, A., & Harley, B. (2018). *Business Research Methods*. Oxford University Press.

Bell, L., Brunton-Spall, M., Smith, R., & Bird, J. (2017). *Agile Application Security: Enabling Security in a Continuous Delivery Pipeline* (1st ed.). " O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

Bera, P. (2012). Does Cognitive Overload Matter in Understanding BPMN Models? *Journal of Computer Information Systems*, *52*(4), 59-69.

Bernhart, M., Mauczka, A., & Grechenig, T. (2010). *Adopting code reviews for agile software development* 2010 Agile Conference, (pp. 44-47). IEEE

Bernsmed, K., Cruzes, D. S., Jaatun, M. G., & Iovan, M. (2022). Adopting threat modelling in agile software development projects. *Journal of systems and software*, *183*, 111090.

Bezerra, C. M. M., Sampaio, S. C., & Marinho, M. L. (2020). *Secure agile software development: Policies and practices for agile teams* International Conference on the Quality of Information and Communications Technology, (pp. 343-357). Springer International Publishing.

Bhaskar, R. (2008). *Dialectic: The Pulse of Freedom*. Routledge.

Bodden, E. (2018). *State of the systems security*. Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings, (pp. 550-551)

Bodin, N., & Golberg, H. K. B. (2021). *Software security culture in development teams: An empirical study.* (Master's Thesis, NTNU).

Boehm, B. (1986). A spiral model of software development and enhancement. SIGSOFT Softw. Eng. *Notes*, *11*(4), 14-24.

Boehm, B., & Huang, L. G. (2003). Value-based software engineering: a case study. *Computer*, *36*(3), 33-41.

Breaux, T., & Antón, A. (2008). Analyzing regulatory rules for privacy and security requirements. *IEEE Transactions on Software Engineering*, *34*(1), 5-20.

Bunt, S. (2018). Critical realism and grounded theory: Analysing the adoption outcomes for disabled children using the retroduction framework. *Qualitative Social Work*, *17*(2), 176-194.

Burrell, G., & Morgan, G. (2017). *Sociological Paradigms and Organisational Analysis: Elements of the Sociology of Corporate Life*. Routledge.

Campanelli, A. S., & Parreiras, F. S. (2015). Agile methods tailoring–A systematic literature review. *Journal of systems and software*, *110*, 85-100.

Carlsson, S. A. (2012). The Potential of Critical Realism in IS Research. In *Information Systems Theory* (pp. 281-304). Springer.

Casola, V., De Benedictis, A., Rak, M., & Villano, U. (2020). A novel Security-by-Design methodology: Modeling and assessing security by SLAs with a quantitative approach. *Journal of systems and software*, *163*, 110537.

Chaouch, S., Mejri, A., & Ghannouchi, S. A. (2019). A framework for risk management in scrum development process. *Procedia Computer Science*, *164*, 187-192.

Charmaz, K. (2000). *Grounded Theory: Objectivist and Constructivist Methods* (2nd ed.). Denzin & Y.S. Lincoln (Eds.), Handbook of Qualitative Research. Thousand Oaks, CA: Sage Publications, Inc.

Charmaz, K. (2006). *Constructing Grounded Theory: A Practical Guide Through Qualitative Analysis*. Sage.

Charmaz, K. (2008). *Constructionism and The Grounded Theory Method.* J.A. Holstein & J.F. Gubrium (Eds.). Handbook of Constructionist Research. The Guilford Press.

CLASP. (2009). *Comprehensive, lightweight application security process;*. Retrieved March, 1 from https://us-cert.cisa.gov/bsi/articles/bestpractices/requirements-engineering

Cohen, D., Lindvall, M., & Costa, P. (2004). An introduction to agile methods. *Adv. Comput.*, *62*(03), 1-66.

Cohn, M. (2010). *Succeeding with Agile: Software Development Using Scrum*. Pearson Education.

Conboy, K., Coyle, S., Wang, X., & Pikkarainen, M. (2011). People over Process: Key Challenges in Agile Development. *IEEE Software*, *28*(4), 48-57. https://doi.org/10.1109/MS.2010.132

Cope, D. G. (1969). Methods and meanings: Credibility and trustworthiness of qualitative research. *Number 1/January 2014*, *41*(1), 89-91.

Creswell, J. W., & Creswell, J. D. (2017). *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications.

Crnkovic, I. (2001). Component-based software engineering — New challenges in software development. *Software focus*, *2*(4), 127-133.

Cruzes, D. S., Dybå, T., Runeson, P., & Höst, M. (2015). Case Studies Synthesis: A Thematic, Cross-case, and Narrative Synthesis Worked Example. *Empirical Software Engineering*, *20*(6), 1634-1665.

Cruzes, D. S., Felderer, M., Oyetoyan, T. D., Gander, M., & Pekaric, I. (2017). *How is security testing done in agile teams? a cross-case analysis of four software teams.* 18th International Conference on Agile Software Development, (pp. 201-216). Springer International Publishing.

Da Veiga, A., Astakhova, L. V., Botha, A., & Herselman, M. (2020). Defining organisational information security culture—Perspectives from academia and industry. *Computers & Security*, *92*, 101713.

Dännart, S., Constante, F. M., & Beckers, K. (2019). *An assessment model for continuous security compliance in large scale agile environments: exploratory paper*. Advanced Information Systems Engineering: 31st International Conference, CAiSE 2019, Rome, Italy, June 3–7, 2019, Proceedings 31, (pp. 529-544). Springer International Publishing.

Davenport, T. H. (1993). *Process innovation: reengineering work through information technology*. Harvard Business Press.

Davison, R. M., & Martinsons, M. G. (2011). Methodological practice and policy for organisationally and socially relevant IS research: an inclusive–exclusive perspective. *Journal of Information Technology*, *26*(4), 288-293.

de Vicente Mohino, J., Bermejo Higuera, J., Bermejo Higuera, J. R., & Sicilia Montalvo, J. A. (2019). The application of a new secure software development life cycle (S-SDLC) with agile methodologies. *Electronics*, *8*(11), 1218.

Dick, B. (2007). The SAGE Handbook of Grounded Theory. In. SAGE Publications Ltd. https://doi.org/10.4135/9781848607941

Diebold, P., & Dahlem, M. (2014). *Agile practices in practice: A mapping study*. Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, (pp. 1-10).

Diefenbach, T. (2009). Are case studies more than sophisticated storytelling?: Methodological problems of qualitative empirical research mainly based on semi-structured interviews. *Quality & Quantity*, *43*(6), 875-894.

Dingsøyr, T., Dybå, T., & Moe, N. B. (2010). Agile Software Development: An Introduction and Overview. In *Agile Software Development* (pp. 1-13). Springer. https://doi.org/https://doi.org/10.1007/978-3-642-12575-1_1

Dingsøyr, T., Fægri, T. E., & Itkonen, J. (2014). *What is large in large-scale? A taxonomy of scale for agile software development* Product-Focused Software Process Improvement: 15th International Conference, PROFES 2014, Helsinki, Finland, December 10-12, 2014. Proceedings 15, (pp. 273-276). Springer International Publishing.

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *85*(6), 1213-1221.

Doss, O., Kelly, T., Stålhane, T., Haugset, B., & Dixon, M. (2017). *Integration of the 4+ 1 software safety assurance principles with scrum* Systems, Software and Services Process Improvement: 24th European Conference, EuroSPI 2017, Ostrava, Czech Republic, September 6–8, 2017, Proceedings 24, (pp. 72-82). Springer International Publishing.

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and software technology*, *50*(9-10), 833-859.

Easterbrook, S., Singer, J., Storey, M.-A., & Damian, D. (2008). Selecting Empirical Methods For Software Engineering Research. In *Guide to Advanced Empirical Software Engineering* (pp. 285-311). Springer.

Edhlund, B., & McDougall, A. (2019). *NVivo 12 Essentials*. Lulu. com.

Egere, A. (2020). Cybersecurity atlas, Nigeria. (2020). *International Journal of Computer Science Trends and Technology (IJCST)*, *8*(6), 95-100.

Ehondor, B. A., & Ogbu, S. U. (2020). Personal data protection and Facebook privacy infringements in Nigeria. *Journal of Leadership, Accountability & Ethics*, *17*(2).

Ekong, I., Chukwu, E., & Chukwu, M. (2020). COVID-19 mobile positioning data contact tracing and patient privacy regulations: exploratory search of global response strategies and the use of digital tools in Nigeria. *JMIR mHealth and uHealth*, *8*(4), e19139.

ENISA. (2022). *ENISA Threat Landscape 2022*. ENISA. Retrieved 14 June 2023 from https://www.enisa.europa.eu/publications/enisa-threat-landscape-2022

Esfahani, H. C. (2012). *Transitioning to agile: A framework for pre-adoption analysis using empirical knowledge and strategic modeling*. University of Toronto (Canada).

Ezenwa, C., & Brooks, L. (2014). Understanding the Introduction and Use of a Mobile Device-Supported Health Information System in Nigeria. *The Electronic Journal of Information Systems in Developing Countries*, *62*(1), 1-20.

Felderer, M., Büchler, M., Johns, M., Brucker, A. D., Breu, R., & Pretschner, A. (2016). Security testing: A survey. In *Advances in Computers* (Vol. 101, pp. 1-51). Elsevier.

Fitzgerald, B., Stol, K.-J., O'Sullivan, R., & O'Brien, D. (2013). Scaling agile methods to regulated environments: An industry case study. 2013 35th International Conference on Software Engineering (ICSE), (pp.863-872). IEEE

Foundation, O. (2023). *OWASP;* . Retrieved March, 1 from https://owasp.org/

Fowler, M., & Highsmith, J. (2001). The agile manifesto. *Software development*, *9*(8), 28-35.

Franqueira, V. N., Bakalova, Z., Tun, T. T., & Daneva, M. (2011). *Towards agile security risk management in RE and beyond* Workshop on Empirical Requirements Engineering (EmpiRE 2011), (pp. 33-36). IEEE

Futcher, L., & von Solms, R. (2012). *SecSDM: A usable tool to support IT undergraduate students in secure software development* Proceedings of the Sixth International Symposium on Human Aspects of Information Security & Assurance (HAISA 2012), (pp. 86-96).

Gandomani, T. J., & Nafchi, M. Z. (2015). An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach. *Journal of systems and software*, *107*, 204-219.

Glaser, B., & Strauss, A. (1967). *The Discovery of Grounded Theory.* . Aldine Publishing Company.

Glaser, B. G. (1978). *Theoretical Sensitivity: Advances in the Methodology of Grounded Theory.* The Sociology Press, Mill Valley, CA, USA.

Glaser, B. G. (1992). *Emergence vs. Forcing: Basics of Grounded Theory Analysis.* (1st ed.). Sociology Press, Mill Valley, CA, USA.

Glaser, B. G. (1998). *Doing Grounded Theory: Issues and Discussions.* Sociology Press, Mill Valley, CA, USA.

Glaser, B. G. (2001). *The Grounded Theory Perspective: Conceptualization Contrasted with Description.* Sociology Press, Mill Valley, CA, USA.

Glaser, B. G., Strauss, A. L., & Strutzel, E. (1968). The discovery of grounded theory: Strategies for qualitative research. *Nursing research*, *17*(4), 364.

Gondree, M., Peterson, Z. N., & Denning, T. (2013). Security through play. *IEEE Security & Privacy*, *11*(3), 64-67.

Gralha, C., Damian, D., Wasserman, A. I., Goulão, M., & Araújo, J. (2018). *The evolution of requirements practices in software startups*. Proceedings of the 40th International Conference on Software Engineering, (pp. 823-833).

Gregor, S. (2006). The nature of theory in information systems. *MIS quarterly*, 611-642.

Grossoehme, D. H. (2014). Overview of qualitative research. *Journal of health care chaplaincy*, *20*(3), 109-122.

Hajjdiab, H., & Taleb, A. S. (2011). Adopting agile software development: issues and challenges. *International Journal of Managing Value and Supply Chains (IJMVSC)*, *2*(3), 1-10.

Hammer, M., & Champy, J. (2009). *Reengineering the corporation: Manifesto for business revolution, a*. Zondervan.

Highsmith, J. (2013). *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. Addison-Wesley.

Hoda, R., Noble, J., & Marshall, S. (2011). The Impact of Inadequate Customer Collaboration on Self-organizing Agile Teams. *Information and software technology*, *53*(5), 521-534.

Hoda, R., Noble, J., & Marshall, S. (2012). Developing a grounded theory to explain the practices of self-organizing agile teams. *Empirical Software Engineering*, *17*(6), 609-639.

Hutchison, A. J., Johnston, L. H., & Breckon, J. D. (2010). Using QSR-NVivo to Facilitate the Development of a Grounded Theory Project: An Account of a Worked Example. *International journal of social research methodology*, *13*(4), 283-302.

Ionita, D., van der Velden, C., Ikkink, H.-J. K., Neven, E., Daneva, M., & Kuipers, M. (2019). *Towards risk-driven security requirements management in agile software development* Information Systems Engineering in Responsible Information Systems: CAiSE Forum 2019, Rome, Italy, June 3–7, 2019, Proceedings 31, (pp. 133-144). Springer International Publishing.

Iyawa, G. E., Herselman, M. E., & Coleman, A. (2016). Customer interaction in software development: A comparison of software methodologies deployed in Namibian software firms. *The Electronic Journal of Information Systems in Developing Countries*, *77*(1), 1-13.

Jabangwe, R., & Nguyen-Duc, A. (2020). SIoT framework: Towards an approach for early identification of security requirements for internet-of-things applications. *e-Informatica Software Engineering Journal*, *14*(1). https://doi.org/10.37190/e-Inf200103

Jalali, S., & Wohlin, C. (2010). *Agile practices in global software engineering-A systematic map*. 2010 5th IEEE International Conference on Global Software Engineering, (pp. 45-54). IEEE.

Jeyaraj, A., Sauter, V. L., & St, M. (2014). Validation of business process models using swimlane diagrams. *Journal of Information Technology Management*, *25*(4), 27-37.

Jøsang, A., Ødegaard, M., & Oftedal, E. (2015). *Cybersecurity through secure software development* Information Security Education Across the Curriculum: 9th IFIP WG 11.8 World Conference, WISE9, Hamburg, Germany. (pp. 53-63), Springer International Publishing.

Kanniah, S. L., & Mahrin, M. N. r. (2016). A review on factors influencing implementation of secure software development practices. *International Journal of Computer and Systems Engineering*, *10*(8), 3032-3039.

Kempster, S., & Parry, K. W. (2011). Grounded theory and leadership research: A critical realist perspective. *The leadership quarterly*, *22*(1), 106-120.

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

Khaim, R., Naz, S., Abbas, F., Iqbal, N., Hamayun, M., & Pakistan, R. (2016). A review of security integration technique in agile software development. *International Journal of Software Engineering & Applications*, *7*(3), 49-68.

Klein, H. K., & Myers, M. D. (1999). A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS quarterly*, 67-93.

Kreitz, M. (2019). Security by design in software engineering. *ACM SIGSOFT Software Engineering Notes*, *44*(3), 23-23. https://doi.org/https://doi.org/10.1145/3356773.3356798

Krishnamurthy, B., & Wills, C. E. (2009). *On the leakage of personally identifiable information via online social networks*. Proceedings of the 2nd ACM workshop on Online social networks, (pp. 7-12).

Kvale, S., & Brinkmann, S. (2009). *Interviews: Learning the craft of Qualitative Research Interviewing* (2nd ed.). Sage Publications, Inc, Thousand Oaks.

Leite, L., dos Santos, D. R., & Almeida, F. (2021). The impact of general data protection regulation on software engineering practices. *Information & Computer Security*, *30*(1), 79-96.

Leung, L. (2015). Validity, reliability, and generalizability in qualitative research. *Journal of family medicine and primary care*, *4*(3), 324.

Li, H., Yu, L., & He, W. (2019). The impact of GDPR on global technology development. *Journal of Global Information Technology Management*, *22*(1), 1-6. https://doi.org/10.1080/1097198X.2019.1569186

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic Inquiry* (1st ed.). SAGE Publications, Inc.

Lui, A. K., Ngai, E. W., & Lo, C. K. (2016). Disruptive information technology innovations and the cost of equity capital: The moderating effect of CEO incentives and institutional pressures. *Information & Management*, *53*(3), 345-354.

Macarthy, R. W., & Bass, J. M. (2020). *An empirical taxonomy of DevOps in practice* 2020 46th euromicro conference on software engineering and advanced applications (seaa), (pp. 221-228). IEEE

Machamer, P., & Silberstein, M. (2008a). *The Blackwell guide to the philosophy of science*. John Wiley & Sons.

Machamer, P., & Silberstein, M. (2008b). *The Blackwell Guide to the Philosophy of Science* (Vol. 19). John Wiley & Sons.

Maria, R. E., Rodrigues Jr, L. A., & Pinto, N. A. (2015). *ScrumS: A model for safe agile development* Proceedings of the 7th International Conference on Management of computational and collective intElligence in Digital EcoSystems, (pp. 43-47).

Masood, Z., Hoda, R., & Blincoe, K. (2020). How agile teams make self assignment work: A grounded theory study. *Empirical Software Engineering*, *25*(6), 4962-5005.

Maxwell, J. A. (2012). *Qualitative research design: An interactive approach*. Sage publications.

McGraw, G. (2006). *Software Security: Building Security In* (1st ed.). Addison-Wesley Professional Upper Saddle River, NJ, USA.

Merriam, S. B. (1988). *Case study research in education: A qualitative approach*. Jossey-Bass.

Microsoft. (2021). *Microsoft Security Engineering Portal. Security Engineering Portal*. Retrieved March, 1 from https://www.microsoft.com/en-us/securityengineering/sdl/practices

Mihelič, A., Vrhovec, S., & Hovelja, T. (2023). Agile development of secure software for small and medium-sized enterprises. *Sustainability*, *15*(1), 801.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: An Expanded Sourcebook* (2nd ed.). Sage Publications, Inc.

Mingers, J. (2004). Real-izing Information Systems: Critical Realism as an Underpinning Philosophy for Information Systems. *Information and organization*, *14*(2), 87-103.

Mohallel, A. A., & Bass, J. M. (2019). *Agile software development practices in Egypt SMEs: A grounded theory investigation* Information and Communication Technologies for Development. Strengthening Southern-Driven Cooperation as a Catalyst for ICT4D: 15th IFIP WG 9.4 International Conference on Social Implications of Computers in Developing Countries, ICT4D 2019, Dar es Salaam, Tanzania. (pp. 355-365), Cham: Springer International Publishing.

Mohan, V., & Othmane, L. B. (2016). *Secdevops: Is it a marketing buzzword?-mapping research on security in devops*. 2016 11th international conference on availability, reliability and security (ARES), (pp. 542-547). IEEE

Morgan, J. M., & Liker, J. K. (2020). *The Toyota Product Development System: Integrating People, Process, and Technology*. Productivity press.

Moyón, F., Almeida, P., Riofrío, D., Mendez, D., & Kalinowski, M. (2020). *Security compliance in agile software development: A systematic mapping study* 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), Portoroz, Slovenia. (pp. 413-420). IEEE

Moyon, F., Beckers, K., Klepper, S., Lachberger, P., & Bruegge, B. (2018). *Towards continuous security compliance in agile software development at scale*. Proceedings of the 4th International Workshop on Rapid Continuous Software Engineering, (pp. 31-34).

Moyón, F., Méndez, D., Beckers, K., & Klepper, S. (2021). *Using process models to understand security standards*. In International Conference on Current Trends in Theory and Practice of Informatics, (pp.458-471). Cham: Springer International Publishing.

Myers, M. D., & Avison, D. (2002). *Qualitative Research in Information Systems: A Reader*. Sage.

Nägele, S., Watzelt, J.-P., & Matthes, F. (2022a). Investigating the Current State of Security in Large-Scale Agile Development. International Conference on Agile Software Development,

Nägele, S., Watzelt, J.-P., & Matthes, F. (2022b). *Investigating the current state of security in large-scale agile development* Agile Processes in Software Engineering and Extreme Programming: 23rd International Conference on Agile Software Development, XP 2022, Copenhagen, Denmark. (pp. 203-219), Cham: Springer International Publishing.

NDPR. (2019). https://ndpr.nitda.gov.ng/Content/Doc/NigeriaDataProtectionRegulation.pdf

Nerur, S., & Balijepally, V. (2007). Theoretical reflections on agile development methodologies. *Communications of the ACM*, *50*(3), 79-83.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, *48*(5), 72-78.

Neumann, M. (2021). *Towards a taxonomy of agile methods: The tree of agile elements*. In 2021 9th International Conference in Software Engineering Research and Innovation (CONISOFT), (pp. 79-87). IEEE

Newton, N., Anslow, C., & Drechsler, A. (2019). *Information security in agile software development projects: A critical success factor perspective* 27th European Conference on Information Systems (ECIS), Stockholm & Uppsala, Sweden.

Nicolaysen, T., Sassoon, R., Line, M. B., & Jaatun, M. G. (2010). Agile software development: The straight and narrow path to secure software? *International Journal of Secure Software Engineering (IJSSE)*, *1*(3), 71-85.

Nigeria Bureau of Statistics. (2021). *Nigeria Bureau of Statistics: Nigerian Economy Largest in Africa (2019).* https://nigerianstat.gov.ng/

Oates, B. J. (2005). *Researching Information Systems and Computing* (1st ed.). SAGE Publications Ltd.

Ogunyemi, A., Lamas, D., & Eze, E. (2018). Exploring the state of human-centred design practice in software development companies: A cross-case analysis of three nigerian software companies. *Interacting with computers*, *30*(5), 444-467.

Oliver, C. (2012). Critical realist grounded theory: A new approach for social work research. *British Journal of Social Work*, *42*(2), 371-387.

Olsen, C., & St George, D. (2004). Cross-sectional study design and data analysis. *College entrance examination board*, *26*(03), 2006.

Oueslati, H., Rahman, M. M., & ben Othmane, L. (2015). *Literature review of the challenges of developing secure software using the agile approach* 2015 10th International Conference on Availability, Reliability and Security, (pp. 540-547). IEEE.

Oueslati, H., Rahman, M. M., ben Othmane, L., Ghani, I., & Arbain, A. F. B. (2016). Evaluation of the challenges of developing secure software using the agile approach. *International Journal of Secure Software Engineering (IJSSE)*, *7*(1), 17-37.

Oyetoyan, T. D., Cruzes, D. S., & Jaatun, M. G. (2016). *An empirical study on the relationship between software security skills, usage and training needs in agile settings* 2016 11th International Conference on Availability, Reliability and Security (ARES), (pp. 548-555). IEEE

Palmer, S. R., & Felsing, M. (2001). *A practical Guide To Feature-Driven Development*. Pearson Education.

Palvia, P., Midha, V., & Pinjani, P. (2006). Research models in information systems. *Communications of the Association for Information Systems*, *17*(1), 47.

Patton, M. Q. (2002). *Qualitative Research & Evaluation Methods* (3rd ed.). SAGE Publications, Inc.

Patton, M. Q. (2014). *Qualitative Research & Evaluation Methods: Integrating Theory and Practice*. Sage publications.

Paul, M. (2013). *Official (ISC) 2 guide to the CSSLP CBK*. CRC Press.

Petersen, K., & Gencel, C. (2013). *Worldviews, Research Methods, and Their Relationship to Validity in Empirical Software Engineering Research* 2013 Joint Conference of the 23rd International

Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement, (pp. 81-89). IEEE

Philips, Z., Claxton, K., & Palmer, S. (2008). The half-life of truth: what are appropriate time horizons for research decisions? *Medical Decision Making*, *28*(3), 287-299.

Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., & Still, J. (2008). The impact of agile practices on communication in software development. *Empirical Software Engineering*, *13*(3), 303-337.

Pikkarainen, M., Salo, O., Kuusela, R., & Abrahamsson, P. (2012). Strengths and barriers behind the successful agile deployment—Insights from the three software intensive companies in Finland. *Empirical Software Engineering*, *17*(6), 675-702.

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit*. Addison-Wesley.

Poppendieck, M., & Poppendieck, T. (2009). *Leading Lean Software Development: Results Are Not The Point*. Pearson Education.

Poppendieck, M., & Poppendieck, T. D. (2007). *Implementing Lean Software Development: From Concept To Cash*. Pearson Education.

Presthus, W., Sørum, H., & Andersen, L. R. (2018). *GDPR compliance in norwegian companies* Norsk konferanse For Organisasjoners Bruk av IT (NOKOBIT), Svalbard, Norway. (pp. 1-14)

Rahy, S., & Bass, J. (2018). Information flows at inter-team boundaries in agile information systems development. In Information Systems: 15th European, Mediterranean, and Middle Eastern Conference on Information Systems, Proceedings 15 (pp. 489-502). Springer International Publishing.

Rahy, S., & Bass, J. (2021). Managing non-functional requirements in agile software development. *IET Software*. 16(1), 60-72.

Rahy, S., & Bass, J. M. (2020). *Implementation of agile methodology in developing countries: Case study in lebanon* Information and Communication Technologies for Development: 16th IFIP WG 9.4 International Conference on Social Implications of Computers in Developing Countries, Manchester, UK. (pp. 217-228). Springer International Publishing.

Rahy, S., Kreps, D., Bass, J. M., Gaber, T., & Ardo, A. (2020). *A post-colonial analysis of agile software development methods in ICT4D* Information and Communication Technologies for Development: 16th IFIP WG 9.4 International Conference on Social Implications of Computers in Developing Countries, Manchester, UK. (pp. 66-77). Springer International Publishing.

Ramias, A. J., & Rummler, R. (2009). The evolution of the effective process framework: A model for redesigning business processes. *Performance Improvement*, *48*(10), 25-32.

Reed, M. S. (2016). *The Research Impact Handbook* (2nd ed.). Fast Track Impact.

Regassa, Z., Bass, J. M., & Midekso, D. (2017). *Agile methods in ethiopia: An empirical study* 14th IFIP WG 9.4 International Conference on Social Implications of Computers in Developing Countries, ICT4D Yogyakarta, Indonesia. (pp. 367-378). Springer International Publishing.

Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2018). *Aligning security objectives with agile software development* Proceedings of the 19th International Conference on Agile Software Development: Companion, (pp. 1-9).

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.

Rindell, K., Ruohonen, J., & Hyrynsalmi, S. (2018). *Surveying secure software development practices in finland* Proceedings of the 13th International Conference on Availability, Reliability and Security, Hamburg, Germany. (pp. 1-7).

Robey, D., Welke, R., & Turk, D. (2001a). Traditional, iterative, and component-based development: A social analysis of software development paradigms. *Information Technology and Management*, *2*(1), 53-70.

Robey, D., Welke, R., & Turk, D. (2001b). Traditional, iterative, and component-based development: A social analysis of software development paradigms. *Information Technology and Management*, *2*, 53-70.

Robson, C., & McCartan, K. (2016). *Real World Research: A Resource for Users of Social Research Methods in Applied Settings*. Wiley.

Royce, W. W. (1987). *Managing the development of large software systems: concepts and techniques* Proceedings of the 9th International Conference on Software Engineering, (pp. 328-338)

Rubin, K. S. (2012). *Essential Scrum: A Practical Guide To The Most Popular Agile Process*. Addison-Wesley.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*, 131-164.

Sadowski, C., Söderberg, E., Church, L., Sipko, M., & Bacchelli, A. (2018). *Modern code review: A case study at google.* In Proceedings of the 40th international Conference on Software Engineering: Software Engineering in Practice, (pp. 181-190)

SAFECode. (2009). *Fundamental Practices for Secure Software Development.* . Retrieved March, 1 from https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf

Saher, N., Baharom, F., & Ghazali, O. (2017). *Requirement change taxonomy and categorization in agile software development.* 2017 6th International Conference on Electrical Engineering and Informatics (ICEEI), (pp. 1-6). IEEE

Sánchez-Gordón, M., & Colomo-Palacios, R. (2020). *Security as culture: A systematic literature review of DevSecOps* Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, (pp. 266-269).

Saunders, M., Lewis, P., & Thornhill, A. (2007). *Research Methods for Business Students* (4th edition ed.). Pearson Education Limited, England.

Saunders, M., Lewis, P., & Thornhill, A. (2009). *Research Methods for Business Students*. Pearson Education.

Schwaber, K. (2004). *Agile Project Management with Scrum*. Microsoft press.

Seaman, C. B. (1999). Qualitative Methods in Empirical Studies of Software Engineering. *IEEE Transactions on Software Engineering*, *25*(4), 557-572.

Sebega, Y., & Mnkandla, E. (2017). Exploring issues in agile requirements engineering in the South African software industry. *The Electronic Journal of Information Systems in Developing Countries*, *81*(1), 1-18.

Sekaran, U., & Bougie, R. (2016). *Research methods for business: A skill building approach*. john wiley & sons.

Sharma, A., & Bawa, R. (2020). Identification and integration of security activities for secure agile development. *International Journal of Information Technology*, 1-14.

Sharp, H., Robinson, H., & Petre, M. (2009). The role of physical artefacts in agile software development: Two complementary perspectives. *Interacting with computers*, *21*(1-2), 108-116.

Shastri, Y., Hoda, R., & Amor, R. (2021). The role of the project manager in agile software development projects. *Journal of systems and software*, *173*, 110871. https://doi.org/https://doi.org/10.1016/j.jss.2020.110871

Siavvas, M., Tsoukalas, D., Jankovic, M., Kehagias, D., & Tzovaras, D. (2020). Technical debt as an indicator of software security risk: A machine learning approach for software development enterprises. *Enterprise Information Systems*, 1-43.

Sidky, A., Arthur, J., & Bohner, S. (2007). A disciplined approach to adopting agile practices: The agile adoption framework. *Innovations in systems and software engineering*, *3*(3), 203-216.

Siiskonen, T., Sars, C., Vah Sipila, A., & Pietikain, A. (2014). Generic security user stories. *Handbook of the Secure Agile Software Development Life Cycle; Pietikinen, P., Rning, J., Eds*, 9-14.

Sindre, G., & Opdahl, A. L. (2005). Eliciting security requirements with misuse cases. *Requirements engineering*, *10*(1), 34-44.

Siponen, M., Baskerville, R., & Kuivalainen, T. (2005). *Integrating security into agile development methods.* Proceedings of the 38th Annual Hawaii International Conference on System Sciences, Big Island, HI, USA.

Sodanil, M., Quirchmayr, G., Porrawatpreyakorn, N., & Tjoa, A. M. (2015). *A knowledge transfer framework for secure coding practices* 2015 12th International Joint Conference on Computer Science and Software Engineering (JCSSE), (pp. 120-125). IEEE

Soriyan, H. A., Mursu, A. S., Akinde, A. D., & Korpela, M. J. (2001). Information systems development in Nigerian software companies: Research methodology and assessment from the healthcare sector's perspective. *The Electronic Journal of Information Systems in Developing Countries*, *5*(1), 1-18.

Sowunmi, O. Y., Misra, S., Fernandez-Sanz, L., Crawford, B., & Soto, R. (2016). An empirical evaluation of software quality assurance practices and challenges in a developing country: A comparison of Nigeria and Turkey. *SpringerPlus*, *5*(1), 1-13.

Srinivasan, J., & Lundqvist, K. (2010). Agile in India: Challenges and lessons learned. Proceedings of the 3rd India software engineering conference, (pp. 125-130).

Stallings, W., Brown, L., Bauer, M. D., & Bhattacharjee, A. K. (2012). *Computer security: principles and practice*. Pearson Education Upper Saddle River, NJ, USA.

Stapleton, J. (1997). *DSDM: Dynamic Systems Development Method: The Method in Practice*. Cambridge University Press.

Strauss, A., & Corbin, J. (1990). *Basics of Qualitative Research*. Sage Publications.

Stray, V., Sjøberg, D. I., & Dybå, T. (2016). The daily stand-up meeting: A grounded theory study. *Journal of systems and software*, *114*, 101-124.

Sulayman, M., Urquhart, C., Mendes, E., & Seidel, S. (2012). Software process improvement success factors for small and medium Web companies: A qualitative study. *Information and software technology*, *54*(5), 479-500.

Taylor, P. S., Greer, D., Sage, P., Coleman, G., McDaid, K., & Keenan, F. (2006). *Do agile GSD experience reports help the practitioner?* Proceedings of the 2006 International Workshop on Global Software Development for the Practitioner, (pp.87-93).

Terpstra, E., Daneva, M., & Wang, C. (2017). *Agile practitioners' understanding of security requirements: Insights from a grounded theory analysis* 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW), (pp. 439-442). IEEE

Tobin, G. A., & Begley, C. M. (2004). Methodological rigour within a qualitative framework. *Journal of advanced nursing*, *48*(4), 388-396.

Tomanek, M., & Klima, T. (2015). Penetration testing in agile software development projects. *International Journal On Cryptography And Information Security*, *5*(1), 1-7.

Tøndel, I. A., Cruzes, D. S., Jaatun, M. G., & Rindell, K. (2019). *The security intention meeting series as a way to increase visibility of software security decisions in agile development projects* Proceedings of the 14th International Conference on Availability, Reliability and Security, CA, Canterbury, United Kingdom. (pp. 1-8).

Tøndel, I. A., Cruzes, D. S., Jaatun, M. G., & Sindre, G. (2022). Influencing the security prioritisation of an agile software development project. *Computers & Security*, *118*, 102744.

Tøndel, I. A., Jaatun, M. G., Cruzes, D. S., & Moe, N. B. (2017). Risk centric activities in secure software development in public organisations. *International Journal of Secure Software Engineering (IJSSE)*, *8*(4), 1-30.

Tøndel, I. A., Jaatun, M. G., Cruzes, D. S., & Williams, L. (2019). Collaborative security risk estimation in agile software development. *Information & Computer Security*, *27*(4), 508-535.

Urquhart, C., & Fernández, W. (2013). Using grounded theory method in information systems: The researcher as blank slate and other myths. *Journal of Information Technology*, *28*, 224-236.

Urquhart, C., Lehmann, H., & Myers, M. D. (2010). Putting the 'theory'back into grounded theory: guidelines for grounded theory studies in information systems. *Information Systems Journal*, *20*(4), 357-381.

Usman, M., Börstler, J., & Petersen, K. (2017). An effort estimation taxonomy for agile software development. *International Journal of Software Engineering and Knowledge Engineering*, *27*(04), 641-674.

Usman, M., Britto, R., Börstler, J., & Mendes, E. (2017). Taxonomies in software engineering: A systematic mapping study and a revised taxonomy development method. *Information and software technology*, *85*, 43-59.

Usman, M., Felderer, M., Unterkalmsteiner, M., Klotins, E., Mendez, D., & Alégroth, E. (2020). *Compliance requirements in large-scale software development: An industrial case study* Product-Focused Software Process Improvement: 21st International Conference, PROFES 2020, Turin, Italy, November 25–27, 2020, Proceedings 21, (pp. 385-401). Springer International Publishing.

Vaivio, J. (2012). Interviews –learning the craft of qualitative research interviewing. *European Accounting Review*, *21*(1), 186-189. https://doi.org/10.1080/09638180.2012.675165

Valdés-Rodríguez, Y., Hochstetter-Diez, J., Díaz-Arancibia, J., & Cadena-Martínez, R. (2023). Towards the integration of security practices in agile software development: A systematic mapping review. *Applied Sciences*, *13*(7), 4578.

van der Heijden, A., Broasca, C., & Serebrenik, A. (2018). *An empirical perspective on security challenges in large-scale agile software development* Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, (pp. 1-4).

Venson, E., Guo, X., Yan, Z., & Boehm, B. (2019). *Costing secure software development: A systematic mapping study* Proceedings of the 14th International Conference on Availability, Reliability and Security, (pp. 1-11).

Villamizar, H., Kalinowski, M., Garcia, A., & Mendez, D. (2020). An efficient approach for reviewing security-related aspects in agile requirements specifications of web applications. *Requirements engineering*, *25*, 439-468.

Villamizar, H., Kalinowski, M., Viana, M., & Fernández, D. M. (2018). *A systematic mapping study on security in agile requirements engineering* 2018 44th Euromicro conference on software engineering and advanced applications (SEAA), (pp. 454-461). IEEE

Von Solms, R., & Van Niekerk, J. (2013). From information security to cyber security. *Computers & Security*, *38*, 97-102.

Wang, X., Conboy, K., & Pikkarainen, M. (2012). Assimilation of agile practices in use. *Information Systems Journal*, *22*(6), 435-455.

Williams, C. (2007). Research methods. *Journal of Business & Economics Research (JBER)*, *5*(3).

Williams, L. (2010). Agile software development methodologies and practices. In *Advances in computers* (Vol. 80, pp. 1-44). Elsevier.

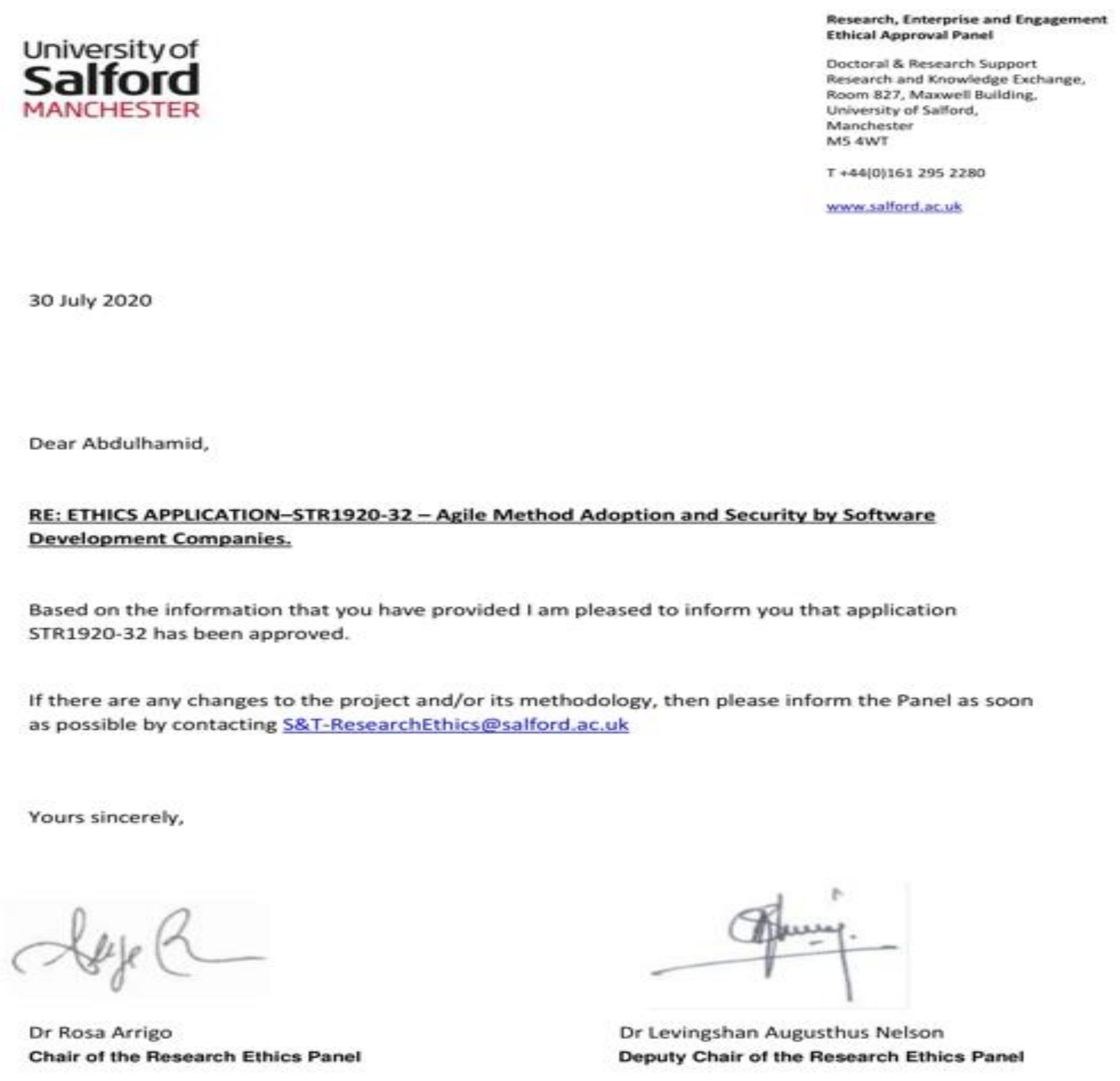
Williams, L., Meneely, A., & Shipley, G. (2010). Protection poker: The new software security" game". *IEEE Security & Privacy*, *8*(3), 14-20.

Zerlang, J. (2017). GDPR: A milestone in convergence for cyber-security and compliance. *Network Security*, *2017*(6), 8-11.

Zhang, S., Caragea, D., & Ou, X. (2011). *An empirical study on using the national vulnerability database to predict software vulnerabilities.* 22nd International conference on database and expert systems applications (DEXA), (pp. 217-231).

Appendix

A - Ethical Approval Form



University of Salford MANCHESTER

Research, Enterprise and Engagement
Ethical Approval Panel

Doctoral & Research Support
Research and Knowledge Exchange,
Room 827, Maxwell Building,
University of Salford,
Manchester
M5 4WT

T +44(0)161 295 2280

www.salford.ac.uk

30 July 2020

Dear Abdulhamid,

RE: ETHICS APPLICATION–STR1920-32 – Agile Method Adoption and Security by Software Development Companies.

Based on the information that you have provided I am pleased to inform you that application STR1920-32 has been approved.

If there are any changes to the project and/or its methodology, then please inform the Panel as soon as possible by contacting S&T-ResearchEthics@salford.ac.uk

Yours sincerely,

[illegible] [illegible]

Dr Rosa Arrigo
Chair of the Research Ethics Panel

Dr Levingshan Augusthus Nelson
Deputy Chair of the Research Ethics Panel

B - Interview Guide

University of Salford MANCHESTER

Agile Security Practices Implementation in Organisations

Initial Interview Guide

August 2020

Abdulhamid Aliyu Ardo, Julian Bass & Tarek Gaber

Interview Guide

Introduction

The aim of this research is to help practitioners make better use of agile methods to create secure agile process to be adopted in the software industry. The study will examine practitioner perceptions of the impact of security on software quality and productivity in some selected companies. The study will further explore how security influences the development of high-quality agile software.

I would like to ask you some questions about your experience of security engineering while using agile software development methods. Additionally, I would like to know your perception of security integration into normal agile processes. I will be interviewing a cross section of practitioners ranging from technical (software developers, security specialists, systems analyst, business analyst) to non-technical (decision maker/managers, project managers) to understand different business context.

PART I – Agile Methods Adoption

1. Please describe your experience of using agile software development methods in your current job role?
2. Can you describe the roles that make up your project team?
3. What strategies are adopted by your team to meet customer requirements?
    a. (Probing) Do you experience the problem of tasks coming along outside the normal sprint planning process?
4. How do you communicate and coordinate job tasks with your team?
    a. Within your organization
    b. Across multiple organizations, countries, and cultures
5. (Open-Ended) What problems have you faced working with your team members in your current job?
    a. (Probing) How have you been able to overcome the challenges?
    b. (Probing) What lessons were learnt?

PART II – Security Practices in Agile Software Development

1. Please describe how you do security requirements gathering?
2. How do you consider security issues during the design stage of software development?
3. Please describe the software security testing methods adopted in your organization?
4. How do you build security into deployment processes?
5. (Open-Ended) Does your company face any security resource constraints during agile software development activities?

PART III - Personal Details

1. What is your educational level and professional background?

2. What is your current job title?

3. How many people are in the current projects you are handling?

4. How long have you been working at the current company?

5. How long have you been working in the software industry?

University of Salford MANCHESTER

Implication of Regulatory Policy for Building Secure Agile Software in Nigeria

Reviewed Interview Guide

April 2021

Abdulhamid Aliyu Ardo, Julian Bass & Tarek Gaber

Interview Guide

Introduction

The aim of this research is to help practitioners make better use of agile methods to create secure process for developing software. The study will examine practitioner perceptions of the impact of NDPR policy and its challenges on the security of agile software in some selected companies in Nigeria. The study will further explore how security influences the development of high-quality agile software.

I would like to ask you some questions about your security practices while developing agile software. Additionally, I would like to know your perception of the NDPR policy on software development activities. I will be interviewing a cross section of practitioners ranging from technical (software developers, security specialists, systems analyst, business analyst) to non-technical (decision maker/managers, project managers) to understand different business context.

PART I – Security Practices in Agile Software Development

1. Please can you describe your job role?

2. Please describe how you do security requirements gathering?

a. (Probing) How are security requirements discussed and disseminated within your organization?

b. How is security involved at the requirements gathering stage?

3. How do you consider security issues during the design stage of software development? (Probing) – Software, Hardware, Network, Storage?

4. Please describe the collaboration practices that you use in your organization to handle security?

a. (Probing) Who is responsible for handling security audits in your current company?

5. Please describe the software security testing methods adopted in your organization?

a. (Probing) How do you manage security vulnerability testing activities within your organization?

b. Does your organization use security tools for the following:

i. Vulnerability checks

ii. Software testing

6. How do you build security into deployment processes?

    a. (Probing) How does security impacts the CI/CD pipeline processes?

    b. Do Security Deployment tools fit into your deployment processes?

7. Would you say your organization has a security culture?

    a. (Probing) Please can you describe how the security culture is built or developed in your organization?

8. (Open-Ended) Does your company face any security challenges during software development?

    a. (Probing) How often do you give security trainings to your staff?

PART II – NDPR Adherence During Secure Software Development

1. What strategies has your organization adopted to ensure adherence to the NDPR act for developing secure agile software?

    a. (Probing) How do you ensure data collected for developing secure agile software are not identifiable as prescribed in sections 1.2 – 1.3 of the NDPR Act?

2. Would you describe the NDPR Act as being helpful in your agile software development activities? In what ways?

    a. (Probing) What aspects the NDPR Act do you find most important for securing your software?

3. What techniques do you use for identifying potential gaps and weaknesses in your organization as prescribed in article 4.1(5) of the NDPR Act?
4. Please describe how you prevent against breaches as prescribed in Article 2.6 (Information Security Architecture Improvement) of the NDPR Act?

    a. (Probing) Do you have a mechanism of notifying NITDA of breaches within 72 hours of becoming aware of it?

5. Do you encounter any challenges while adhering to the NDPR Act during software development?

Closing Question - Is there anything else you think is relevant that has been missed?

PART III - Personal Details

1. What is your educational level and professional background?
2. What is your current job title?
3. How many people are in the current projects you are handling?
4. How long have you been working at the current company?
5. How long have you been working in the software industry?

Practice-Based Model Validation - Focus Group Interview Questions

1.Does the model represent the security practices you are using during agile software development?

[Probing] (a.) If yes, what other practices do you use that is not included in the model?

(b.) If no, what practices are you using to create secure agile software?

2.Do you think the model looked logical and understandable?

3.Is there anything you think I should have added in my model?

4.Is there anything you think shouldn’t have been added onto the model?

C - Consent Form

The School of Science, Engineering & Environment
Newton Building
University of Salford
Manchester
M5 4WT
United Kingdom

Email: a.a.ardo@edu.salford.ac.uk

Title of study: Secure Agile Software Development Process

Name: Abdulhamid Aliyu Ardo

Please indicate your consent and Sign below.

Please tick both if you agree:

- ❑ I have read the Participant Information Sheet and Understood the purpose of the research
- ❑ I freely agree to participate in this interview as described and understand that I am free to withdraw from the research at any time

Please tick one:

- ❑ I consent to being referred to by my name in the final PhD thesis and any other publications relating to the PhD Thesis; or
- ❑ I consent to being referred to by the name of my workplace and job title in the PhD Thesis and any other publications relating to the PhD Thesis; or
- ❑ I consent to the information I provide being used for the purposes of the aforementioned PhD Thesis only if it is fully de-identified (anonymized)

Optional:

- ❑ I would like to receive a copy of the PhD Thesis when it becomes ready and publicly available

Name of Participant (please print):____

Signature of Participant:____

Date:____

Declaration by Researcher:

I have given a verbal explanation of the interview; its study activities and risks and I believe that the research participant has understood that explanation. I have also explained the EU General Data Protection Regulation (GDPR), United Kingdom Data Protection Act 2018 and the Nigeria Data Protection Regulation 2019 to the research respondent.

Researcher Signature:

Date: **11th January 2022**

D - Participant Information Sheet

Secure Agile Software Development Process

Participant Information Sheet

August 2020

Abdulhamid Aliyu Ardo

1. Research Project Title

Secure Agile Software Development Process

2. Invitation

You are being invited to take part in this research project. Before you decide to do so, it is important that you understand why the research is being conducted and what it will involve. Please take the time to read the following information carefully and discuss it with others if you wish. Ask us if there is anything not clear or if you would like more information.

3. What is the Project's Purpose?

The purpose of the project is to help practitioners make better use of security practices to create more secured applications by developing a secure-by-design agile development process to be adopted in the software industry.

4. Why Have I been chosen?

You have been chosen because you are a practitioner with specialist skills and knowledge in agile software development.

5. Do I have to take part?

It is up to you to decide whether to take part or not. If you do decide to take part, you will be able to keep a copy of this information sheet and you should indicate your agreement on the consent form.

6. What will happen to me if I take part?

You will be asked to take part in an interview session, either face-to-face or via video teleconferencing applications (Skype, Zoom, Teams etc). The average interview duration will 40 minutes.

7. What do I have to do?

You will be expected to answer some semi-structured open-ended interview questions. The questions will focus on understanding your experience and perception of security practices adopted during software development. Also, the interview will seek to examine your views on encouraging and inhibiting factors for secure agile software development in Nigeria

8. What are the possible disadvantages of taking part?

Participating in the research is not anticipated to cause you any disadvantages or discomfort. The potential physical and/or psychological harm or distress will be the same as any experienced in everyday life.

9. What are the possible benefits of taking part?

Whilst there are no immediate benefits for research participants, it is hoped that this work will have a beneficial impact on the development of secured agile software applications. Research findings will be shared with those participants who request it on the consent form, in order to inform your professional work.

10. Will my taking part in this project be kept confidential?

You will have the choice for your responses to be kept confidential. You will also have the choice to allow us to use your name, job title or affiliation by granting such permission on the consent form.

11. Will I be recorded, and how will the recorded media be used?

Interview participants will be either be audio or video recorded during the interview. Interview recordings will be kept confidential. The recording will be transcribed verbatim into a script describing your words. These words will be analysed and maybe quoted in the final PhD thesis and any other reports or publications relating to the research. The quotes will be kept anonymous or attributed to you or your affiliation depending on the permission you grant on the consent form.

12. What type of information will be sought from me and why is the collection of this information relevant for achieving the research project's objectives?

You will be asked about your experience and perception of security engineering and agile methods adoption in the development of high-quality and secure applications in the software industry. Your views and experience are just what the project is interested in exploring.

13. What will happen to the results of the research project?

Results of the research will be published. You will not be identified in any report or publication, unless you have given us permission to do so on the consent form. Your institution will not be identified in any report or publication, unless you have given us permission to do so on the consent form. If you wish to be given a copy of any reports resulting from the research, please ask us to put you on our circulation list.

14. Who is organising and funding the research?

This project is being conducted by Abdulhamid Aliyu Ardo with funding from Petroleum Technology Development Fund (PTDF), Nigeria

15. Who has ethically reviewed the project?

This project with application number [STR1920-32] has been approved by the University of Salford Research Ethics Panel.

16.Contacts for further information

Abdulhamid Aliyu Ardo
PhD Candidate/ Project Director
School of Science, Engineering & Environment
Newton Building
University of Salford
The Crescent
Manchester, M5 4WT

Telephone (mobile) +44 (0) 781 810 9439
email: a.a.ardo@edu.salford.ac.uk

Professor Julian M. Bass CEng FBCS CITP SFHEA
Head Computer Science & Software Engineering
School of Science, Engineering & Environment
Room 218, University of Salford
Newton Building
The Crescent
Manchester, M5 4WT

Telephone (external) +44 (0) 161 295 2883
Telephone (internal) ext. 52883
email: j.bass@salford.ac.uk

Dr Tarek Gaber
Senior Lecturer of Computer Science & Software Engineering
School of Science, Engineering & Environment
Room 205, University of Salford
Newton Building
The Crescent
Manchester, M5 4WT

Telephone (external) +44 (0)161 295 5037
email: t.m.a.gaber@salford.ac.uk

E - Sample Interview Transcript

File: 2020_UK_Agile Security_6.0

Duration: 00:36:43

Date: 01/11/2020

Interviewer: I would like to ask you some questions about your experience of security engineering while using agile software development methods. The aim of this research is to help practitioners make better use of agile methods to create secure process for developing software. Particularly, I am interested in understanding your perceptions of the impact of security on software quality and productivity of your company, so that we can learn for the future. As explained in the participant information sheet, you were chosen as a participant for this research because you are an agile practitioner. Also, I want to record your responses and assure you once more that the information provided will be kept confidential in accordance with the permission granted on the consent form. Can I switch on the recorder now?

START RECORDING

Thank you so much Richard for your time. As I earlier explained, this research is basically about software development security and its impact on the process.

Interviewer: To start with, please can you describe your job role?

Richard: Ahhh… Yeah certainly, I am the Vice President of Cyber Operations Security for CGI in the UK. So underneath me seats are Security Operations Centre; so that's an intelligence led proactive monitoring service. It covers all areas; so we got freight intelligence monitoring, malware revise engineering, vulnerability management and all the usual things you would expect to have with inside sock. Ahhhhh… the Penetration testing and Red team. So we provide routine feasibility both physical and logical as well as traditional penetration testing for companies. And the last area that have under me is our Research lab, so we have a commercial licensed valuation facility that provides analysis of products under the UK government C tests and CPK schemes.

Interviewer: So currently what projects are you working on?

Richard: Well, we have multiple projects that are going on which are primarily around protective monitoring and securing of customers infrastructure so that they can make the right decisions based on the perceived threats and risks profiles for the organization and the threat actors that are targeting. The sector or the organizations themselves. So, the projects at the moment are very much around the operational security of organizations.

Interviewer: Please can you describe how security issues influences the way you manage projects?

Richard: Hahhahhaaa… Well my entire life is security. So, everything I do is influenced by security. The CGI is a bit different; we have had a real push over the last sort of seven or eight years. The mantra within CGI is that Cyber security is part of everything we do. That road goes from a policy perspective right the way to the operations. So the GRC (Governance Risk & Compliance) understanding what the risk landscape is, understanding what the threats are to the organization then coming up with a secure design to make sure you mitigate those risks and obviously the monitoring to make sure those technical and logical controls put in place are actually maintaining and mitigating the risk that were identified that cycle that goes all the way round. For me, I think that mantra is one of things that needs to be driven home in all SDLCs and especially with inside agile. One of the concerns I always had with agile when it first started coming out was, agile was very much part of world class sort of 1980s programming. When you sit down and got a really good idea and come up with something and then they wouldn't be the governance structure around it to say what are the requirements? What are things you are trying to achieve? Is security taken care of in that? Obviously, that has changed a lot now with DevSecOps. I think the DevSecOps is actually wrong, it should be SecDevOps because security should be first rather than just an after though in the development. So, thinking about secure coding practices, making sure you understand from a requirements perspective the security requirements from there as well. So, its not just validating the code against Black Duck or Syno Cube or some of those common programming states but actually understanding the requirements for security like functional and non-functional requirements. Ahhhh… The success I think of assuring that Cyber security is part of all the decisions that are been made. Cyber security is not seen as a hurdle to overcome but actually seen as an enabler to assure what is been delivered.

Interview: When you are planning for a new software project, how do you take into account security features?

Richard: Ahhhh… So, security been front and foremost in my mind all the time. When we are engaging in new projects is looking at what the baseline security requirements are. So, we have developed baseline security standards which is applied against all projects. Whether it is software development or system integration, so by adding those functional and non-functional security requirements along with all the other functional and non-functional requirements means that security is then considered right in the beginning rather than traditionally where it was which is at the end. Then suddenly you go like a week before going life..ohh we forgot about security and actually make sure you are happy before we go live. I have had many many years of where we've been involved in the last months or so before a product goes live we do a Pen test on it and it fails the Pen test and they get really upset because it now gets delay for their project. Going live is sometimes actually impacted quite some major timelines so I can remember I did one probably like ten years ago now where they were looking to launch the website for Christmas on the 23rd December. I did the Pen test and put a month delay in their lurch. So they were not happy with the development team so I had to put it across in such a way that it's actually a good thing you are not going live, these are all the issues and next time maybe you want to ensure that Cyber security is part of, well it was IT security then. People weren't talking about Cyber security 10 years ago.

Interviewer: Please describe how you do security requirements gathering?

Richard: Security requirements gathering is done in a number of ways. Number one is from an intelligence perspective; so looking who the threat actors are operating within the industry, who are likely to be targeting the organization. What the thing is the organization wants to deliver and how they are looking to deliver it because out of that will drive the risk and threat profiles for that customer or their product. Ahhhhhh… and depending on where it is, if it is an internet facing product that got higher risk than something that is just used internally. So again, is making sure that you understand the threat landscape for that to determine what the security controls are required to mitigate the risks that are likely to be identified during that risk assessment.

Interviewer: How are security requirements discussed and disseminated within your organization?

Richard: Ahhh…. So obviously we have the baseline security standards everyone knows that already …. Ahhhhh… then it's very much done mostly on a sort of consultancy engagements basis I suppose really as part of the development of the software. There is engagement with security consultant working with the development team to make sure that the security controls are met and obviously penetration testing is then taken care of as at when is available to be pen tested. Also, we use code nursing tools like Black Duck or Syno cube tool to actually go through and validate the code from a security perspective as well, so its actually embedded as part of that SDLC.

Interviewer: Who else is involve in security conversation during the requirement gathering stage?

Richard: All the stakeholders are required, what I mean by that is you got the Business, the development leads, and you also got the security consultants in there. So, security is not something done in isolation, it has to be done with all the key stakeholders because there will be decision that are made about security that are business decisions because there are some financial impact. Therefore, the commercial team will potentially have to be involved as part of that decision making as well. Also, the legal team as there might be legal requirements as well that needs to be met. So, its engaging the right stakeholders at the right time and it's not just a security talking to a techy and saying these are the requirements. It needs to be understood across the board.

Interview: How do you consider security issues during the design stage?

Richard: Ahhhhhh…. Well the security issues really come out from the functional and non-functional requirements when things are starting to be developed because there may be some functional or non-functional requirements which cannot be met. So, we then think about the additional mitigation that needs to be put in place to mitigate the risk of that control not been either fully met or even partially met. So that constant risk analysis and re-baselining has to take place during the software development life cycle.

Interviewer: How do you select or develop appropriate secure software design methods?

Richard: Ahhhhh.. so again there should be no … ahhh.. from a security perspective no real difference between whether you are doing agile development or a standard waterfall development. In some cases, there are…. If there is a need to get to market quickly, then the agile methodologies work well. If it is a sort of very rigorous and well-defined outcome that you are looking to achieve then a waterfall methodology works fine. So really its not a security decision as to whether a waterfall or an agile methodology is used, it's very much driven by the business's requirement and how quickly something needs to go to market in the best possible way to take it there. As I said if it's a very well-defined thing that is going to developed, then waterfall works fine. If its something that is been developed as you go then obviously that is where the agile methodologies work well in my mind.

Interviewer: Do you have an organizational security policy?

Richard: Yes we do. As I say we do have the baseline security standards that is within CGI that covers all the requirements that needs to be considered during design, implementation and run.

Interviewer: How do you ensure adherence to security technical strategies?

Richard: That is done through assurance process. So, security is part of the sign-off as things move through, so you have to have the security sign-off as well as UAT sign-off, user sign-off and business sign-off. By embedding security as part of the assurance process then you know that security controls are been met.

Interviewer: Please can you describe the collaboration practices that you use in your organization in handling security?

Richard: Ahhhhhh….. So, the collaboration practices really are a sort of the embedding of security within the team. So, there is collaboration going on all the times not just something that happens in isolation, so it has to be part of the software teams. They have to have that engagement and ideally all the developers have been trained in secure life-cycle development as well. They understand the security requirements without security having to say this is what you need to achieve. So, education is absolutely key to help the developers understand how to do secure development.

Interviewer: Is there any individual responsible for security audits in your organization?

Richard: So, security audits yes, ahhhh… so there are two types, we have internal audits, and we have audits which are performed as part of the SDLC. So obviously security audits can be code analysis, penetration testing and then it can be a general security reviews security audit especially if you are working with cyber frameworks for like since 2005 or if you are working in US you can be against CIS. There are a number of auditing frameworks that can be used, and they are used as part of the assurance process.

Interviewer: Can you please describe the software testing methods adopted in your organization?

Richard: Ahhhh… Yes, so we have a lot of automated testing that we use. So, there is a lot of test tools. From a code analysis perspective, we use commercial tools like Black Duck or Syno Cube. Syno cube get used quite a lot because its open source. Ahhhhh….and then obviously from an automated testing perspective we are using the usual likes of Jenkins and Cucumber to standard tooling that you would use for automated testing and obviously from a Pen testing perspective that's very manual process but we do automated testing of vulnerability assessments against the codes as well to tell us whether libraries are right or whether there are vulnerabilities in the library or the actual builds of system.

Interviewer: Do you use the same tools for vulnerability checks as well?

Richard: Yes … So, we use the vulnerability analysis tools that we use are the same tools that we would use in production as well in development as well. So, as I said double checking to make sure the libraries are the latest version, and we are not introducing risk by using outdated libraries with lots of vulnerabilities in them. So, vulnerability testing using Qualys or Rapid7 or Necess takes place at all points so we know that we are not introducing any vulnerabilities.

Interviewer: Do you have any strategies for conducting security risk and business impact assessment?

Richard: Ahhh…. So, well there are very well-defined methodologies that we use. We have our own methodologies that we use as well as part of that. Also, by making sure that there is a continual process that helps to assure the SDLC is meeting the security requirements and the security requirements are updated as the development continued, and requirements may change.

Interviewer: How do you build security into deployment processes?

Richard: Well, security has to be in deployment otherwise …. Ahhhh.. so the way that we tend to do it is that we will go through some staging so we confirm that nothing is been introduced in the various further move from development into production. Ahhhh… and obviously there are checkboxes that have to be ticked to say yes, we have moved into a pre-production environment. We have tested it and we are happy from a security perspective. We can now move it into production and then we can confirm that no changes have been made to the configurations between moving from pre-production to production so you not introducing any risk. So, you got that assurance all the way through from development into test and into pre-production and into production.

Interviewer: How does security impacts the CI/CD pipeline processes?

Richard: Well, it shouldn’t impact it, its part of it. You can’t have anything that is really looking at CIA without security in it. Ahhhh… so it should not impact it, it should be part of it.

Interviewer: Do security deployment tools fit into your deployment processes?

Richard: Yes, they do because they should be part of it. From a security deployment, the only thing that we should be looking at from the outside of the application from the security deployment perspective is the assurance tools and obviously any automated patching tools that are been deployed. So yeah, they need to be part of.

Interviewer: Would you say your organization has a security culture?

Richard: Ohhh absolutely yeah. As I said earlier if you talk to our global CSO he talk of security as been baked in not bolted on. In the UK we have always had the mantra cyber security as part of everything we do. We very much have a cyber security culture.

Interviewer: Can you describe how the security culture is built or developed in your organization?

Richard: Well its built over years of hard work of getting people to see security not as a hurdle but as an enabler and as something that set us apart from the competition. So very much security is seen as part of doing business rather than a bolt on that is there to block things from happening.

Interviewer: What security resource constraints do you face during software development?

Richard: Ahhhh… so it really depends on who the customer is as to where the development that can be performed. Whether its performed with inside cloud or on-premise with inside secure systems. So, its very much driven by the requirement from the customer.

Interviewer: How often do you give training to your staff?

Richard: Ahhh… security training is a very very regular thing. We have standard sort yearly mandated stuff but for inside a project. Ahhh there will be security regular especially for developers so new developers coming on board will have to go through an induction process which includes security training because when coming on board of the project, it’s a software development but actually there might be different security considerations for this particular project over and above the previous project. So part of the on-boarding process for everybody there is a security training and then there is on-going security training to refresh everybody to make sure they understand any changes coming in or any additional security concerns that may have come out so from lessons learnt and things like that maybe new training because something happen couple of times and therefore people need to be trained and the mistake is not continued forward.

Interviewer: What security skills do you think are the most important for your staff?

Richard: I think the most important security skill anybody can have is the mindset. Everything else we can teach; we can teach them how to find a phishing email or how to make sure they are not introducing buffer overflows but the mindset is the number one thing of getting people just to think securely and think what the bad guy will be looking for. So, mindset I think is the number one security.

Interviewer: Is there anything else that you think is relevant that I have missed from the questions I have asked you so far?

Richard: Ahhhh… so I think the relevancy is making sure that people embed security, so they have that mindset of security from day one. That they are given the freedom to do what they need to do but they understand how to do it securely, so the right checks and balances are in place to capture when mistakes are made. Mistakes will always be made because people are involved, there is a need to make sure they have the culture because would make mistakes. If they identify them early there is a punishment for doing it. Culture of punishment and reward and showing that there to a support and help with the development rather than delay it.

Many thanks once more Richard for sparing time to participate in this research interview but before I let you go, I would like to ask you some demographic information just to keep record of the data collected.

Interviewer: What is your full name?

Richard: Richard Lush

Interviewer: What is your educational level and professional background?

Richard: My educational level is Degree, so B Sc and my professional background is I have been in the industry for over 25 years as an external security consultant since 1998.

Interviewer: What is your current job title?

Richard: Vice President Cyber Operational Security for CGI IT UK Limited

Interviewer: How many people are in the current projects you are handling?

Richard: Ahhh…. Oh my God, so my direct reports I have 95 staff who are reporting to me. In terms of the projects that I have been involved with, I think the largest project that I have been involved with providing security requirements was 127 people.

Interviewer: How long have you been working at the current company?

Richard: 7 years

Interviewer: Many thanks once more Richard but again before I let you go, do you have some other practitioners I can talk to like middle level managers or top managers I can interview to get their own perspective of agile methods security issues.

Richard: Yeah, I will see if I can get a couple of the agile leads to talk to you. Would that be helpful?

Interviewer: Yeah yeah very helpful please if you can do that for me

Richard: That’s fine, I will forward your email on and get them to reach out to you

Interviewer: Ok ok, that’s fine, thank you so much for your time

Richard: You are very welcome, Good luck with your research Mr. Ardo

Interviewer: Yeah, thank you so much. Enjoy the rest of your day. Bye-Bye

F – Participants' Description

Company	Business Sector	Size	Participant Code	Job Titles	Software Development Experience (Years)	Cybersecurity Experience (Years)
Company A	Cybersecurity Solutions	Large	CYBERFOUDco1_ADL1	Cybersecurity Analyst	11	5
			CYBERFOUDco1_SE	Security Engineer	-	13
Company B	Educational Software Solutions	SME	ESSco1_PROD-MGR	Product Manager	13	4
			ESSco1_CTO1	Chief Technology Officer	24	11
			ESSco1_SSE1	Senior Software Engineer	9	3
			ESSco1_BEE	Back-End Developer	9	3
			ESSco1_PROD-MGR	Product Manager	11	3
			ESSco1_PROJ-MGR	Project Manager	16	2
			ESSco1_DE	DevOps Engineer	9	3
Company C	Healthcare Services Company	SME	HEALTHco1_SD1	Software Developer	8	4
Company D	Financial Services	Large	Finco1_SSE2	Senior Software Engineer	17	4
Company E	IT Consulting	SME	ITCONCco1_SC1	Security Consultant	11	2
			ITCONCco1_ITS	Manager, IT Security	26	8
Company F	IT Service Management Company	Large	ITSERVco1_STP-MGR	Security Technical Program Manager	9	6
Company G	Telecommunications Company	Large	ITSERVco1_VP-COS1	VP, Cyber Operational Security	27	18

			ITSERVco1_VP-SSA&E	VP, Security Strategy, Architecture & Engineering	-	29
Company H	Manufacturing	Large	MFGco1_ITA	Manager, IT Security & Operational Risk	17	8
Company I	Customer Relationship Management	Large	CRMco1_FSSD1	Full-Stack Software Developer	8	3
Company J	Digital Forensics Services	SME	LAWENFco1_SC1	Security Consultant	12	8
Company K	Financial Solutions & Services	SME	FSSco1_SDE1	Senior DevOps Engineer	11	6
			FSSco1_FLM	Frontline Manager	11	2
Company L	IT Services & Consulting	SME	ITSERVco2_SE1	Software Developer	6	-
Company M	Digital Services	SME	DSco1_SETL1	Security Team Lead	10	2
Company N	IT Services & Consulting	SME	ITSERVco3_QAA	Quality Assurance Analyst	8	5
Company O	Healthcare Services Company	SME	HCSco1_FED1	Front-End Developer	9	2
Company P	IT Services & Consulting	SME	ITSERVco4_ITS	Manager, IT Security	-	26
			ITSERVco4_SSA	Security Solution Architect	-	7
Company Q	Healthcare Software Solutions	Large	HealthCo1_CTO1	Chief Technology Officer	20	10
			HealthCo1_STL	Security Team Lead	-	12
			HealthCo1_PM	Project Manager	16	5
			HealthCo1_SSE1	Senior Software Engineer	14	9
			HealthCo1_SSE2	Senior Software Engineer	7	5
			HealthCo1_SSA	Software Security Analyst	13	10
			HealthCo1_SD1	Software Developer	5	-

			HealthCo1_SD2	Software Developer	4	-