University of Cape Town

Misalignment - The core challenge in integrating security and privacy requirements into mobile banking application development

Submitted as a Partial Requirement for the Degree Master of Commerce in Information Systems

Memory Machiridza

MCHMEM001

Dedication

To my son Kutenda, for all the times I could not play with you.

PLAGIARISM DECLARATION

1. I know that plagiarism means taking and using the ideas, writings, works or inventions of another as if they were one's own. I know that plagiarism not only includes verbatim copying, but also the extensive use of another person's ideas without proper acknowledgement (which includes the proper use of quotation marks). I know that plagiarism covers this sort of use of material found in textual sources and from the Internet.

2. I acknowledge and understand that plagiarism is wrong.

3. I understand that my research must be accurately referenced. I have followed the rules and conventions concerning referencing (APA 6th Edition), citation and the use of quotations as set out in the Departmental Guide.

4. This assignment is my own work, or my group's own unique group assignment. I acknowledge that copying someone else's assignment, or part of it, is wrong, and that submitting identical work to others constitutes a form of plagiarism.

5. I have not allowed, nor will I in the future allow, anyone to copy my work with the intention of passing it off as their own work.

Student Name: Memory Machiridza **Student Number: MCHMEM001**

Signed by candidate

Signed: ……

Date: 22/05/2016

Acknowledgements

Firstly, I would like to thank God for giving me this opportunity and perseverance to complete this Master's thesis. Only He is my enabler. I would not be where I am if it were not for His mercy, love and kindness.

My deepest gratitude and love goes to my parents for instilling in me an attitude of hard work. Thank you for believing in me and for all the encouragement since my first day of school.

I would like to also thank Professor Michael Kyobe for believing in me, for the words of encouragements, for pushing me (and the rest of the class) and for all the kindness he showed me.

My gratitude goes to my supervisor, Prof Irwin Brown for providing me with directions on this study. His excellent guidance, patience and kindness made me feel confident that I could complete this thesis. I would not have been able to complete this study if it were not for his positive encouragement and advice. I am grateful to you Prof Brown.

I would like to thank my classmates for the push and encouragement I received throughout the duration my studies.

Finally, to Brian, thank you for standing by me and for babysitting our son when I was studying. I will always be grateful.

Abstract

This study identifies and explores the core challenge faced when integrating security and privacy requirements into the mobile banking software development life cycle. Studies on key issues in Information Systems (IS) have been on-going for several decades, with security and privacy moving up the ranks of top issues in IS. Security and privacy requirements can be added into the mobile application development processes by practising secure coding, and/or, by adding a third party security tool. This study gathered data from a single case study; it employs grounded theory methodology to reveal misalignment as the core challenge to integrating security and privacy requirements into mobile banking application development. The forms of misalignment are between security and privacy requirements and (1) external entities, (2) roles, (3) skills and (4) system requirements. The nature of the mobile application domain results in the misalignment forms identified above. Some of the findings indicate the need for further research. Research indicates that mobile application development follows agile methods for development. Agile methods have been compared with Complex Adaptive Systems (CAS). For this reason, research in IS could benefit from studies that focus on CAS as a theory to provide a better explanation on the misalignment issues in mobile application development.

Keywords: Security, Privacy, Mobile Application Development, Misalignment

Table of Contents

List of Figures

List of Tables

Abbreviations

M - Banking-Mobile Banking

IS - Information Systems

IT - Information Technology

SMS - Short Message Structure

USSD - Unstructured Supplementary Service Data

XHTML - Extended Hyper Text Markup Language

SDLC - Software Development Life Cycle

APPS - Mobile downloadable applications (apps)

BA - Business Analyst

**Internet Banking and Online Banking will be used as synonyms in this study

**IS and IT will be used as synonyms in this study

Chapter 1

Background

1.1 Overview of Mobile Banking

Mobile banking has grown rapidly in the last decade, with more banking institutions investing in mobile banking technologies. Mobile banking is defined as the provision of banking services via a mobile device (Angelakopoulos & Mihiotis, 2011). Mobile banking facilitates a better banking experience, with cost reduction and greater accessibility as a consumer can bank anywhere, anytime. In-turn, this can reduce the workload of the bank staff as some functions can be done from a mobile phone. Nonetheless, there are issues that hinder mobile banking from achieving its full potential (Angelakopoulos & Mihiotis, 2011).

Mobile banking can take the form of USSD (Unstructured Supplementary Services Data), SMS (Short Messaging Service), XHTML (Extended Hyper Text Markup Language) and native mobile applications (apps). When making use of USSD, the user accesses the USSD menu by dialing a USSD service code. The USSD transaction results are displayed in real-time; some transactions will repeat via SMS in order to allow a message to remain active on a user's phone. USSD has been adopted across the globe, with banks in African countries, such as South Africa, Kenya and Nigeria, having embraced the use of USSD (Botha et al., 2010). SMS allows information in the form of short messages to be sent between mobile phones. Banks also make use of SMS (Rotimi, Awodele & Bamidele, 2007). A bank's clients can send requests and receive banking information via SMS. Clients can request account balances, pay bills and transfer money (Rotimi, Awodele & Bamidele, 2007). XHTML, which is also referred to as the mobile web, allows Internet-enabled mobile devices to gain access to the web. Bank clients can then do their banking online. A recent trend in mobile banking is the use of downloadable mobile banking applications (apps) which has recently gained popularity. Mobile banking apps have been embraced worldwide (Darsow & Listwan, 2012). Although all the above-mentioned forms of mobile banking are still in use, for the purpose of this study, the focus will be on mobile banking apps.

1.2 Issues in Mobile Banking

In this section, the researcher discusses issues which affect the end-users' use of mobile banking. This is imperative in understanding issues affecting mobile banking end users. Yang (2009) identifies the perceived usefulness, ease of use, credibility and efficiency of mobile banking as the main drivers of the adoption of mobile banking. Al-Jabri and Sohail (2012) used Rogers' diffusion of innovation to identify the factors which influence the adoption of mobile banking. Factors such as relative advantage and complexity (Al-

Jabri & Sohail 2012) correspond to the concepts of perceived usefulness and ease of use (Yang, 2009) with the concept of usability (Angelakopoulos & Mihiotis, 2011).

Issues around security and privacy have been identified as major factors hindering the full adoption of mobile banking. There is an increasing concern among financial institutions and banking consumers about the growth and impact of security issues on mobile banking (Angelakopoulos & Mihiotis, 2011; Worku, 2010). Angelakopoulos and Mihiotis (2011) state that *"people see and hear everywhere about hackers, crackers, computer viruses, identity theft, phishing attacks, spyware, malware and many more other terms that refer to security issues"* (p. 303). The operating systems running on most smartphones and tablets are almost as advanced as the operating systems on desktop computers. Therefore, smart devices are as susceptible to security attacks as desktop computers (Bickford, O'Hare, Baliga, Ganapathy & Iftode, 2010).

It is common for mobile users to make use of the basic protection on their mobile devices for example, passwords and PINs (Personal Identification Number). However, these do not prevent malicious virus attacks or stop skilled hackers from gaining access to personal information on mobile devices (Poon, 2008). The questions around security and privacy have resulted in many bank customers being reluctant to adopt mobile banking (Angelakopoulos & Mihiotis, 2011) because the perceived risk is *"even more important due to the threats of privacy and security"* (Al-Jabri & Sohail, 2012, p. 382).

There are mobile antivirus software that may help safeguard mobile devices from malicious attacks. The main issue is not a question of availability, but a lack of awareness on the availability of the software (La Polla, Martinelli & Sgandurra, 2013). In addition, mobile phones get lost phones, and there is also the risk of theft (Coursaris, Hassanein & Head, 2003).

1.3 Motivation of the Study

Security and privacy requirements are major concerns in software development, especially when developing mobile banking applications. These requirements have been identified as one of the top issues when evaluating the quality and usability of an application (Daud, 2010). Therefore, it is important to look at the challenges faced when seeking to integrate security and privacy requirements into the software development process and find measures to address these challenges. Though security and privacy are different requirements, they cannot be dealt with in isolation.

1.4 Research Objectives and Questions

The main research objective is to gain an understanding of the core challenge faced when integrating security and privacy requirements into mobile banking application (apps) development.

Research Questions

- *What is the core challenge when integrating security and privacy requirements into mobile banking applications?*
- *What factors lead to the core challenge in integrating security and privacy requirements for mobile banking applications?*

An inductive methodology based on grounded theory was employed to uncover the core challenge. The use of open ended questions is useful when making use of grounded theory methodology as it limits the researcher from being influenced by preconceived ideas (Adolph, Kruchten & Hall, 2012).

1.5 Thesis Structure

Chapter 1 Introduction This chapter provides a description of the motivations behind the research and gives the structure of the thesis.

Chapter 2 Preliminary Literature Review This chapter presents a preliminary literature review. In keeping with the research method (which is described in chapter 3), the researcher undertook a preliminary review of the literature before data collection. In light of the research findings a more comprehensive literature review is presented in the discussion section at the end of the results.

Chapter 3 Research Design The research methods and a detailed description of grounded theory methodology are covered in this chapter.

Chapter 4 Findings This chapter presents the findings of the study.

Chapter 5 Discussion The researcher discusses the findings in relation to the existing literature.

Chapter 6 Conclusion This is the final chapter. The researcher offers ideas for future research which were identified in the findings.

It is important to mention that findings from the study have been submitted to the 2016 International Conference on Information Resources Management and the paper is current under review.

Chapter 2

Preliminary Literature Review

2.1 Introduction

In this chapter, the focus will be on a preliminary review of literature related to secure software development. This literature review is preliminary since the research study is following the classical grounded theory methodology that seeks to identify a core challenge (Adolph et al., 2012) in integrating security and privacy requirements to mobile banking applications. At the start of the study it is not known what the core challenge is, as the core challenge will only manifest after the data collection and analysis is completed. Further literature on the emergent core challenge will be discussed in relation to the findings in Chapter 5.

2.2 Software Development

Software development methodologies are a set of guidelines that are followed during the process of software development. It is necessary to begin with an explanation of software development methodologies and how they fit into the process of developing a mobile application. There are two main categories of software development approaches, namely, the traditional development methodologies and agile methods.

2.2.1 Traditional Software Development

Traditional software development methodologies are founded on following a sequence of steps. The sequence begins with the gathering of requirements and ends with the maintenance of the software product. It is essential to have a set of requirements established before development commences (Boehm, 1988). There are several versions of the traditional software development methodologies. The most common ones include the *waterfall model*, the *spiral model* and the *unified model*. This study provides an overview of the *waterfall model* and the *spiral model.*

THE WATERFALL MODEL

The waterfall model was discovered by Winston Royce in 1970. This model describes the process of software development as a number of '*successive stages'*, with the first stage as the gathering of requirements and the last as the release and maintenance of the software (Boehm, 1988). The waterfall model is used mainly on the development of large systems (Royce, 1970). Figure 1 represents the stages that make up the waterfall model.

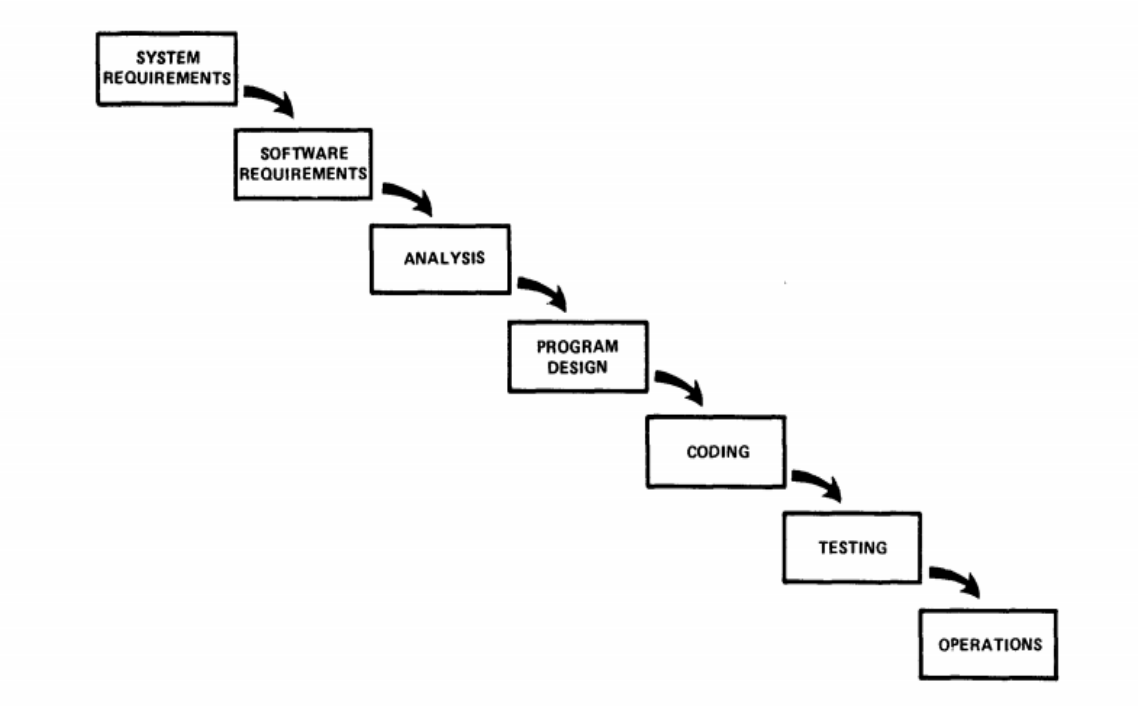

Figure 1: Waterfall Model (Royce, 1970)

According to Royce (1970), the essential stages of the waterfall model are the analysis and the coding phases because they contribute directly to the usefulness of the software product. The waterfall model is document driven. All the requirements must be collected upfront before any development can begin (Boehm, 1988). Figure 1 depicts a linear model; there is no provision to revert to previous stages. If problems are uncovered at a later stage, it may be costly to try and rectify the issues. The waterfall model has been criticised for its lack of flexibility. Real world scenarios do not always progress in linear steps as Royce proposed in 1970 (Boehm, 1988).

SPIRAL MODEL

Barry Boehm developed the spiral model as a means of addressing some of the limitations of the waterfall model (Boehm, 1988). Figure 2 depicts the spiral model. Unlike the waterfall model, which is document driven, the spiral model takes a risk-driven approach (Boehm, 1988).

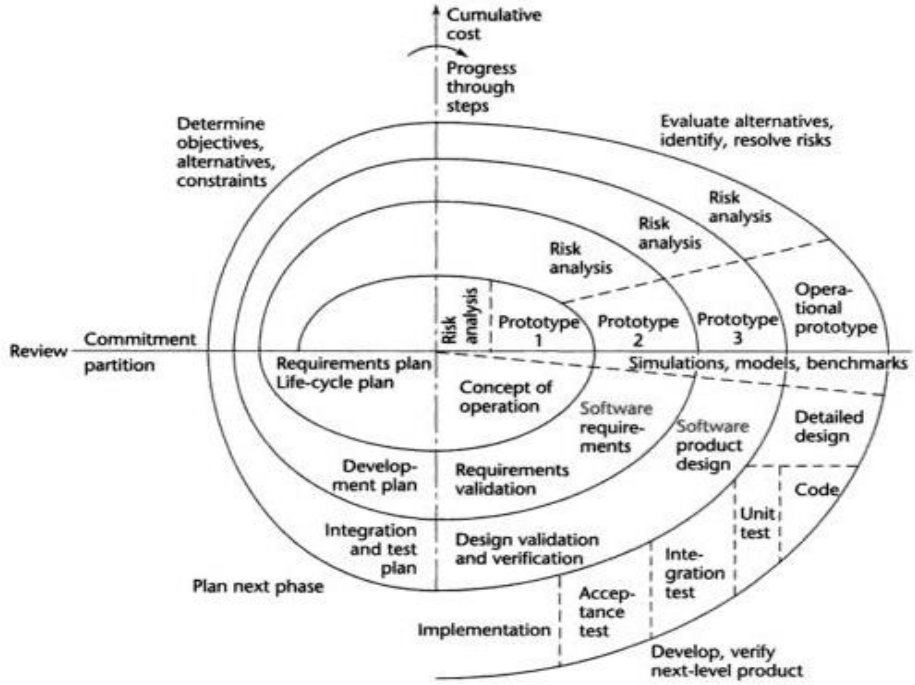

Figure 2: Spiral Model (Boehm, 1988)

Each of the cycles in the software development process goes through repeated iterations which determine the following: the objectives and constraints, the alternatives which will be evaluated in order to identify and resolve risks, the product level that will be developed and tested next, the planning of the next phase and lastly, the review of the end product. Prototyping is an important process that must be carried on throughout each of the cycles. The main strengths of the spiral model are that it incorporates the capability to determine and mitigate risk early in a project and, in addition, provides project stakeholders with prototypes. However, the spiral model does come with certain weaknesses. Since the spiral model is risk-focused, this will mean a waste of resources if applied to low-risk projects (Boehm, 1988).

2.2.2 Agile Software Development

In the last few years, agile methods are among the most important advancements in rapid software development methodologies. The main goal of agile methods is to address the limitations of the traditional software methodologies (Vlaanderen, Jansen, Brinkkemper & Jaspers, 2011). Agile methods focus on providing a quick response to customers' requirements. According to the *'Manifesto for Agile Software Development'*, agile ideas are characterised by adaptability, iteration, prototyping and testing right from the start of the development lifecycle (Fowler & Highsmith, 2001).

Requirements gathering evolves throughout the development life cycle. This facilitates a quick response to changes in customers' preferences, new technological developments and to the emerging competition. Although agile methods support minimum documentation, it is important to have a High-Level Design (HLD) before any coding commences (Cao & Ramesh, 2008).

The more popular agile methods are Scrum, Extreme Programming (XP), Dynamic Systems Development Method (DSDM), Test Driven Development (TDD), Feature Driven Development (FDD) and Kanban. The most used are Scrum and XP (Pikkarainen, Haikara, Salo, Abrahamsson & Still, 2008). The sections that follow provide an overview of Scrum and XP.

SCRUM

Schwaber (2004) describes Scrum methodology as a methodology that is useful for implementing complex tasks that have unpredictable outcome. Scrum methodology is associated with a time-boxed work cycle (2-4 weeks), known as a sprint. Based on a customers' priority list, the team makes a commitment on work delivery. The requirements selected by the customer are translated into a work-cycle. A sprint's committed work must result in a shippable product (Schwaber, 2004). Team members are expected to attend a daily Scrum meeting which should not take more than 15 minutes. The purpose of this meeting is to provide feedback about what has been achieved, i.e. the work accomplished so far, and on any blockers that turned up on the previous day. It also provides a platform where the team can socialise (Schwaber, 2004). After a sprint is completed, the team must demonstrate the functionality of the new component to the relevant stakeholders. At the end of the demonstration, a decision is made whether to *'go live'* with the component or to make some changes (Schwaber, 2004).

The Scrum methodology makes use of a *product backlog* and *sprint backlog*. The product backlog contains all the potential requirements related to a specific product, while the sprint backlog specifies the work to be done in a particular sprint (Schwaber, 2004). Scrum methodology has specific roles and activities that must be clearly distinguished. The roles that make up a Scrum team must include a Scrum Master, the Product Owner and the team.

> **Scrum Master**: The Scrum Master plays the role of a facilitator. It is the responsibility of the Scrum Master to get the team progress in order to facilitate the needs of each team member, thus ensuring productivity and meeting of project deadlines (Schwaber, 2004).
>
> **Product Owner**: The role of a Product Owner encompasses creating and prioritising the product backlog. The Product Owner acts as an interface between the team and the customers. Their role is

to interpret a customer's business requirements and objectives so that they are readily understood by the team (Schwaber, 2004).

The Team: The team is responsible for ensuring that what was committed to, is achieved. When the solution is ready, the team is responsible for demonstrating it to the Product Owner. The team includes business analysts, developers and testers (Schwaber, 2004).

EXTREME PROGRAMMING (XP)

XP is an agile method that relies on team collaboration and interaction. Beck (1999) defines XP as a methodology that enables software development to be interesting, yet at the same time yielding positive results. XP's strength is that it is capable of adapting very quickly to any change to the requirements. This method is characterised by a short development life cycle and continuous feedback (Beck, 1999; Sharp & Robinson, 2008). All the team members interact with each other and collaborate closely; for this reason most communication is by word of mouth (Beck, 1999). The team members frequently share a physical location (Sharp & Robinson, 2008). Beck (1999) provides a guide to the different roles that make up an XP team.

Programmer: At the centre of XP is the programmer. The programmer is responsible for writing the software and the unit tests. This role also touches on communication and collaboration, which are vital to the fu.nctioning of an XP team. Programmers often work in pairs (Beck, 1999).

Customer: A customer knows what is required of the software, usually from a high-level perspective. Customers have a huge influence on a project as they provide written accounts of the requirements. In addition, the customers perform functional testing as they are most knowledgeable about the requirements (Beck, 1999).

Tester: In an XP team the tester's role is to guide the customers who write the functional test cases (Beck, 1999).

Tracker: A tracker is involved in tracking the progress of the team to enable them to plan more effectively. Trackers look at the project from an overall perspective in order to identify and manage risk (Beck, 1999).

Coach: The coach is responsible for providing guidance to the team. He or she points out any deviations from the project schedules and can make decisions on changes to the project schedule. The coach is expected to have an in-depth knowledge of XP practices (Beck, 1999).

2.2.3 Challenges in Software Development

For the past three decades' security and privacy issues have been areas of interest in Software Engineering and related IS fields (Daud, 2010; Davis, Humphrey, Redwine, Zibulski & McGraw, 2004; Kim, 2011). Key stakeholders, such as management, software development teams, end-users and law enforcement officials need to be aware of the various problems in these areas if they are to provide better services to the user (Brancheau, Janz & Wetherbe, 1996). Usability is one of the key IS issues in managing mobile banking applications. Usability is directly related to security as it is important to design an interface that is both secure and that provides a good user experience (Venkatesh, Ramesh & Massey, 2003). Since the traditional usability guidelines may not be fully applicable to mobile applications this can pose a challenge to the development teams of mobile apps (Zhang & Adipat, 2005).

2.2.4 Mobile Application Development

Mobile native applications, which are also known as mobile apps, are software programs, which run on smart phones and other smart devices such as tablets. Mobile application development is the process of creating a mobile application. The demand for mobile applications is increasing, as the various mobile platforms compete to outdo each other in improving user experience and performance. This increase in demand is influenced by the availability of cheaper smart mobile phones, greater access to the Internet and cloud computing (Pãvãloaia, 2013). Software development for mobile applications is unique because of the levels of maturity and the volatile mobile environment. Mobile application development is characterised by short periods for software deployment, many competing software providers, a number of stakeholders, undefined requirements and changing mobile platforms (Abrahamsson, 2007). Agile methods appear to be suitable methods for mobile software development as they follow short delivery cycles. The next section presents an overview of agile development and how it relates to mobile application development.

AGILE AND MOBILE APPLICATION DEVELOPMENT

No specific software methodologies have been created for the sole purpose of developing of mobile applications. Consequently, software development teams take a risk when they use one of the available software development methodologies (Stubbs & Wilford, 2014). Mobile application development is about responding to changes in the mobile application consumer market. Feedback from the mobile apps consumers can help when deciding what the best software project to invest in is, as well as the choice of components to be developed and released into the mobile app market. Traditional software development methods such as the waterfall methodology are unsuitable for mobile application development because traditional methodologies require pre-defined, unchanging requirements. Agile methods provide the means

for mobile application providers to develop and release their applications as smaller components so that they can test the reaction of the market before they spend large amounts of money developing the software further (Stubbs & Wilford, 2014).

Schadler and McCarthy (2012) provide a comparison of the '*PC era'* and the '*mobile age*'. The preferred development methodologies of these two periods are different. The '*PC era*' is distinguished by its use of the waterfall development method while the 'mobile age' employs agile methods. A number of factors have been identified as the driving force behind the adoption of agile methodology in the development of mobile applications, for example, competitiveness in the market, shorter delivery cycles and the ever changing customer requirements (Abrahamsson, 2007; Abrahamsson, et al., 2004).

2.3 Secure Software Development

The terms *'application security'*, *'software security'* and *'information security'* are often used interchangeably; however, they actually denote to the different dimensions of security that apply to software. The focus of this study is software security. Although there is no standard definition for software security, for the purpose of this study, software security is understood as the *"building[of] secure software: designing software to be secure; making sure that software is secure; and educating software developers, architects, and users about how to build security in"* (McGraw, 2006, p. 20). McGraw (2006) makes a clear distinction between application security and software security. Application security focuses on securing and supporting an application from attacks by viruses and other harmful software packages, after the software has been developed. Application security looks at aspects as the patching of software (McGraw, 2006).

Whitman and Mattord (2003) define information security as "*the protection of information and its critical elements [such as] systems and hardware that use, store and transmit that information"* (p. 8). Information security focuses on preventing theft, the deletion and modification of information in existing systems, all of which can result in harm to the stakeholders. Thus, information security ensures that the right people have access to the correct information. Information security is composed of a number of levels or layers. The layers include physical security, personal security, operations security, communication security, network security and information security. By addressing the different levels of security, potential threats can be mitigated (Whitman & Mattord, 2003).

2.3.1 Developing Secure Software

To ensure that software is secure requires that security is in-built. This means, among other things, adherence to the correct coding standards and prescribed guidelines. It is essential that the whole development process complies with both internal and external security policies (Oueslati, Rahman & Othmane, 2015). Software security should receive attention right from the earliest phases of the development lifecycle to prevent issues going undetected which could disrupt later phases of the development cycle (Daud, 2010).

Daud (2010) pointed out that during software development, security issues can be created intentionally or unintentionally. Baca and Carlsson (2011) list some common security vulnerabilities that may occur during software development. These include buffer overflow, misplaced trust, race condition and poor random number generation. A buffer overflow occurs when the amount of data exceeds the capacity of the buffer. This commonly happens when using '*unsafe*' programming languages such as C and C++ (Baca & Carlsson, 2011; Davis et al., 2004). A race condition arises when two or more processes share a data item; the end result will depend on which process or processes are executed first. This condition can introduce instability into the system, which, in turn, can lead to a denial of service because the system has crashed (Baca & Carlsson, 2011). Other threats to security include memory leaks and deadlocks (Yin, Yuan, Zhou, Pasupathy & Bairavasundaram, 2011).

There are a number of key functions that ensure the security of software. Among such functions are the definition of security requirements, the creation of a security mitigation plan, threat modelling, documenting known security vulnerabilities, security testing and security management (Daud, 2010; Kim, 2011). Security requirements can be non-functional, functional or derived. A non-functional security requirement refers to the essential properties of a system, for example, performance. A functional security requirement represents the behaviour of a system, for example, masking of a password. A derived requirement combines functional and non-functional security requirements. Security requirements are usually documented by the customer or the security engineer or security analyst (Daud, 2010; Kim, 2011). Figure 3 depicts an iterative method for integrating security requirements into the SDLC.

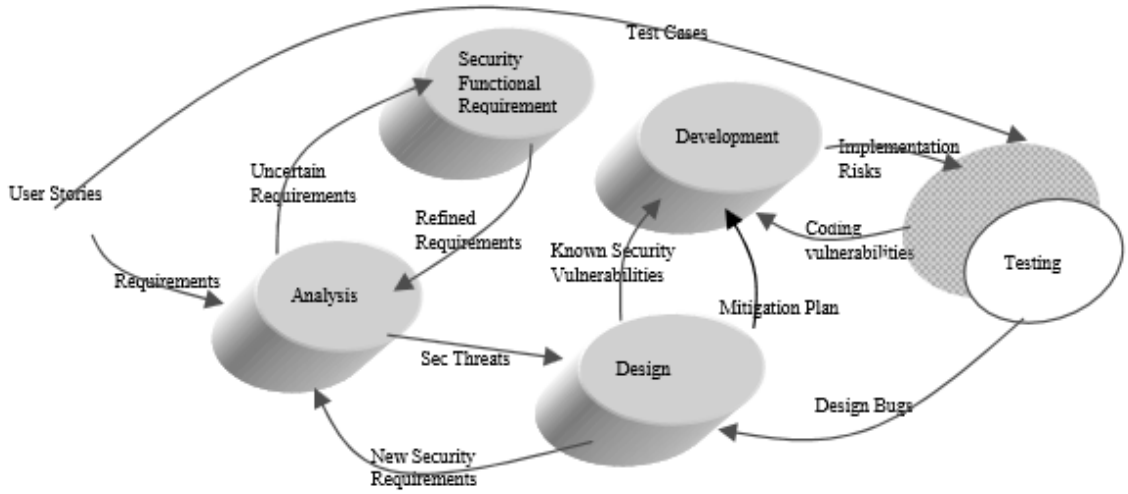

Figure 3: Iterative Approach to Security Integration into the SDLC

Based on the analysis process, a threat model is designed. The analysis process must be performed until the security analyst or security engineer is satisfied that there is no missing information. Possible security issues can be prevented by identifying the entry and exit points. A data flow diagram can be drawn up for this purpose (Daud, 2010; Kim, 2011). When development is completed, the security requirements are tested. Security testing involves functional and non-functional testing. Security testers are normally referred to as ethical hackers because of their ability to think and test the software in the same way an attacker would (Daud, 2010, Kim, 2011).

2.3.2 Security Standards and Best Practices

In order to build a secure system, simply following established standards and guidelines does not guarantee that a system is secure. Standards and guidelines have been created to provide help and guidance. The International Organization for Standardization (ISO), the National Institute of Standards and Technology (NIST), International Electro-technical Commission (IEC) are a few of the bodies that have set up security standards. In 1974, Jerome Saltzer and Michael Schroeder proposed eight principles, which still apply to secure software development (Davis et al., 2004). Table 1 provides a summary of the eight principles as documented in Davis et al., (2004).

Principle	Definition
Economy of mechanism	*Simple and small designs*
Fail-safe defaults	*Focus on permissions instead of inclusion*
Complete mediation	*Access must be checked by authority*
Open design	*Design should not be a secret*
Separation of privileges	*Multi-factor authentication*
Least privileges	*Minimise unnecessary access*
Least common mechanism	*Access control*
Psychological acceptability	*Usability*

Table 1: Secure Software Development Principles

Further guidelines, such as the validation of data fields, simplicity, code reuse, using trusted and tested encryption mechanisms and encryption of passwords, will further strengthen the security of software (Davis et al., 2004). It is also important to note that there are tools which can aid in the development and delivery of secure code, for example, static scans and dynamic scans. Code reviews can also help minimise security issues (Davis et al., 2004).

2.3.3 Privacy in Software Development

Protection of consumers' privacy is crucial when building software. It is essential to ensure that consumers' personal information is safe from illegal access and use. Consumers are normally concerned about how their personal information is used (Liebenau, Elaluf-Calderwood, Karrberg, & Hosein, 2011). There are cases when certain information is collected without the consumers' knowledge (Degirmenci, Guhr, & Breitner, 2013). If consumer information is to be used, the consumer should be made aware of this before the information is disclosed. It is the duty of governments everywhere to protect national security. Governments are also responsible for drawing up regulations which specify how the personal information of every individual must be protected. It is essential for development teams to know the laws and regulations which regulate the protection of personal information; in the case of South Africa it is *The Protection of Personal Information (POPI) Act,* act number 4, 2013. Information storage and use must adhere to the regulations and legislation prescribed by governments (Degirmenci et al., 2013).

2.4 Security Issues in Information Systems

Much IS research has been devoted to matters of security (Daud, 2010; Davies et al., 2004; Kim, 2011). Security is a key issue in software delivery and can affect a business negatively if the proper measures are not taken to avoid security breaches or any software anomalies. In traditional software development methodologies, for example, in case of the waterfall method, security testing is normally performed at the

end of a project just before the product is released. If security vulnerabilities are found at this final stage, it can create serious problems around software delivery. Agile methodology has been widely adopted in the software development industry due to some of the advantages cited in one of the sections above (Vlaanderen et al., 2011); even though agile methods do not specifically address the security risks which can afflict the development of software applications (Siponen, Baskerville & Kuivalainen, 2005). Peeters (2008) points out that the agile methodologies do not provide the means to track security requirements. Technology has advanced; nonetheless, security still remains a key issue (Siponen et al., 2005).

2.5 Mobile Application Platforms

Until recently, Apples' iOS has been the leading operating system used in smart phones. The Android system is now in the lead mainly because of the platform's openness and the Google brand name. Windows and Blackberry also have some market share (Elmer-DeWitt, 2013). In this section, the focus will be on Android and iOS as they are the most popular platforms for smart phones. The two platforms take almost 75% of the smart phone market worldwide. According to a study conducted by Canalys, Apple and Android apps make up 74% of the total number of downloaded applications; Android has had the largest number of downloads (Elmer-DeWitt, 2013).

2.5.1 Android Platform

With millions of users worldwide, the Android platform is the fastest growing mobile platform. Android is based on open source Linux and can be developed using a number of existing operating systems, for example, Mac OS, Windows and Linux (Goadrich & Rogers, 2011). Such openness allows anyone who has sufficient experience to develop a mobile application (Goadrich & Rogers, 2011). The Android platform is popular because it is compatible with the smart devices produced by several manufacturers, such as HTC, Samsung, LG and Motorola. The Integrated Development Environment (IDE) does not require hardware that is made specifically for the Android operating system. Eclipse is the IDE which is commonly used when building Android apps. Android has managed to support cheaper and more affordable smart phones and this has allowed mobile users to change from basic features phones to a smart phone. Compared to its main competitor, iOS, Android is fairly new in the marketplace and has had its first release in 2007 (Goadrich & Rogers, 2011). Android offers a marketplace for application distribution for sellers and for buyers to easily purchase and install mobile applications.

2.5.2 iOS Platform

iOS is the operating system developed by Apple. It is derived from the Macintosh OS X and runs on hand held devices such as iPhones, iPads and iPods. iOS is considered one of the earliest mobile operating

systems. Unlike Android, iOS only runs on Apple devices. The IDE on which iOS mobile applications are developed is called XCode and it only runs on the Mac OS. The language in which these applications are developed is called Objective-C programming language. However, with the introduction of iOS 8, there is a move towards the use of the SWIFT programming language to build iOS apps (Nahavandipoor, 2014). Additional applications, such as instruments, can be used for more complex tasks such as memory management (Goadrich & Rogers, 2011).

2.6 Summary

This chapter has provided a preliminary literature review that has highlighted important literature on software development, the development of secure software and on mobile application development. The focus has been on different software development methodologies, with an emphasis on agile methods and how they relate to mobile application development. Secure software development also received attention because it was necessary to provide an understanding of what secure software development involves and what issues can arise when no allowance has been made for dealing with human error by members of the development team.

Chapter 3

Research Methodology

3.1 Introduction

In the previous chapter, the researcher provided a preliminary literature review on software development, security and privacy in software development. Researchers in the fields related to software engineering and information systems are continuously focusing on improving the quality of research undertaken by implementing the most appropriate research methods (Walsham, 1995). The purpose of this chapter is to provide a detailed description of the research methodology that was followed in this study.

The study adopts a form of the grounded theory methodology often termed Glaserian or classical (Jones & Alony, 2011; Matavire & Brown, 2013). The term 'grounded theory' is often used as an excuse for not having a sound research methodology (Suddaby, 2006). To alleviate this concern, the procedures and principles of the Glaserian grounded theory methodology, such as open coding, memo-writing, constant comparison, selective coding and theoretical coding will be transparently explained and executed in the study. Data collection techniques, reliability, validity and ethical consideration will be discussed in subsequent sections.

3.2 Grounded Theory Methodology

Grounded theory methodology and 'grounded theory', the final product of a research should clearly be distinguished with the former being a method, approach or methodology followed in the research while the latter is the end result of a research project (Matavire & Brown, 2013; McCallin, 2003). Grounded theory methodology is based on the belief that an individual within a group defines situations in a common pattern of behaviour that is prevalent within the group (McCallin, 2003).

Grounded theory methodology follows three set principles; emergence, constant comparative analysis and theoretical sampling. The principle of emergence involves the researcher(s) having no theoretical framework as a study lens. This methodology is based on the belief that the research process and the research product will evolve during the research (Matavire & Brown, 2013). Pickard (2007) also includes the research question and memo-writing as part of the principles of grounded theory methodology. The research questions should emerge during the data collection and analysis. In other words, a researcher should approach a study with an open mind, that is, will not impose pre-conceived ideas or knowledge onto the data, but will allow the data and explanations to emerge (McCallin, 2003; Pickard, 2007). The process

of open coding – which will be discussed later in the chapter, assists the researcher in identifying the emerging concepts.

McCallin (2003) states that the main advantage of grounded theory methodology is that it seeks out explanations to what is actually going on in real life scenarios instead of what should be going on. Grounded theory methodology has proved an effective way of understanding and explaining a problem from the perspectives of the participants by paying attention to the ways that the participants are affected when dealing with the problem (Adolph, Kruchten & Hall, 2012). Grounded theory methodology is a qualitative method of data collection and analysis. The data collected in the course of a research project is used to build theory inductively (Matavire & Brown, 2013). However, it is worth mentioning that grounded theory methodology is not always used for theory generating. In some cases, it is used to gather information for a foundational study at the start of a more extensive research project (McCallin, 2003).

Grounded theory methodology is widely used in social studies; but has been adopted in Software Engineering and Information Systems related studies (Matavire & Brown, 2013). The use of grounded theory methodology for Software Engineering and Information Systems related research is supported by Myers (1997) as *"extremely useful in developing context-based, process-oriented descriptions and explanations of the phenomenon"* (p. 9), and useful way of analysing data in Software Engineering and Information Systems research (Urquhart, 2000). Other researchers in the fields of Information Systems (Douglas, 2003; Matavire & Brown, 2013) and Software Engineering-related domains (Adolph, Kruchten & Hall, 2011; Adolph et al., 2012; Coleman & O'Connor, 2007; Joorabchi, Mesbah & Kruchten, 2013) have successfully adapted grounded theory methodology in their research studies.

3.2.1 Approaches to Grounded Theory Studies

Grounded theory methodology was originally conceived by Barney Glaser and Anselm Strauss, but due to some differences in views, two main types of grounded theory methodologies emerged from their differences; the Glaserian and the Straussian schools (Jones & Alony, 2011; Matavire & Brown, 2013). The Glaserian school advocates that a researcher keeps an open mind from the beginning of the study to the end; the Straussian school encourages a researcher to begin with a general guideline which will allow him or her to prepare semi-structured questions for the interviews (Jones & Alony, 2011). This does not mean that a researcher who adopts the Glaserian approach will begin a research study without knowing anything about the field under study. Suddaby (2006) points out that it is a myth "*that the researcher is a blank sheet devoid of experience or knowledge…. without a defined research question"* (p. 634).

The concept of the 'emergence' of categories is associated with the Glaserian approach while the Straussian approach is associated with the 'forcing' of categories. The reason behind this conviction is that the Glaserian approach allows for data categories to manifest on their own while the Straussian approach employs a paradigm model (Matavire & Brown, 2013) which means that the topic being studied has been identified before the research commences (Douglas, 2003).

The Straussian approach is more structured and the researcher is an active participant (Douglas, 2003; Jones & Alony 2011). The Straussian approach provides the novice researcher with practical guidelines to research (McCallin, 2003). Whichever approach a researcher chooses to follow will depend on the individual's understanding of the different approaches. A researcher must have a solid grounding in the grounded theory methodologies and various qualitative research methods to establish an understanding of grounded theory methodology and how it fits or differs with other qualitative methods (McCallin, 2003; Pickard, 2007).

3.2.2 Approach Used in the Study

Figure 4 is an illustration of the grounded theory methodology that will be followed in this research. Data collection is the first step in the research process. Data from interviews is recorded as memos and notes. Other sources of data may include documents, prior knowledge and existing literature.

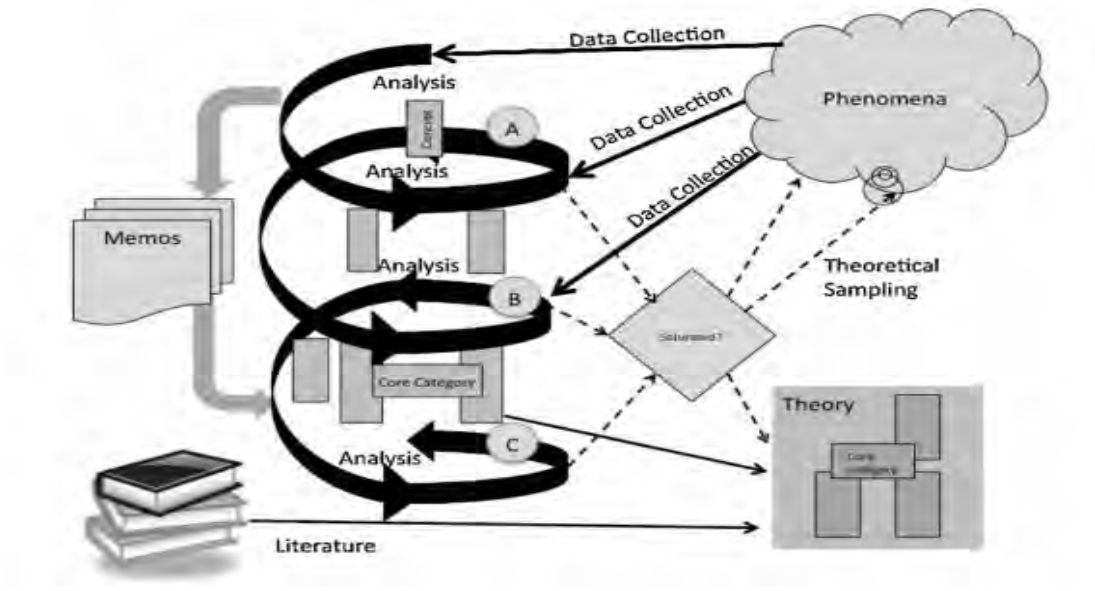

Figure 4: Grounded Theory Methodology (Adolph et al., 2011)

After data is collected, the researcher begins the process of analysis in order to identify any patterns that recur in the data. The patterns are the building blocks for theory building (Adolph et al., 2011). The core

category is identified by means of the iterative research process which allows the researcher compare the concepts that emerge from the interviews with all the participants. The process continues until saturation point is reached when concepts raised in the interviews are repeated (Adolph et al., 2011). At this point, a literature review is undertaken in order to substantiate the patterns identified at the end of data collection and analysis.

In the sections that follow, the researcher provides an overview on areas that are dominant when discussing grounded theory methodology such as the role of literature review, theoretical sampling and theoretical saturation.

3.2.3 Grounded Theory Methodology and Literature Review

Grounded theorists such as Glaser are proponents of the classical grounded theory methodology, which considers first collecting and analysing the data then examine the literature to substantiate the concepts that have emerged from the data (Adolph et al., 2011). However, Suddaby (2006) argues that the original theorists of the grounded theory methodology did not insist that researchers ignore existing empirical knowledge. Dey (2007) supports this notion by stating that researchers should pay attention to the extant literature and not confuse an *"open mind with an empty head"* Dey (2007, p. 20). McCallin (2003) points out that we live in a world that contains vast amounts of information. It is therefore highly unlikely that a researcher will begin with no knowledge of the topic under study. The researchers' previous knowledge becomes part of the data. It is possible that if the relevant literature is ignored, the ideas that emerging during the data analysis may be presented as if they are new or original (Bryant & Charmaz, 2007). The literature can validate the emerging concepts. In turn, the researcher can weigh the relevance and validity of the literature and prior knowledge, and then use all this information to build theory (Andrade, 2009).

Researchers are encouraged when using grounded theory methodology to make use of existing literature and previous personal experiences to create a theoretical base of the study (Urquhart, 2000). Adolph et al., (2011) support that literature can be an indication of what has already been studied. For this reason, research should be done in the context of current research. It is, however, important for researchers to refrain from being presumptuous based on the existing empirical knowledge (Suddaby, 2006). In this study, the initial literature review guided the researcher on what has been studied in relation to the research questions and thus helped identify the gaps in the literature. The initial literature review provides the researcher with some level of confidence on the value of the study to the field of research. The literature on key concepts such as security, mobile application development, software methodologies exists. However, there is a lack of literature that addresses the challenges faced when integrating security and privacy requirements in

software development, especially in agile development methods, which are software development methods mobile application development is likely to follow (Siponen, Baskerville & Heikka, 2005).

3.2.4 Sampling

In grounded theory methodology, the population under study relates to the phenomenon that is being studied hence the use of statistical sampling may not be applicable (Adolph et al., 2012). The choice of a sampling technique largely depends on the research questions and the research strategy. Sampling techniques can often be combined. Grounded theory methodology may make use of two main types of sampling types; theoretical sampling and purposive sampling (Savin-Baden & Major, 2013). For the purposes of this study, the researcher made use of both theoretical sampling and purposive sampling.

PURPOSIVE SAMPLING

Purposive sampling involves the researcher selecting participants based on their likelihood to provide answers to the research questions (Savin-Baden & Major, 2013). This sampling technique allows for the selection of a case based on the aspects that matched the research objective, i.e. a case in which security and privacy requirements were being integrated into the mobile banking application development process. Purposive sampling was also used as a means to identify and recruit the first interview. Heterogeneous perspectives on challenges faced when integrating security and privacy requirements were obtained from developers, business analysts, testers and the project manager. The main aim was to generate as many open codes as possible until no new codes were identified (Adolph et al., 2012). In the current study, the team involved in the developing of mobile banking applications is approximately twenty. The researcher collected data until a 'saturation' point was reached.

THEORETICAL SAMPLING

The principle of theoretical sampling is an iterative process that involves collecting, coding and data analysis (Adolph et al., 2011; Glaser & Strauss, 1967; Matavire & Brown, 2013). Glaser (1978) define theoretical sampling as *"the process of data collection for generating theory whereby the analyst jointly collects codes and analyses his data and decides what data to collect next and where to find them, in order to develop his theory as it emerges"* (p. 30). The sampling technique is based on the emergence of theory from the data collected. This guides the researcher on the selection of the next interview participant and questions. Concepts identified in one interview will be evaluated and validated in another interview. The researcher may need to confirm findings collected from a previous interview or may need another participant to elaborate on a concept that materialised in a previous interview (Glaser & Strauss, 1967).

In the initial phases of data analysis, several prospective categories can result from theoretical sampling. Categories then become saturated and the theory becomes more defined. The amount of data required for the data analysis also decreases. The researcher is then focused on more specific categories. Once a core category is identified, the process of theoretical sampling comes to an end (Glaser & Strauss, 1967).

Although the principle of theoretical sampling explicitly indicate that the participants number and details are unknown to the researcher before embarking on data collection, for the practical purpose of this study, the researcher identified the potential participants for the study by selecting a specific case. Theoretical sampling is undoubtedly an important aspect of the grounded theory methodology. This sampling technique can be time-consuming, and for the purposes of this research it was not fully employed.

THEORETICAL SATURATION

Glaser and Strauss (1967) define theoretical saturation as *"the criteria for judging when to stop sampling"* (p. 59). The researcher would have reached a point where no new data is emanating from the data collection and analysis. Recurring incidents indicate that the category is saturated. Once a category is saturated, the researcher will then try to saturate other categories (Glaser & Strauss, 1967). When saturation is reached, the researcher can affirm that the core category is an indicator of the variation of the data collected. It is important to note that the data within the core category can still further be modified by constant comparison or and literature review. The processes in grounded theory methodology are known to be a continuous process (Glaser & Strauss, 1967). The process of theoretical sampling determines the number of participants for the study but that does not justify the use of a small sample size (Bryant & Charmaz, 2007).

3.3 Philosophical Perspective and Grounded Theory Methodology

The section below provides a summary of some of the research paradigms followed in research studies. Particular focus is on interpretivism and positivism. According to Pickard (2007), the choice of a research paradigm should be based on the '*ontological question*', '*epistemological question*' and the '*methodological question*'. The ontological question is *"What is the nature of reality?"* while epistemological question relates to *"What is the relationship between knower and the known?"* and lastly, the methodological question is *"How do we come up to know it?"*. Pickard (2007, p. 6) believes the answers to these questions will help a researcher determine the most suitable choice of a research paradigm for their study. In a study that follow the grounded theory methodology, some researchers state that grounded theory

methodology is not embedded in any philosophical approach. The researcher can determine what philosophical approach to follow based on the nature of the research questions (Matavire & Brown, 2013).

Positivism is a research paradigm that stems from natural sciences research. Positivists believe that what is real is what can be seen. The reality is objective in that the researcher is there as an observer and is not part of the research. Based on the observations made, the researcher can come to a conclusion on what is real (Pickard, 2007). The objectiveness of the study makes replication possible as the researcher is independent of the study settings. In a positivist research, hypothesis formulation is essential to the research process. These hypotheses are then tested for substantiation purposes thus, the choice of tools in positivist studies is usually statistical analysis. The main purpose of a positivist research is to predict events based on statistical probability (Pickard, 2007).

Interpretivists believe in multiple complex realities which are contextual. The researcher and the research phenomenon interactions produce results that are specific to a given time and context (Pickard, 2007). Interpretivists use qualitative data to in order to understand an individual's behavior (Pickard, 2007). An interpretive study tries to gain an understanding of a phenomenon in order to make decisions on the appropriateness of transferability of findings to a similar context (Pickard, 2007). The use of *'thick description'* associated with qualitative interpretive research can help in clarifying issues that were initially unclear (Carcary, 2009; Walsham, 1995) and this is through the eyes of individuals experiencing the situation and the researchers' interpretation of these views (Andrade, 2009). The researcher plays a key role as his/her quality of arguments and interaction with participants is critical for the research outcome (Andrade, 2009). In the current study, although the researcher is also involved in the day-to-day implementation of mobile banking application, the researcher takes the role of an observer, which in-turn allowed the researcher to provide more descriptions that are detailed to help to answer the research questions (Walsham, 1995).

Interpretive research has been used in software engineering and IS related research to provide a deeper understanding of a phenomenon in an organizational environment. Software engineering and IS researchers are faced with complex problems which, when analysed within a social and organizational context at a particular moment, can lead to misinterpretations. In this study, the need was to understand the core challenge faced when integrating security and privacy requirements in mobile banking applications. The focus was to understand challenges from the product owners'/business analysts' perspective, the developers' perspective and the software testers' perspectives.

Grounded theory methodology does not follow a specific paradigm; hence, it is up to the researcher to make an indication of the theoretic stance. The researcher chose to follow an interpretive stance based on the research questions and the way the data was collected.

Research Problem

The researcher begins with a general focus and an open mind from the beginning of the research, taking into consideration the nature of a study that follows the grounded theory methodology. Research can be exploratory, descriptive or explanatory. In exploratory research, the researcher seeks to achieve a better understanding and insight on preliminary knowledge. Consequently, the researcher may have no prior knowledge to the problem under study. Exploratory research does not seek to gain final results but forms a foundation for other future studies. Exploratory research is more appropriate when the phenomenon under study is new or the researcher is looking at a different angle of an existing phenomenon (Savin-Baden & Major, 2013).

Descriptive research is used to give more detailed information to an already known phenomenon to create a fuller picture. What is already known is supported with what is found during research. The main objective of descriptive research is to provide more precise information on a known issue (Savin-Baden & Major, 2013). On the other hand, an explanatory research involves understanding the cause and effect relationships of the ideas in the research. The explanatory research considers how concepts correlate. Explanatory research can only be conducted after exploratory and descriptive research has been conducted (Savin-Baden & Major, 2013).

The current study is exploratory as the nature of the variables is unknown at the beginning of the study (McCallin, 2003). Specific concepts on misalignment as the core challenge emerged throughout the research process (Charmaz, 1995). The grounded theory methodology followed in the study supports openness with the assumption that participants are willing to provide the required information (McCallin, 2003). Prior knowledge indicates the existences of challenges into mobile software development (Joorabchi, Mesbah & Kruchten, 2013; Hammershoj, Sapuppo, & Tadayoni, 2010; König-Ries, 2009; *Wasserman*, 2010). Challenges noted are not specific to security and privacy requirements and may simply be as a result of incorrect or inaccurate information and personal biases (Creswell, 1994 as cited in McCallin, 2003). On another note, mobile application development is still in its early phases and the software development processes and procedures are still being established. This makes the whole process of mobile application development to be considered ‘new’ although software development has been on-going for the past decades. Therefore, an exploratory study would be most appropriate.

3.4 Data Collection

The credibility of any research results is largely influenced by the time period the research was done, the location of the research and participants involved. Time is an important factor, especially in qualitative research as it has an impact on the responses of the participants (Savin-Baden & Major, 2013). The case study used in the current study is a mobile application development team that focuses on a mobile banking application. The team follows the scrum methodology. In the section that follows, a case study is described as the research site and the boundary of the study.

3.4.1 Case Study

The use of case study in Software Engineering and IS related research has helped in understanding the interactions between humans and technology. A case study enables the researcher to get a deeper understanding of the phenomenon under study (Andrade, 2009). A case study can be defined using three perspectives. Firstly, case studies as a boundary of the study, secondly a case study as a research approach to data collection and analysis and lastly, a case study as the end product of the research or study. In the current study, a case study will not be employed as a research approach as the research in making use of the grounded theory methodology.

Using a case study as a boundary limits the study to a specific place and time period (Savin-Baden & Major, 2013). A case study as a boundary of the study relates to the software development team that is used in the current study. Lastly, the case study is the end result of the research which is a written document detailing the core challenge faced in developing banking mobile applications and integrating security and privacy.

A case study can be used in positivist, critical and interpretive research; can be qualitative, quantitative or both (Klein & Myers, 1999). A researcher is directly involved with the participants when using a qualitative interpretive case study. This involvement with participants has been seen as a limitation but Andrade (2009) sees it as one of the advantages as it helps the researcher to comprehend and deeply understand the research problem. In the current study, the researcher will be making use of a qualitative case study and is direct involvement with the participants.

The use of case study in research has been adopted widely in different domains of study such as business, nursing, social work and psychology, with the aim of increasing knowledge about a group, organization and a related phenomenon (Yin, 2013) and similarly, in the field of Software Engineering (Runeson & Höst, 2009; Walsham, 1995). Case studies can be adopted in Software Engineering studies as the process of software development is conducted by a group of individuals subjected to certain conditions. Social and

political questions play a role in this process, making it suitable to adopt case study as a boundary of study and as a project output (Runeson & Höst, 2009). In this research, the researcher will adopt an interpretive case study approach as there is a need to understand the phenomenon related to the core challenge faced in integrating security and privacy in developing mobile apps through the interpretation of the business analysts, developers, testers and the project manager.

Case studies can be categorized by purpose. In an exploratory case study, research questions are only defined after some data collection has occurred. Descriptive case study makes use of little or no theory to guide the study. Instrumental case study makes use of existing theory in order to validate or/and improve the existing theory. Interpretive case study aims at developing theory. Comprehensive findings are presented in the form of 'thick descriptions'. An explanatory case study looks at a cause-effect relationship. This can be either quantitative or qualitative. Lastly, an evaluative case study is used to explain real world scenario that cannot be achieved by the survey (Savin-Baden & Major, 2013). For the purpose of this research, the interpretive case study will help in developing "*conceptual categories or theories*" (Savin-Baden & Major, 2013, p. 155) as the main purpose of the research is to build theory using grounded theory methodology.

Single or multiple case studies can be applied to a research. In this study, the research will adopt a single case study. Savin-Baden and Major (2013) discussion some advantages and disadvantages of a single case study. Some of the advantages include flexibility in research goals, philosophical stance and research approach. In the end, it facilitates an in-depth study of a phenomenon. The purpose of a case study is to ultimately develop theory. Although case studies have been used to build theory in positivist studies, the same concepts can be applied to build theory in an interpretive study. The use of a case study only may not be enough for an interpretive study. Andrade (2009) recommends combining grounded theory methodology with a case study to "*assist the researcher in the definition of unit of analysis*" (p. 46). The researcher has espoused the recommendation by making use of a case study with the grounded theory methodology.

The research was carried out when a security project was in progress. In this case, the time had an influence on the responses of the participants to the research questions. Space, site or location of a study is an important factor. Space is not only limited to the physical location; but also to factors influencing thoughts and actions. A researcher may make use of a single site or multiple sites depending on the goals of the research. When using a single site, researchers must not make unjustifiable generalisations. The use of single sites may result in limited information revealed by the respondents due to issues of data security and privacy. Multiple sites, on the other hand, allow for comparison of data between sites, enabling the

justification of generalization. When a researcher is uses multiple sites, time may limit the researcher in gaining comprehensive information on each individual site (Savin-Baden & Major, 2013). In the current study, the researcher makes use of a single case study. It is worth mentioning that the participants in the study are in two different countries. Identifying the right participants for the research study is an important aspect for any research. Participants are the selected group of individuals whom the researcher collects data from. The researcher must consider time, population and accessibility when selecting participants for the study. Accessibility is an important factor as it determines whether the researcher will have access to the necessary information (Savin-Baden & Major, 2013).

3.4.2 Interviews

Collecting data using interviews is commonly used in qualitative research as interviews allow the researcher to gain comprehensive information from the participants. An interview gives the researcher a one–on–one conversation with a participant, allowing the researcher to observe body language. Interviews can be structured, semi-structured or unstructured (Savin-Baden & Major, 2013). In a structured interview, the researcher follows a set of well-defined closed ended questions. These questions will be asked to all the participants; which can be helpful if the researcher intends to compare the answers of individual participants. This form of interviewing is also useful if the research results need to be replicated in a similar study. The semi-structured interview process is when the researcher follows sets of defined questions, at the same time creating additional questions in response to the participant's comments or actions. This approach is suitable when the chances of follow-up interviews are limited. The unstructured interview process is when the researcher asks questions randomly depending on the context. Questions are open–ended and result in a conversation of that specified topic. This is useful when a researcher has to revise questions based on previous answers of the participants'. A researcher may use unstructured questions as a starting point for creating more structured questions (Savin-Baden & Major, 2013).

In this study, interviews are the main data collection tool. Unstructured questions were used as a preliminary means of data collection. The answers from the preliminary interviews were used to create open-end questions which were used in the semi-structured interviews. The open-ended questions allowed the participants' to express their views thus preventing limiting questions based on the researcher's views. Interviews were recorded for a more effective capturing of participants' views and interpretations (Walsham, 1995). The recording also helped in getting '*thick descriptions*' used during data analysis and as direct quotations from the participants. All recorded data was transcribed to extract concepts for the study. Extensive notes taken during each interview session helped in supporting the interview recordings.

Types of interviews include face-to-face interviews, telephone interviews, instant messaging interviews, emails, chat rooms and online spaces (Savin-Baden & Major, 2013). For the purpose of this study, face-to-face interviews, email interviews and instant messaging interviews were used. For the participants in the same location as the researcher, at least one face-to-face interview was conducted. Participants in other countries were interviewed via Google chat for the preliminary interview and email for the semi-structured interview. Table 2 indicates the profiles of the study participants.

Participant	*Gender*	*Position*	*Qualification*
P1	*Female*	*Senior Developer*	*PHD Computer Science*
P2	*Male*	*Developer*	*BSC Computer Science*
P3	*Male*	*Developer*	*BSC Computer Science*
P4	*Male*	*Developer*	*MSC Computer Science*
P5	*Male*	*Architect*	*BSC Computer Science*
P6	*Male*	*Developer*	*BTech Computer Science*
P7	*Male*	*Developer*	*BSC Computer Science*
P8	*Male*	*Developer*	*BSC Computer Science*
P9	*Female*	*Tester*	*BA*
P10	*Female*	*BA*	*BSC Computer Science*
P11	*Female*	*BA*	*BCom Information Systems*
P12	*Male*	*Project Manager*	*BSC Computer Science*
P13	*Male*	*Developer*	*BSC Computer Science*

Table 2: Participant Profile

3.5 Data Analysis

Analysis of data followed the grounded theory methodology concepts of coding and analysis. Data collected in an interview was compared to previously collected data using the process of constant comparison. Data analysed from previous interviews influenced subsequent interviews (Adolph et al., 2012). Three coding types can be used in the classical grounded theory methodology; open coding, selective coding and theoretical coding (Adolph et al., 2012). During each coding phase, it is important to note any thoughts on memos (Hernandez, 2009). This section provides an overview of the different processes that can be used during data analysis.

3.5.1 Constant Comparative Analysis

Comparative analysis is an approach that is used for generating theory (Glaser & Strauss, 1967). The principle of constant comparative analysis helps the researcher in identifying recurring concepts as a means

of validating facts. By using comparative analysis, the researcher can verify if what was initially said is correct (Glaser & Strauss, 1967). Another important use of constant comparative analysis in studies that follow the grounded theory methodology is to establish facts that can be generalised (Glaser & Strauss, 1967). Identified concepts are compared and if similar, renamed to facilitate the generalisation of these concepts. Glaser and Strauss (1967) identify the third use of comparative analysis as 'specifying a concept'. The process of constant comparative analysis is also used to validate the data with already existing theory. The researcher verifies the theory found in the data collection and analysis. The final use of comparative analysis which according to Glaser and Strauss (1967) should be the main goal of the researcher is that of theory generating. This process of theory generating is enabled by a comprehensive theory verification process. In this case, there is no need for proof to generate new theory. The process also helps to categorise concepts and identify relationships within the concepts. The researcher made use of constant comparison as recommended by Glaser and Strauss (1967) as a means of grouping similar concept and reducing the data categories for the study.

3.5.2 Open Coding

Strauss and Corbin (1998) define open coding as *"the analytical process through which concepts are identified and their properties and dimensions are discovered in data"* (p. 101). Open coding is the first level of data analysis as it generates emergent categories (Urquhart, 2000; Walker & Myrick, 2006) which become building blocks of theory (Glaser, 1992 as cited by Adolph et al., 2012). These broken data are often referred to as substantive codes or in vivo codes (Hernandez, 2009). Open coding involves looking at words, lines or a segment of the data collected through interviews, notes, documents and any other form of text in order to understand the meaning and subsequently create concepts (Glaser & Strauss, 1967; Walker & Myrick, 2006). Strauss and Corbin (1998) define a concept as a *"labeled phenomenon"* (p. 103). Glaser and Strauss (1967) refer to these concepts as indicators. Concepts in open coding are generated by asking 'generative questions' (Adolph et al., 2012) and then assigning labels or codes to the data (Charmaz, 2014; Urquhart, 2000) in order to note the similarities and differences in the data collected (Strauss & Corbin, 1998; Walker & Myrick, 2006). The allocated codes should be related to the data thus forming the backbone of the analysis phase. This forms the basic unit of analysis (Charmaz, 2014). When labelling concepts, one needs to ensure that the allocated label will not change meaning when other people interpret these concepts (Urquhart, 2000).

A researcher may have multiple concepts during the initial phases of open coding. When the process of labelling concepts has been completed, concepts that are closely related are grouped together to form a

more collective grouping called a category. Grouping the concepts helps in reducing the amount of data the researcher has to work with as well as helping in identifying the properties of the data and the common aspects (Strauss & Corbin, 1998). For every category identified, the researcher provides an interpretation. Data collected will be compared with previously collected data using a method called constant comparison analysis. This iterative method has been widely used in grounded theory methodology and other qualitative studies to formulate concepts and themes. The process of constant comparative analysis will assist the researcher in identifying possible patterns in the participants' views as derived from their wording (Walker & Myrick, 2006).

Open coding was used in the study to identify all the possible codes that resulted from the study. Codes were compared with other codes in order to classify similar concepts, differences in opinions, thus enabling the researcher to create patterns from the data (Savin-Baden & Major, 2013). Issues identified in previous interviews were used as reinforcement in subsequent interviews to validate a concept and understand the same concept from a different perspective (Adolph et al., 2011). The process of open coding was then concluded when the researcher began to see a theory that encompassed all the concepts identified (Walker & Myrick, 2006).

3.5.3 Selective Coding

The second phase of data analysis identified in the Glaserian grounded theory methodology is selective coding. Selective coding focuses more on the data that is around the core category which in turn allows for the process of verification of concepts (Walker & Myrick, 2006). Selective coding involves identifying the core category that best suit the other related categories. Grounded theory methodology helps in identifying the related set of concepts. A single story is built around all the codes that are similar (Adolph et al., 2012). The core concepts will then be used to build theory (Andrade, 2009). The researcher used selective coding to integrate data around the core category as recommended by Walker and Myrick (2006), with the results discussed in detail in the findings chapter.

3.5.4 Theoretical Coding

Theoretical coding is defined as the process of generating theoretical codes that integrate to form the core category. In this phase of coding, the researcher *"is simply detecting the relationships between two or more categories"* (Hernandez, 2009, p. 54). Relationships between categories are indicative in categories, sub-categories and main category (Hernandez, 2009). The researcher is concerned with *"integrating the data around a central theme, hypothesis or story to generate a theory"* (Walker & Myrick, 2006, p. 556) with the core category summing up the relationship of the all the other categories (Hernandez, 2009). It is

important that all theoretical codes emanate from the data instead of 'pet code' that the researcher may be having in his/her mind (Hernandez, 2009). Theoretical coding is dependent on open coding thus the two cannot be separated as they occur simultaneously (Hernandez, 2009).

Theoretical codes can either be implicit or explicit. Those that are explicit will emanate during data collection and analysis. Glaser provides possible different theoretical coding families which include degree family, dimension family, type family, models family and several others. The list of coding families is not stagnant. With more researchers making use of the grounded theory methodology, the researcher can potentially add to the existing list of coding families (Hernandez, 2009). Theoretical coding is an important phase that separates grounded theory methodology and other types of qualitative research. Hernandez (2009) identifies four ways in which a researcher can identify the emergence of theoretical codes. The first strategy is theoretical sensitivity which deepens the researcher understanding of the possible different types of codes. The second strategy is identifying concepts during open coding which *"can point to possible theoretical codes"* (Hernandez, 2009, p. 57). The third strategy, memo-writing helps in identifying relationships between the different concepts identifies. Lastly, the use of models can help in the possible theoretical codes that are emanating from the data (Hernandez, 2009). Some researchers in grounded theory methodology state that the use of theoretical coding is not always necessary for a study following the grounded theory methodology. In the study, however, the researcher followed Hernandez (2009) strategies of ensuring theoretical coding. These will be discussed in relationship to the findings chapter.

3.5.5 Memo-writing

The concept of memo-writing is crucial to a study that makes use of the grounded theory methodology, especially to the writing of the final research report (Pickard, 2007; Walker & Myrick, 2006). Memo-writing involves continuously making side comments during the process of constant comparison. The process of memo-writing assists the researcher to visualise the analysis process. As a researcher, the process of memo-writing implicates *"thinking aloud you are seeing, reading, feeling and understanding in your data"* (Pickard, 2007, p. 161). The researcher has the freedom to write the ideas that arise from each concept in accordance to their understanding. By writing down memos, the researcher may come up with different several ideas. Memo writing is an on-going process, which can result in changes of previously written memos due to newly identified information during constant comparison. From the written memos, the researcher is able to make note of the relationships, similarities and differences between data codes. For the purpose of this study, all the memos were documented on a Google spreadsheet, next to the identified

code. This facilitated a more efficient way of storing, searching and editing of each memo. The researcher did not, however, have corresponding memos for all the all data.

3.6 Quality Standard

In research, the trustworthiness of data collected is important to justify reliability, validity and generalizability of the data. Interpretive research is considered by some positivist researchers as lacking scientific proof because of the possibility of bias due to the researcher role. Although principles of reliability, validity and generalizability of the data are mostly associated with positivist quantitative research, interpretive researchers can apply the same principles by aligning them to suit an interpretive study (Andrade, 2009; Carcary, 2009). Researchers such as Sikolia, Mason, Biros and Weiser (2013) identify four terms equivalent to reliability, validity and generalizability which are more suitable in qualitative studies. Sikolia et al., (2013) associate credibility, transferability, dependability and conformability as measures of trustworthiness of data collected in qualitative research using grounded theory methodology. Regardless, some researchers maintain the use of validity, reliability and generalizability in an interpretive study (Carcary, 2009; Andrade, 2009).

3.6.1 Validity

The issue of validity is concerned with the credibility of the chosen research method in answering the research questions and the research design in explaining the case being studied. Validity is on the data collected as well as on the way the researcher interprets the findings (Carcary, 2009). Andrade (2009) refers to this as construct validity, which determines whether data has been evaluated correctly. Sikolia et al., (2013) associate validity with a concept called credibility, which refers to the possibility of reproducing similar results yielded by the data in a similar setting. Credibility can be ensured in qualitative research by means of data triangulation, using multiple sources of information (Andrade, 2009), sharing the emergence concepts and categories with the participants (Sikolia et al., 2013) and member checking and respondent validation (Andrade, 2009; Carcary, 2009). To ensure the credibility of the data, emerging concepts from the interviews were discussed with one of the Business Analyst (BA) and the one developer as a means to verify if they agreed with the way the researcher analysed the data.

3.6.2 Generalizability

Generalizability is the common term normally used in research to refer to the applicability of the theory generated in one case to another (Carcary, 2009). Sikolia et al., (2013) refers to this as transferability and also known as external validity (Andrade, 2009). In quantitative research, external validity refers to the extent to which the research instrument can be applied to a similar research (Andrade, 2009). Sikolia et al.,

(2013) define transferability as *"the applicability of one set of findings to another setting"* (p. 2). Appropriate demonstration of the research and the underlying research issues have to be noted in order for one to apply the concept of generalizability. Human nature and the way in which organizations behave allow for some cases to be generalized or the finding can be used as a starting point in a similar study (Andrade, 2009; Carcary, 2009).

3.6.3 Reliability

Reliability is a concept which is concerned with whether a study can be repeated. This can be difficult in qualitative research to imitate the exact case that was there in the original case (Andrade, 2009). Sikolia et al., (2013) correspondent reliability to dependability. Data collected should be reflective of the same pattern regardless of time, the researchers or the analysis techniques employed. A researcher may use the same participants, but may get the same exact responses because the second response will be based on the reflections from the initial responses. This, however, does not imply that the second response will be different from the initial response (Andrade, 2009). In qualitative interpretive research, the researcher can show that the data is real by being transparent about the results and how the results were obtained. In addition to this, an audit trail of the research process can be performed to confirm the data representation. This is an additional concept which Sikolia et al., (2013) refer to as conformability. This is especially applicable when using grounded theory methodology. Audit trail, negative case analysis and peer reviews will ensure the quality of the research (Carcary, 2009; Sikolia et al., 2013).

3.7 Ethics and Confidentiality

Ethical issues in qualitative research need to be addressed as this form of research results in answers that are narrations of real life incidences that may have ethical implications on individuals participating in the research (Savin-Baden & Major, 2013). Dresser (1998) as cited in Savin-Baden and Major (2013) indicates the importance of ethical considerations in qualitative research to ensure the best interest of all participants. Appointed individuals help in ensuring the protection of participants from *"physical harm…, psychological harm..., social or economic harm"* (Savin-Baden & Major, 2013, p. 323). To ensure compliance to ethical consideration in this study, the legality of the research was reviewed by the University of Cape Town Ethics Committee.

Ethical issues of particular importance to this study include informed consent, anonymity and confidentiality. Informed consent implies that the participants are aware of the purpose of the research and are comfortable in providing answers to the questions that are asked during data collection (Pickard, 2007). The designated ethics committee ensures all participants have given informed consent which is best done

by *"a formal informed consent form that is read, understood and signed by all research participant"* (Pickard, 2007, p. 74). The participants must be informed of the potential risk that may arise during or after the research (Savin-Baden & Major, 2013). In this study, all participants signed a consent form which was first read to them and given to them to read for more understanding and then provide a signature. The form included the purpose of the study, data handling and information on the possibility to withdraw from the study at any time and their independence in answering questions. The company intended for the case study was aware of the purpose of the study and the use of all collected data. The company vice president signed a form in agreement with continuing to use the company as a case study.

The ethics committee ensures that a study maintains the anonymity of all participants. Anonymity means that *"nobody knows who the participant is"* (Pickard, 2007, p. 77). Anonymity in research is not always possible as the researcher in some cases, has to meet the participants especially during interviews, observations and focus group (Pickard, 2007). The same case applies in this study as the researcher had to perform one-on-one interviews with all the participants. However, the researcher can ensure confidentially to the participants. Confidentiality in research means that *"nobody will be told the identity of the participant"* (Pickard, 2007, p. 77). The researcher can either use pseudonyms or codes to make reference to a participant or the organisation used in the study but this may not always be useful in some scenarios such as academic institutions with specific roles assigned to a specific person such as the head of the department. It is important to ensure that the researcher upholds to the given promises of anonymity and confidentially as to avoid causing harm to the participants (Pickard, 2007). In the current study, pseudonyms are used for both the participants and the company.

3.8 Summary

In this chapter, the researcher presented the Glaserian grounded theory methodology that was used to collect and analyse data for the study. Although the grounded theory methodology is not embedded in any research paradigm, the researcher indicated the use of the interpretive philosophical stance and the reasons behind the choice of paradigm were stated. The concepts of quality standards were discussed, in particular how grounded theory methodology quality standards can be mapped to validity, generalizability and reliability. The chapter ends with information regarding ethics and confidentiality for the study. The next chapter will focus on the analysis of data. From the data analysis; concepts, sub-categories and categories were identified. The sub-categories and categories will be discussed in the following chapter.

Chapter 4

Findings

4.1 Introduction

The main purpose of the study was to identify the core challenge software development teams face when trying to integrate security and privacy requirements into the development life cycle of a mobile application. The study was prompted by the security and privacy issues that end users of mobile banking apps, in particular, experience. Data collection and analysis were carried out using the classical grounded theory methodology. The research design was discussed in the preceding chapter. This chapter presents a summary of the findings of a case study of an agile team that develops mobile banking applications.

Data collection and analysis was performed in an iterative manner. As the interviews were being conducted, the researcher made note of concepts that were emerging from the data. Each interview was transcribed into a Microsoft Word document. The initial interview provided the researcher with pointers to possible questions for subsequent interviews and to who should be interview next. In order to get a different perspective, another developer was interviewed next. As with the first interview, the subsequent interview was transcribed, broken down and data codes were identified and labelled. The researcher then began the process of constant comparison to allow for the identification of similar labels in order to re-labelling. In some case, labels such as *complexity*, *misalignment* and *ignorance in users* were taken from the words of the participants. For all the interviews that followed, the same process was taken into consideration. The remaining interviews were handled in the same way.

During this initial process of data collection and analysis, the researcher documented thoughts on the concepts that were emerging from the first round of the analysis. When the process of coding had been completed, the results of the data analysis were documented in a Google Spread sheet. The column which contained the answers to the question *'What concepts does this incident indicate?'* had over 30 open codes. Selective coding of the data revealed the concept of Misalignment as the core category. Different types of misalignment were identified. Although the classical grounded theory methodology conditions on identifying a core category, a second category, Nature of the domain emerged in the study. This concept represented an important relationship to Misalignment. In the sections that follow, it is important to mention that the researcher will be making reference to Table 2 in Chapter 3 for the purpose of cross-referencing

what was said and who said it (participant) between these two categories, Misalignment and Nature of the domain.

4.2 Case Study Profile

Company X is a multinational company that specialises in electronic payments for financial institutions, retailers and payment processors worldwide. However, for the purposes of this study, the focus was on the team that builds the mobile banking application. The firm provides software applications for bill payments and transfers. The team that builds the mobile applications draws its members from two different countries. The team follows an Agile-Scrum methodology with the Product Owner/Business Analysts (BA) playing the role of the 'client'. The developers are involved in the programming while the testers test the software based on the requirements documented by the BA.

4.3 Misalignment

This section discusses the different types of misalignment as identified in the study. Table 3 presents the challenges that the team may encounter when integrating security and privacy requirements into the mobile banking apps. Before exploring the different types of misalignment, it is necessary to define misalignment. According to the Oxford dictionary, misalignment is *"the incorrect arrangement or position of something in relation to something else"*. In IS research, the term 'misalignment' has been applied to a number of situations or states, where elements that are supposed to work in unison do not do so.

Category	***Definition***
External Misalignment	*External misalignment occurs when the software development processes conflicts with any other elements that are external and out of the control of the development team such as, the customers, regulations and third party applications.*
Role Misalignment	*Role misalignment occurs between specific roles, such as developer and tester misalignment.*
Skills Misalignment	*Skills misalignment occurs when the team's current skills do not match those required by the workload so that there are mismatched responsibilities and incorrect implementation.*
Requirements Misalignment	*Requirements misalignment occurs when there are conflicting issues between the security requirements and the general system requirements.*

Table 3: Misalignment Categories

Table 4 presents the frequency with which the different sub-categories of misalignment were found to occur. Occurrences relate to the number of times the category appeared in the data analysis column on categories. Recurrence refers to the number of participants that indicated the category. External misalignment occurs most frequently than any other types, with twelve participants indicating that external entities, for example, customer requirements, standards and guidelines, regulatory requirements and third party applications are the source of some of the challenges when the team is integrating security and privacy requirements. Requirements misalignment is the second most frequent area of misalignment, followed by skills misalignment, and lastly by role misalignment. Four of the participants indicated that the differences in roles created challenges during the integration of security and privacy requirements to mobile banking applications.

Category	*Occurrence*	*Recurrence*
External Misalignment	*49*	*12*
Role Misalignment	*6*	*4*
Skills Misalignment	*28*	*11*
Requirements Misalignment	*32*	*8*

Table 4: Misalignment Categories Statistics

The sections that follow provide more details on the team's view on the different forms of misalignment.

4.3.1 External Misalignment

External misalignment occurs when the software development processes involving security and privacy requirements integration conflict with other elements that are outside the control of the development team. External misalignment includes misalignment with security and privacy standards and guidelines, regulatory requirements, third party libraries' and end user requirements. Although a number of external factors have been identified as the source of misalignment during the software development process; for the purposes of this study, the focus will be on the four facets identified above. The main reason for this choice is that these emerged as the prominent sources of external misalignment during data collection and analysis.

CUSTOMER REQUIREMENTS

Customers in this study refer to the banks that acquire the mobile banking solution provided by **Company X**. Misalignment with customer needs can result in a dissatisfied customer. However, customer requirements can result in security hazards if implemented as is. In one example, the customers wanted *'Web Views'* inside their banking application.

> ***"Customers want web views but this is a security issue."*** *P1*

The security standards and guidelines offered by 'Now Secure', a company that specialises in mobile security, state that the use of *'Web Views'*, especially in an Android application, can result in exploitable vulnerabilities. *'Web Views'* do not usually have a URL address bar, which is supposed to indicate the authenticity of the website. If a user clicked on a fake website and entered his or her login credentials, he or she would have exposed their personal details. This can compromise security and expose the user to dangers such as identity theft.

A customer may base his or her requirements on the solution that they currently have in place. In some cases, when acquiring additional security and privacy measures, a customer may want a new solution that aligns with an existing solution. This is a challenge for the company providing the mobile app as they cater for many customers who have different requirements. It is difficult to try and cater for each and every customer's requirements. One of the BA described this difficulty as follows:

> ***"So some customers prefer RSA, and they say they want [the] RSA to work[;] so currently we don't have it – we say we have OOBA, and they say [,'No'], they want RSA and they want Trusteer – and also we have these products we want to integrate [with]."*** *P10*

These customers have already established a relationship with their security application providers. For the customers, it is important that a new mobile banking solution does not conflict with the solutions that they have purchased previously. This matter requires that the BA ensures that there is no conflict between the existing and the new solutions. If a financial institution needs to change provider, such a move would incur additional costs because it would take time to get to know and be comfortable with a new provider. The second BA who was interviewed added:

> *"Typically the customers we work with already have a relationship with one or more vendors. We have noticed that a couple of our customers will have multiple security vendors – they will have someone who specialises at login and authentication for example, and another one that does authentication and transaction release."* P11

Consequently, a customer's requirements are always taken seriously. A tester indicated that it was necessary to focus a great deal of attention on the functional requirements, which in most cases are prescribed by the BA and the customer.

STANDARDS AND GUIDELINES

It is important to note that, in the case of mobile apps security, the team may make use of a tool from a third party or it may build the security components in-house, or, do both. According to the BAs, there is no set of guidelines, either internal or external, which can assist a banking institution in the selection of a security vendor. The BA, who was interviewed first, had been involved in acquiring a third party tool which would ensure device security. This BA said:

> *"Then I think [that] for [the] customers themselves – it makes it confusing for what they should be supporting…because each vendor has got a different set of authentication mechanisms. And there is no clear guideline on what vendors they should select or support. So certainly, within the industry there is work that should be done to identify which vendor someone would [sic] choose."* P11

Participant 10, the other BA, agreed with the above statements, adding that this can result in additional costs and can delay the completion of a project.

> *"Across the industry- there are a number of reputable security vendors out there- for us as an intermediary vendor it's quite difficult for us to then support all the multiple vendors out there. We actually had to analyse them [vendors] and build some sort of a checklist of what we want."* P10

There is need to focus on guidelines for the mobile application domain. It is important to note that the mobile domain is unlike its most compared to platform, the desktop PC. Although there may appear to be similar, the mobile domain has characteristics which make applying standards and guidelines for desktop problematic. One of the BAs acknowledged the lack of security and privacy guidelines and the need to align the guidelines existing for desktop onto the mobile platform.

> *"We took an exercise last year to review some of these guidelines in the MESA region where they were trying to push out for mobile banking security guidelines. So we got an opportunity to review those earlier and provided some comments, and the outcome from our perspective, is that they had not really taken into account a stance to understand the specifics that change on the mobile device. It is not the same as the desktop device you know- I think they were much focused "let's do whatever is on the desktop on the mobile device'."* P11

REGULATORY REQUIREMENTS

Regulatory requirements refer to the laws and regulations passed by governments. Most governments promulgate laws and regulations that govern the use and storage of personal information; of particular interest, are the ways that these legal instruments prescribe how parties other than the owners of the information deal with personal information. Regulatory requirements affect the development process. Unfortunately, these regulations are not always given the necessary attention during the development process and challenges can arise when a development team tries to integrate government regulations into the process of mobile software development. One developer spoke about the way regulations can give rise to alignment problems.

> *"Regulations regarding privacy, especially privacy because, there are quite a lot of regulations – many countries have different regulations from other countries… You need to have a view of where you are going to deploy and figure out what are the regulations for those areas and research…"* P4

When a team is trying to align regulatory requirements in a mobile banking application, it is important for the stakeholders to be aware of the regulations which apply in the countries where the mobile apps will be used. The team must also take the necessary precautions to prevent an inadvertent violation of these rules and regulations which would compromise the security and privacy of the mobile banking app. A developer also explained that:

> *"Balancing security requirements and user requirements with security requirements and regulatory requirements – [the] user get [sic] frustrated when they have to get [to] do something by using multiple flows."* P5

The above statement indicates the added complexity to the end product due to regulatory requirements. This can affect the usability of the mobile banking application.

THIRD PARTY LIBRARIES'

The use of third party applications brings with it a number of alignment challenges into the software development lifecycle. From the analysis of the research data, it emerged that a third party application may be added into an existing software product. In this case study, the organisation under study integrated a third party security application as one of the ways for ensuring the security of a mobile banking application. Aside from the benefits of using a third party application, that will be discussed in section 5.3.1, challenges can still arise when attempting to integrate a third party security application into a mobile banking application.

One of the developers mentioned that misalignment could occur because internal organisational security policies can conflict with those of the security vendor. Participant 1 stated that one of the challenges encountered in the initial phases of integration the third party application process was the need to find ways of ensuring that the internal policies were not violated:

> ***"Understanding how to use the third party application in such a way that it does not violet our privacy and security requirements, want to do."*** *P1*

When third party applications are developed, it is impossible for the developers to be knowledgeable of the regulations of all the countries where the app may be used and with the internal policies of the companies which might employ the app. Conflict between regulations and internal policies may only become evident after the process of integrating the component is completed. The way a third party may decide to employ an app may conflict with the use envisaged by the original developers. If a third party application is misused, there may be a clash between the development team and the third party vendor. One developer stated the following:

> ***"They are not happy with our implementation."*** *P1*

4.3.2 Role Misalignment

Role misalignment occurs between the different roles in a development team. The roles found in the team are easily distinguishable yet connected. In a typical Agile–Scrum environment, there is a high degree of collaboration within the team. In these circumstances, it is essential that the team members understand the tasks assigned to the different roles. This will enable each member to see where his or her role fits in with those of the other members of the team and each team member will know how his or her role complements and supports the other roles. One developer summed up the arrangement, as follows:

> *"If the requirements are clearly stated, the testers do not need to know the code."* P3

This statement indicates a misalignment in roles. The BA documents the requirements. If a requirement needs to be clarified, this should be done by the BA, not by the developer. If the developer does this, he or she will only be able to explain the requirements on the basis of what has developed, instead of the requirements as they were documented by the BA. Mistakes can result from differing versions of the requirements. One developer pointed out the consequences of this type of misalignment.

> *"The first thing is, if you do not have good alignment on the people who do the research on the privacy requirements and [the] people who want to code, you are going to miss stuff."* P4

This example of role misalignment can result in errors and rework. All the team members have to be involved in co-ordinating the team's tasks and be involved in all the processes of the SDLC. This is how one of the developers' described the arrangement:

> *"Developers need to follow the requirements, and [the] developers and [the] business analyst need to align…quality assurance need to have a good understanding of the requirements…so when the developers starts to look at the requirements in parallel, the quality engineer needs to look at the requirements and say ['H]ere is what I am going to test on the software, and [the] tests I am going to run to verify that the requirements are met['].* P6

Lastly, role misalignment can occur if the team does not have the right people in a team. Security is an area that requires specialist knowledge. Some of the participants felt that possession of a university degree in software engineering and experience as a developer did not necessarily mean that one had an in-depth understanding and knowledge of security and its difficulties. However, it is important, when dealing with security requirements that the team members are well versed in the ramifications of security. The project manager made the point:

> *"The first thing is ensuring that someone with security expertise is on the project."* P12

The project manager stated that there was no dedicated security expert who was part of the mobile team. When there is the need, someone who is knowledgeable about mobile security is usually assigned to the team. The company did not have enough security experts so that it could assign one to a specific project throughout the duration of that project.

> *"We do have security resources, but two to a team of twenty is too little, also they can not be involved at every stage."* P12

4.3.3 Skills Misalignment

Skills misalignment occurs when the ability of a team member does not meet the level of competency that is necessary to carry out a specific role efficiently. Skills misalignment can result in idle time, errors and in responsibilities being assigned inappropriately. In the current study, one simple task was executed with a number of errors because the team lacked the knowledge necessary to configure the third party application correctly so that it would work with the mobile application. One of the developers told the researcher that as a result, the task took much longer to complete than was initially planned.

> *"Figuring [out a] configuration so it does what [you want it to do,] took more time than we expected."* P1

The skills deficit in the areas of security and privacy is primarily a consequence of security and privacy education not being included in a software developer's curriculum. Most developers learn how to write code. Security skills are additional proficiency that is acquired through experience.

> *"We have a fairly inexperienced team that are not well vest [sic] in secure coding or standards, so you need to get them in training, you need to have code review cycles. [A]nd I think for an inexperienced team that is sometimes challenging, because they either don't know or the pressure on them is so tight that they don't have the time to do all this [sic] things[;]but this is important, especially in the mobile banking, to do these things as part of your coding cycles …"* P4

The introduction of security and privacy requirements into the software development process increases the complexity of the task. One developer stated that there was also the need to look at additional requirements such as the secure storage of data.

> *"I think security and privacy requirements increase the complexity of the application."* P2

Participant 5 supports the issue around added complexity of the development process as a result of the inclusion of security and privacy requirements. It is not enough for a developer to be able to write code, he or she may need to include cryptography and encryption when creating a mobile banking app that is secure.

> *"You need to think deeper [sic] into it – you need to think about a lot more complicated topics, like encryption, and everything you look at takes longer."* P5

Complexity in security and privacy requirements is not only an issue for developers. Testers are affected by the complexity of security testing, especially, if they lack the appropriate skills and experience. A tester stated that the team relies on the help from the developers to carry out the more technical aspects of testing.

> *"Security on its own is very complex. We needed the help of the developers to set up the testing scenarios. Even after that we could not do the actual scenarios like testing on a jail broken device as we are not allowed to jail break a device. We have limited number of phones- thus the risk was not worth it."* P9

This statement points to a misalignment between the tasks assigned to the tester and the skills that the tester actually has. It is expected that the tester can carry out various types of testing, both functional and non-functional. Similar misalignment can be seen between the theoretical and the actual skills possessed by BAs. One BA spoke about how they relied on the developers to document the process of integrating security requirements into a mobile banking app. It is the responsibility of the BA to document all the requirements on behalf of the development team. The BA did not have enough knowledge on the way security implementation is supposed to work.

> *"You need to ensure that the passwords are masked and that it [sic] is logged in a trace file. A lot of that you need to understand, how the app communicates with the backend for example, the API is secured and any connection [that is] made is encrypted."* P10

Skills misalignment also characterises the way that the development team treats established standards and guidelines and regulatory requirements. Participant 4 commented on the team's inadequate knowledge of security practices and how they lacked the appropriate experience in dealing with them. Another developer said that the university curriculum teaches developers how to write code that works, but very little attention is paid to non-functional requirements such as security and the standards and guidelines. It is a problem for developers that regulatory requirements are usually written by lawyers. A BA revealed that knowledge of the law is essential for understanding most government regulations on security and privacy.

> *"Regulations are quite difficult to read… We need a person to do this… to distil the laws so that it [sic] is applicable and tangible [sic] to us… [Y]ou need a lawyer to be able to understand."* P11

4.3.4 Requirements Misalignment

Requirements misalignment occurs when security requirements and the general system requirements come into conflict. These requirements can be functional or non-functional. Security and privacy requirements

are categorised as non-functional. Although there is a need to pay attention to the users' functional experience, security and privacy requirements are becoming priority to the software development process, especially when considering banking applications. The addition of security and privacy requirements extends the time needed to complete a project. Customer expectations need to make allowances for the complexity of the task of adding security and privacy requirements.

> ***"It should be a fairly easy exercise because all the integration has been done. We underestimated this and it takes longer than expected. This will have a cost implication and project schedules will have to be re-looked at."*** *P13*

In some cases, the incorporation of security measures can mean that the user has to perform additional processes or steps before he or she can get into the application. This may affect the usability of the application.

4.4 Nature of the Domain

The 'nature of the domain' denotes aspects of the mobile ecosystem which are accepted by the IT industry. 'Nature of the domain' includes characteristics such as fragmentation, lack of control and level of security awareness. These characteristics apply to the mobile application domain. Table 5 provides definitions of the main categories of 'nature of the domain' that were identified in this study.

Category	***Definition***
Lack of Control	*Lack of control refers to the inability to prevent an event or an outcome due to the volatile nature of the mobile domain.*
Fragmentation	*Fragmentation refers to the lack of integration in programming languages, mobile operating systems and handsets.*
Security Awareness	*Security awareness refers, first of all, to the software development team members' knowledge of and practices relating to the implementation and testing of a security feature or component; it also denotes the extent to which the customer takes responsibility for preventing security vulnerabilities.*

Table 5: Nature of the Domain Concepts

Table 6 indicates the number of times that the various sub-categories, which constitute the concept 'nature of the domain', were found to have caused challenges in integrating security and privacy requirements. Issues around lack of control occurred most frequently, there were 39 such occurences. 10 participants referred to the seriousness of this challenge. Fragmentation occurred 15 times; six participants mentioned this challenge. Last of all, five participants referred to security awareness issues 10 times.

Category	*Occurrence*	*Recurrence*
Lack of Control	*39*	*10*
Fragmentation	*15*	*6*
Security Awareness	*10*	*5*

Table 6: Nature of Domain Categories Statistics

The section that follows provides a summary of the categories of the issues that fall under the concept the nature of the domain.

4.4.1 Fragmentation

Fragmentation is one of the biggest challenges that face the mobile application development team. Fragmentation can be viewed from a number of perspectives. From the data, the researcher identified the following areas where fragmentation occurs: differences in programming languages, device variations, mobile operating system variations and fragmentation within third party solutions. The most common form of fragmentation is device fragmentation. This is as a result of the differences between device vendors and operating system providers. The BAs are responsible for documenting the requirements of the various devices; the developers create applications in which the various components or functions operate smoothly, and the testers' check how the applications work on various devices.

The fragmentation in programming languages is a major area of concern for the developers. Most of the developers mentioned the differences between the programming languages. Android applications are developed using Java, while iOS applications are developed using Objective C. Blackberry applications can use Symbian or Java, depending on the operating system running on the device; Windows applications use C#. Ideally, a developer will specialise in one platform, for example, iOS. Nonetheless, this does not happen often, instead they have to switch from one programming language to another. A developer who is hired to develop a mobile application for **Company X** will be expected to be adaptable and be able work on any platform.

Fragmentation which arises from differences between devices poses a challenge in the process of developing a mobile application. The addition of security to the requirements makes it even more challenging. Mobile devices range from phones to tablets. These devices come in different sizes, with different screen resolutions and which support various versions of an operating system. One of testers spoke of this type of fragmentation challenge as a combination of operating system fragmentation and the difference in the size of the devices:

> *"If we look at [A]ndroid – there are different flavours of the [A]ndroid phone – each with different sizes and OS capabilities. This makes [the] testing of mobile apps difficult because you may not cover enough device coverage. Also, we do not have all the devices to test on [sic]."* P9

This statement points out that it is not financially feasible to buy one set of every device on the market, for development purposes. The disparity in mobile devices not only affects in-house development of security features; it is also a problem when the company purchases a third party security application. The vendor should make sure that their application supports different mobile devices.

Security and privacy issues involving mobile applications are usually treated as matters of device and/or application security. It is, however, difficult to find a single solution that addresses both these issues. This can force the company to spend more money in trying out the different solutions, developing different solutions for both device and application security otherwise the team can select what they deem is the most important. One of the BAs indicated that:

> *"There isn't a one size fit all…there isn't a single vendor or product that ensures complete security around mobile app."* P10

If a company decides to rely on a single form of security mechanism, it is important that the choice is the right one. For **Company X**, device security was considered more important; hence the team's choice was to work on the integration of a third party application to ensure device security. Unfortunately, device security on its own does not mean that the application is secure, as one of the BAs pointed out:

> *"For mobile – a lot of it is tied to firstly, device security because you want to protect the device, and that is because your device is at risk [from]… other apps and malware running around and then the[re] is security where we secure the application , in that once…it's a secure device and the app is there."* P10

4.4.2 Lack of Control

Lack of control denotes the inability to prevent an event or outcome. In this study, lack of control refers to factors that the development team have no control over, such as the volatile, unknown mobile software environment and the threat of constant attacks on the systems. There are other challenges in the process of the mobile application development that the team will try to control, but they may not succeed in controlling them fully. Human error is a constant problem. In software development, human error is inevitable. One small change to add a function or fix a defect can result in the introduction of other vulnerabilities. Some of these errors, especially if there are in the operating system, can exert a ripple effect if they are not identified early on. One of the developers gave an example of a bug in Android 4.4 which affected the sending and receiving SMSes. Human error cannot be eliminated totally by setting up measures or procedures to control it.

> *"All things in our world were made by human[s] , we all can make mistakes."*
> *P3*

Another challenge identified by the team was that security was 'a moving target'. This makes it difficult to have complete confidence in the security of an application. One question that was asked to the participants was about measures that would ensure that a particular application will always be secure. None of the developers was sure that this was possible. Issues around the volatility of the information technology space, the constant threat of hacking, open platforms and the rapid changes in the industry make it almost impossible to create a solution that is future proof.

> *"Security is not an easy topic. Staying secure is a moving target."* P4

In addition to the pressures exerted by relentless change and a volatile environment, there are so many players in the mobile banking application market. It is impossible for the development team to make allowance for the activities of all the stakeholders. Some of the participants mentioned the challenge posed by hackers. Hackers are a real threat to the software development process. For some, hacking is just the excitement of a challenge; others do it for financial gain. Some responses expressed the feeling that the more secure the application is, the more likely it is that it will be hacked. The challenge then becomes one of staying ahead of the hackers without compromising the security of the application.

> *"It's a very active space. There is [sic] constantly people sitting trying to break systems."* P6

The security of the third party applications was raised by the participants. As a company which often integrates software designed by another company into its own applications, there are a number of factors, for example, security, that must receive attention. When a company purchases a third party application, the team may not be aware of security and privacy issues that are associated with that application. Factors, such as limited time and the gaps in the team's skill set, may prevent or hinder exploration of the third party application. Since the vendors are known to provide security application, there is an expectation that the security application that they provide will indeed be secure.

> *"We didn't test that the Trusteer libraries was [sic] working correctly. You have to make some kind of assumptions and [have] faith that [the] framework you are paying for is going to work properly. You don't know whether it's secure – you don't know the ends [sic] and outs. It's like paying for a service. It's one of those things, you have to make an assumption based on the fact that they are supposedly experts in the field. Yes – there is a risk – but the risk of not having it is much… [greater]."* P6

4.4.3 Security Awareness

Security awareness refers to what individuals and organisations know and take into consideration when developing software or making use of third party software. Security awareness is a major concern especially when the development team are expected to implement security into the mobile banking applications. Both users and the development team need to pay attention to security and privacy requirements. The correct use of the mobile applications can prevent security. The main problem is that people have limited knowledge and understanding of complexity of the areas of security and privacy in software.

> *"People are not aware of security."* P3

Information from the data analysis revealed that security awareness is a major concern, especially for the business analysts, testers and the developers. The more senior developers are more aware of the issues related to security and privacy. The interviewee quoted earlier in one of the above sections added that security and privacy are not usually part of the university computer science curriculum. Thus, it is unlikely that the developers, especially the junior developers to be conversant with security and privacy topic.

> *"People on the ground, like the developers are not that familiar with security and they don't have much detail and understanding [of] what it means to build something that is secure."* P12

Whatever the level of team's security awareness, their most important concern, especially in the case of the developers, is getting the job done. Greater attention and effort are directed at getting the functional requirements right and developing a solution that is based on their understanding of the requirements.

> *"Application security isn't something that is [at the] top of [the] mind for software developers – you just focus on the application that you are building and how you are going to get it out there to the user and how can I improve the user experience."* P8

The lack of security awareness of the end users is a major challenge for mobile application development. This is evident from the interviews. Several of the participants indicated that the end users' ignorance is a major issue. Mobile users' main concern is getting the application onto their phones. They are oblivious to the possible impact of downloading applications from unverified sources, or of automatically granting the permissions that the application requires. Security and privacy issues can be introduced by adding any application which does not come with a mobile phone. In addition to this, users eliminate software constraints specified by the operating system provider by means of 'jailbreak' iOS devices or 'root' their Android phones. This will increase the possibility of their mobile devices being attacked. Users may not be aware of the detrimental effects of 'jailbreaking' or 'rooting' their phone.

> *"It's basically the nature of users. Users can install any app without exercising any diligence. Users do not check permissions on the app – users are quite quickly [sic] to follow trends, quick to root their phones and all those things, so that it is quite difficult for us, I would say."* P7

4.5 Summary

Integrating security and privacy requirements into a mobile banking application depends on securely written coding, and/ or, the correct integration of a third party security application. A number of challenges face any team which undertakes these tasks. This study has sketched the nature of domain and the various types of misalignment, both external and internal, which complicate the integration of privacy and security requirements into the process of developing a mobile banking app. There may be misalignment between the team and external entities; misaligned roles within the team; misalignment between the skills necessary for carrying out various tasks and the skills actually possessed by team members; and misalignment of the security requirements and the system software requirements. Figure 5 shows the theoretical model developed from the data collected and analysed by the researcher.

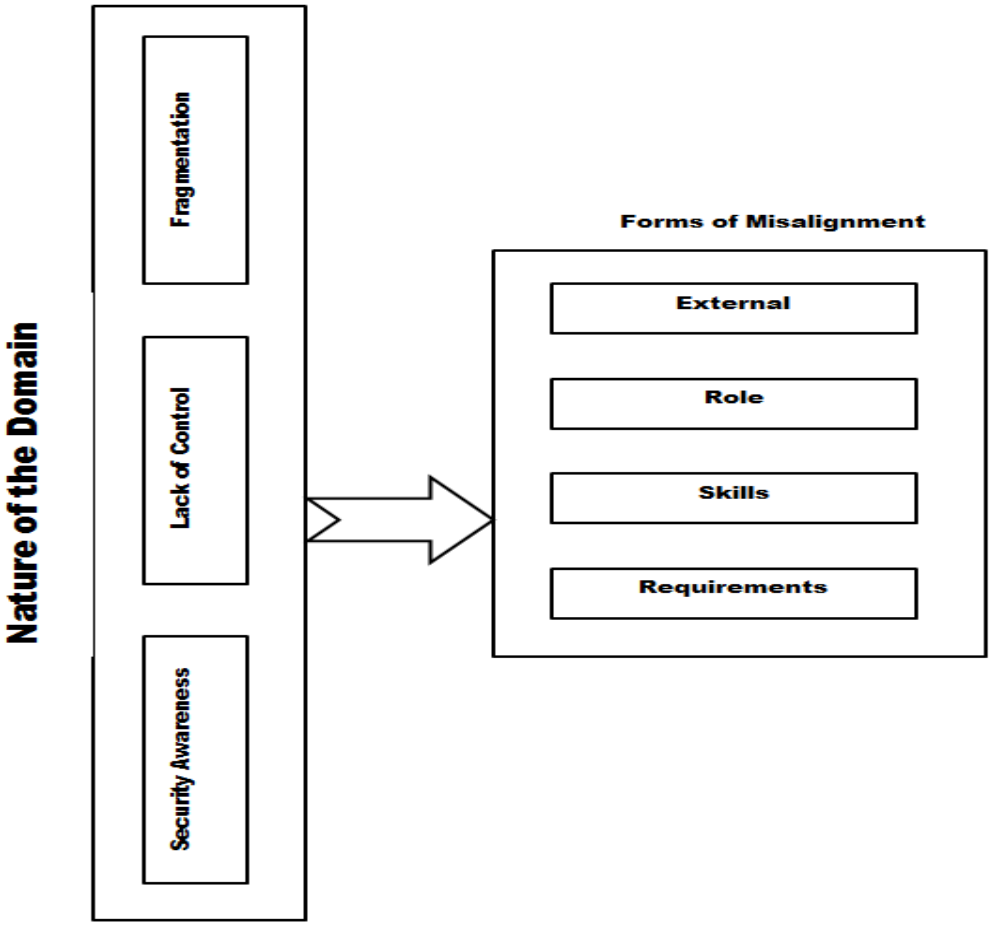

Figure 5: Theoretical Model

This study has identified the nature of the domain as characterised by fragmentation, the lack of control and security awareness. Fragmentation in mobile application development can be affected by device fragmentation, mobile operating system fragmentation and programming language fragmentation. Security awareness is indicated by gaps in or lack of awareness on the part of end- users and software development team members.

The following chapter will compare the findings of this research study with the literature on software development. This will be a more extensive literature review than the one presented in chapter two. The purpose is to conceptualise the challenges faced by the mobile application development team in integrating security and privacy requirements.

Chapter 5

Discussion

5.1 Introduction

This chapter makes a comparison on the concepts that emerged from the study with the existing literature on misalignment and the nature of the domain. The main purpose is to understand if the identified concepts have received attention in the scholarly literature and where they can fit in literature. The literature covered in the preliminary review differs from the literature that will be discussed in this chapter. The preliminary literature review dealt with issues regarding software development and secure software development.

In understanding the challenges that are encountered when attempting to integrate security and privacy requirements, the interviews with the BAs, developers, testers and the project manager were crucial to getting the holistic view of these challenges. Some of the challenges that came to light during the interviews are not new to the domain of software development. The results from the process of data analysis indicate that the concept that is named 'misalignment' is a serious obstacle to the successful integration of security and privacy requirements into the mobile application development lifecycle. The forms of misalignment are influenced by the nature of the mobile application domain for example skills and security awareness.

Volkoff and Strong (2010) made a contribution to IS grounded theory studies on their work on misfits to ERP systems. Their study resulted in six types of misfits in ERP systems. The findings on different types of misalignment in the current study are a similar in style to the outcomes of the study by Volkoff and Strong (2010). The two studies are examples of the type family theoretical coding as indicated by the forms of misalignment and types of misfits.

5.2 Nature of the Domain

This section deals with the literature on 'the nature of the domain' in relation to mobile applications. In this study the term 'nature of the domain' refers to the characteristics of the mobile application development domain and of the environment in which a mobile application is developed. The three main facets of the 'nature of the domain' that were identified during the process of data analysis are: 'fragmentation', 'the lack of control' and 'security awareness'. The concept 'fragmentation' emerged as important when aspects such as the roles in the SDLC, mobile operating systems and programming languages were examined. The concept of 'lack of control' describes the volatility of the mobile application domain. Lastly, the concept of 'security awareness' relates to the knowledge on security and privacy standards and guidelines,

implementation of security requirements and the diligence in accessing and using mobile apps. Security awareness affects the end users and IT personnel. The literature on 'fragmentation', 'lack of control' and 'security awareness' will be discussed in the sections that follow.

The concept of fragmentation refers to differences in mobile application platforms (Hammershoj, Sapuppo & Tadayoni, 2010; Joorabchi, Mesbah and Kruchten, 2013). In relation to mobile application development, Joorabchi, et al., (2013) identify two types of fragmentation, namely, fragmentation across platforms and fragmentation within platforms. Currently, native mobile applications can run on Android, iOS, Windows Mobile, Nokia/OVI and Blackberry. Android and iOS are the main mobile operating systems (Hammershoj et al., 2010). The existence of so many operating systems creates challenges when development team members begin gathering requirements and during the processes of development and testing. Each platform has its own dedicated tools, APIs, programming languages, standards, guidelines and particular user experience. Within a platform, fragmentation exists on the levels of type of device and operating system versioning. Different devices which run on the same platform can have different properties, for example, speed, screen resolution and size. Operating systems can also run different versions (Joorabchi et al., 2013).

In mobile application development, the lack of control affects the development platforms, in particular, the need to keep up with the ever-changing mobile environment. Unlike Android, the iOS and Windows operating systems are closed systems. Android is 'open' because it allows manufacturers, for example, Samsung, Sony, HTC, and LG, who use the Android operating system to introduce modifications that fit in with their brand. The Android platform's openness to changes made by mobile device manufacturers can result in various misalignments within the same platform. This becomes a challenge for the teams developing mobile applications (Hammershoj et al., 2010).

5.3 Misalignment

In this section, the nature of the domain and the relationships with misalignment will be explored in more detail. Misalignment arises when the intended purpose or design is somewhat conflicting with the real outcome. The concept of alignment in IS has been explored especially in IT-Business alignment with researchers such as Chan and Reich (2007) providing an outline on researches related to IT-Business alignment and the work of Luftman and Kempaiah (2007) focusing on the levels of IT-Business alignment. The concept of alignment has also been employed in the field of software development as a way of understanding and addressing issues that hinder or prevent the correct alignment of development processes and testing (Dhaliwal, Onita, Poston & Zhang, 2011; Zhang, Dhaliwal, Gillenson & Stafford, 2013; Zhang, Stafford, Dhaliwal, Gillenson & Moeller, 2014; Mbekela & Brown, 2014; Onita & Dhaliwal, 2011). In

relation to IT security research, issues of alignment between security policy and IS plan (Doherty & Fulford, 2006) and issues of alignment of business objectives and security objectives in order to be able to quantify security in terms of business objective (Fruehwirth, Biffl, Tabatabai, & Weippl, 2010; Frühwirth, 2009).

The concept of alignment is complex, especially in relation to IT as it is quite fragmented and relates to different areas. In order to achieve appropriate alignment, it is important to ensure that the "*focus is on specific components of alignment rather than on the overall alignment"* (Dhaliwal et al., 2011, p.324). The sections that follow provide the different forms of misalignment identified in the study in relation to the existing literature.

5.3.1 External Misalignment

In the previous chapter external misalignment was defined with reference to software development processes where various elements come into conflict because they are outside the control of the development team. In Chapter 4 the following areas or types of external misalignment were identified: customer requirements, standards and guidelines, regulatory requirements and third party libraries.

CUSTOMER REQUIREMENTS

The analysis of the research data revealed the extent to which customer requirements drove the software development process in **Company X**. For the BA, two things are important; firstly, the BA must ensure customer satisfaction. However, the BA must also see that the software product is of good quality. Both the BAs and the developers indicated that the customer's needs were given preference when the development of security and privacy features was under consideration. However, it is clear from the data analysis that the decision to prioritise the customer's preferences can result in security vulnerabilities. An example came to light during one of the team interviews. A customer wanted web banners, which would advertise the customer's other products in a mobile banking application.

Despite advances in technology, security vulnerabilities are still a significant problem because of the human behavior (Lacey, 2009). Security and privacy practices should not only be within the organisation's domain but should extend to external entities such as customers (Lacey, 2009). In a study carried out by Zhu (2015), it was noted that customers are not concerned or familiar with security technologies and possible threats. Although Zhu (2015) interest was primarily directed at the customer's awareness of security issues that affect Internet banking, the same principles can be applied to the mobile banking security awareness as both channel access banking via the Internet. Customers may not be aware of possible security threats and

vulnerabilities arising from requested requirements such as the need of advertisement links inside a mobile banking app.

STANDARDS AND GUIDELINES

According to Lacey (2009), *"It's vital also to ensure that project managers and development staff appreciate the importance of developing secure systems, based on intrinsically secure protocols and coding standards"* (p. xxi). Standards and guidelines are types of requirement that can help in strengthen software security (Rindell, Hyrynsalmi & Leppänen, 2015). However, existing security and privacy guidelines prove to have some misalignment (Notario et al., 2015). In addition, the idea of guidelines and standards implies the understanding that the particular processes and/or procedures will follow a certain order. In the case of an organisation which has adopted agile methods, this might be difficult because agile methods are a much less rigid and more informal way of working (Rindell et al., 2015). Joorabchi et al., (2013) have identified certain standards, such as the Human Computer Interaction (HCI) standards, as problematic when a mobile application is being developed. Each type of mobile device follows a different set of standards as there is great emphasis on enhancing the user experience (Joorabchi et al., 2013).

Android is an open source platform that has a number of stakeholders. Joorabchi et al., (2013) pointed to the risks inherent in this practice: *"each manufacturer modifies the source code to their own desires and releases it; sometimes they do not stick to the standards"* (p. 17). This example of misalignment between an established standards and guidelines and the way mobile devices operate can result in fragmentation within the Android platform. Attempts to add security and privacy requirements to an already fragmented platform can increase the challenges that face a software development team.

Standards and guidelines are especially important when it comes to data storage and privacy practices as this type of security standard or guideline is based on "*ethical values and social perceptions*" (Notario et al., 2015, p. 151). Most of the standards and guidelines which regulate data storage and privacy are formulated by people with a legal background. Few security and privacy practices fully accommodate the field of software engineering. Instead the focus is on the programming languages and the process or processes that the development team must follow. This accounts for the limited awareness or ignorance of the standards and guidelines among the members of the development team (Notario et al., 2015).

REGULATORY REQUIREMENTS

The data analysis identified the complexity of the government regulations which prescribe how personal information is to be kept secure and private. Software development is global industry; different countries

have different laws and regulations. A company may provide software to customers across the world. Consequently the company will have to deal with the different laws and regulations that apply in the jurisdictions where it does business.

Governments promulgate laws and draw up regulations which stipulate how personal data is to be stored and used. The numerous parties, which range from government bodies to sales and marketing teams, collect all sorts of personal information. They have to store and use this information in the manner prescribed by government. The problems surrounding security and privacy regulatory requirements are exacerbated by constantly changing technology and software. There are a few security policies which govern how security and privacy requirements are integrated into the various smart phone platforms. Most, if not all, of the applications which run on the smart phones require a connection to the Internet, from time to time. The Internet is borderless, which makes the formulation of security and privacy policies difficult (Brechbühl, Bruce, Dynes, & Johnson, 2010). There is no generally accepted regulatory framework that a government can use as a template when it formulates its country's security policies on privacy and security issues in software. This adds another layer of complexity to finding solutions to the dilemmas which surround security and privacy. In addition, most governments are not well equipped to deal with security and privacy issues in software (Harknett & Stever, 2011).

Commercial firms are in business to make a profit. To this end they are often willing to "*test legal boundaries and may risk sanctions for privacy breaches to avoid constraining their business*" (Spiekermann, 2012, p. 38). It is important that the stakeholders such as the government are involved in planning security and privacy strategy and policies as this will ensure that all parties have an understanding of external and internal regulations (Oueslati et al., 2015).

THIRD-PARTY SOFTWARE

Third-party applications are ready-made external software components that are used in software development as a means of improving the quality of the application under development. Their use is also a way of keeping down the cost of software development (Arhippainen, 2003; Haddox, Kapfhammer, Colyer & Tsai, 2009). This study has revealed a number of concerns around the use of third-party software; the lack of control is one such concern. Previous studies which dealt with the use of third party applications in software development treated this practice as similar to the reuse of software in Commercial-Off-the-Shelf (COTS) packages. The increased use of third-party applications in software development has led software engineers and IS specialists to research this topic further (Arhippainen, 2003; Javed, Sattar, & Faridi, 2012).

Haddox et al., (2009) identified challenges to integrating third-party software. Many businesses that buy third-party software do not have access to the source code of third party software. Even in cases where the source code is available, the development team may not know how the application will behave and so will have limited control over the outcome. De Jonge, (2009) pointed to a different challenge when a team works at integrating third party software into software that was built in-house. The source of the problem is that most software that is built in-house is not standardised. The result is that the *"third-party software does not fit"* (De Jonge, 2009, p. 64). This example of a misfit between software purchased from an external party and software built in-house results, in part, from the development team not having control over third-party software.

Other circumstances which give rise to a misfit between third-party and in-house software have been mentioned in the literature. Since third-party vendors always provide the updates to their software, the purchaser does not have control of the new functionality or even how it will integrate with the older functionality (Haddox et al., 2009). Quite often, the vendor does not inform the purchaser about an application's functionality or possible defects, another manifestation of lack of control (Haddox et al., 2009).

5.3.2 Role Misalignment

A software development team usually includes the following roles: software developer, tester, business analyst, project manager, security engineer and IT manager. Dhaliwal et al., (2011) refer to this collection of roles as an '*internal IT subunit'*. The different roles that make up a team must be clearly defined and align with each other so that each team member knows which tasks they are expected to perform. The alignment of roles has been defined as "*the congruence between the subunits along several relational and structural dimensions*" (Dhaliwal et al., 2011, p. 327). According to Dhaliwal et al., (2011) "*a well-aligned IT unit*" (p. 335), will help meet organisational strategic IT goals more readily than one where roles are misaligned. If not well aligned, team performance will be affected as "*these subunits need aligned goals and operations in order to deliver software applications that meet business needs*" (Dhaliwal et al., 2011, p. 326).

The results of the data analysis showed there were a number of misaligned roles, for example, the roles of the BA and the tester, the BA and the developer and the developer and tester. At the start of each project, the team members should receive specific instructions as to their duties and how the various roles fit together and support the team effort. This will ensure that the holders of the different roles perform the

tasks allotted to their role. It also prevents the team from relying on one or two role players to drive the project forward (Onita & Dhaliwal, 2011).

A number of studies have investigated how roles and skills align in Software Engineering, IS and other IT fields (Faraj & Sproull, 2000; Onita & Dhaliwal, 2011). Some researchers have done studies on misalignment within the internal IT unit (Dhaliwal et al., 2011). Misalignment between software developers and testers in particular, has received considerable attention (Zhang, Stafford, Dhaliwal, Gillenson & Moeller, 2014; Mbekela & Brown, 2014; Zhang, Dhaliwal, Gillenson, & Stafford, 2013; Ghobadi & Mathiassen, 2015; Liu, Chen, Chen, & Sheu, 2011).

A large portion of an information technology budget is spent on software development and testing. This reveals of the importance of the roles of the developer and tester. The success of any software development project depends on the appropriate alignment between these two functions. Conflict between software developers and testers has been mentioned by a number of researchers (Dhaliwal et al. 2011; Zhang et al., 2014; Mbekela & Brown, 2014). It is difficult to avoid conflict between these two roles because the testers have to try and break the functionality created by the developers. The hostility engendered by the conflict can undermine collaboration and team spirit. Alignment among software development roles can be achieved when there is a *"shared understanding, partnership and competencies"* (Dhaliwal et al. 2011, p. 337).

A shared understanding is fostered when the different role holders appreciate how each role fits in with and supports all the others; how the functions and responsibilities associated with each role contribute to getting the job done. Individual team members will then be able to communicate what they need to do a good job and will be prepared to share ideas. The difficulties of the developer-tester relationship are similar to those that come between the BA, who is responsible for collecting and documenting the requirements, and the software developer. If they *share the same understanding* of how the requirements' documents will be used in the development process, then the BA can align the process with the way the software developer will utilise the document as well as with how the document will be used during testing (Dhaliwal, et al., 2011).

The concept of partnership is based on the notion of people working together in order to define goals and objectives and then put them into practice. Partnership helps facilitates a common understanding and aids working towards achieving the success of the software project (Liu, Chen, Chen & Sheu, 2011). Each role player must know what the other role players' duties and needs are. The team members share an understanding of "*how their role fits within the entire process*" (Dhaliwal et al., 2011, p. 329). Communication will be easier and conflict between roles will be lessened as a result (Liu et al., 2011).

Successful teamwork depends on the role players being willing to co-operate and collaborate. The BA will probably make fewer changes to requirements because there are fewer inconsistencies and the developers and the testers are likely to have a better grasp of the requirements (Liu et al., 2011; Ghobadi & Mathiassen, 2015).

5.3.3 Skills Misalignment

The data collected revealed that there were inadequate skills in the area of security and privacy implementation, as well as a poor understanding of security and privacy guidelines and standards. The deficiency of skills can result in security and privacy concerns being overlooked. The skill set of an individual relates to the competencies of an individual (Preston & Karahanna, 2009). In this study, the researcher refers to this type of deficit as skills misalignment. This type of misalignment is related to role misalignment.

Team members' competence or the lack of competence in dealing with issues of security and privacy are related to an individual's level of security awareness. It is unlikely for one to take into consideration security and privacy standards if they are unaware of these standards and guidelines. Data collection and analysis showed a deficiency in skills to document, develop and test security and privacy requirements. Mouratidis, Giorgini and Mansona (2005) insist that secure software development is a specialist area. They point out that many developers do not have the right skills to develop secure applications. Siponen (2001) describes various dimensions of security awareness which include organizational, general public, socio-political, computer ethical and institutional education dimensions. The public dimension includes IT professionals and end-users. Poor understanding or awareness of security matters is not an issue which involves end users alone.

In organisations that adopt an agile method, the developers are likely to take on the role of the security specialist. This situation is far from ideal as most developers do not have the correct skill set (Rindell et al., 2015) because *"security is very complex and secure systems can only be developed by security experts and not by agent system developers*" (Poslad & Calisti, 2000, p. 2). Role-based training must be offered to all the members of the team as this will ensure that the security requirements are correctly aligned in the software development lifecycle. The product owner or the BA would then know how to include security requirements when documenting the business requirements. The developers and tester would have a good foundation from which to work (Rindell et al., 2015).

Hiring policies need to pay attention to the alignment of skill sets so that the different team roles complement each other. Research indicates that testers and business analysts do not have the same level of

technical qualifications as the developers. This may result in the BA and the tester not having the confidence to perform technical tasks such as documenting security requirements and performing security testing (Onita & Dhaliwal, 2011). In the end, the developers have to perform certain technical functions, for example, documenting technical requirements and performing white box testing (Dhaliwal et al., 2011). In the case of security testing, it is unlikely that they will be able to pick up their mistakes as they would have developed the software (Rindell et al., 2015). Knowledge sharing can help transfer skills. However, knowledge sharing and transfer is not always easy because of the differences between the roles and the skill sets associated with them (Ghobadi & Mathiassen, 2015). There is a great need for more research into the internal alignment of IT roles, both academic and workplace studies are necessary (Dhaliwal et al., 2011).

5.3.4 Requirements Misalignment

Requirements are categorised as either functional or non-functional. Functional requirements relate to the behaviour of the system in terms of its input and output. Non-functional requirements relate to the quality of the system, such as performance, usability and security (Glinz, 2007). It is important to note that security can be a functional requirement. Functional and non-functional requirements are equally important and both must receive attention during software development. The differentiating of requirements into functional and non-functional requirements results in fragmentation of requirements. Fragmentation in requirements classification is important but can result in alienating the non-requirements. After the design stage non-functional requirements receive less attention than the functional requirements as they are not seen as a high priority and not related to customer requirements (Mouratidis et al., 2005).

Security requirements can be defined as *"constraints on functional requirements… [which] stipulate the elimination of vulnerabilities that an attacker can exploit to carry out threats on assets, thereby causing harm*" (Haley, Laney & Nuseibeh, 2004, p. 1). Misalignment of security and privacy requirements can occur with functional requirements that would have been stated from the beginning of the software development life cycle "*since security mechanisms would have to be fitted into a pre-existing design, therefore leading to design challenges that usually translate into software vulnerabilities"* (Mouratidis et al., 2005, p. 610). Privacy and security requirements have a huge impact on how the software development process progresses (Ullah & Lai, 2011). These requirements are introduced to reduce the possibility of vulnerabilities within functional requirements. However, security requirements and functional requirements clearly crosscut each other (Haley et al., 2004).

Ensuring that security requirements are considered throughout the software development process will reduce the likelihood of security and privacy requirements conflicting with functional requirements

(Mouratidis et al., 2005). The security and privacy objectives that must be taken into consideration when developing software are integrity, availability and confidentiality. Integrity relates to the accuracy that the software will produce, availability relates to accessibility and usability of the system while confidentiality relates to prevention of unauthorised accesses to the information that the software will store (Ullah & Lai, 2011).

5.4 Summary

This chapter examined the concepts, 'the nature of the domain' and 'misalignment'. The researcher discussed how these concepts were delved into in the literature. The relationship between misalignment and the nature of the domain was also scrutinised. The nature of the mobile domain contributes to the different forms of misalignment. The researcher identified and discussed the different forms of misalignment that appear in the literature. The next chapter provides a conclusion to the study, a summary of the findings, how the study could contribute to research, the study's limitations and possible directions of future research

Chapter 6

Conclusion

6.1 Summary of Finding

This study has aimed at providing a better understanding of the core challenge that is faced when integrating security and privacy requirements into the mobile application development process by a team which makes use agile methods such as the scrum methodology. Security and privacy requirements can be added to the development process by defining specific security requirements and by acquiring a third party security application. The study focused on a team that creates mobile banking applications. Misalignment was identified as the core challenge. While the concept of misalignment has been previously studied in the field of IT security (Doherty & Fulford, 2006; Fruehwirth et al., 2010; Frühwirth, 2009), this study highlights it as the main concern for secure mobile banking application development.

The main forms of misalignment that were identified in the study include external misalignment, role misalignment, skills misalignment and requirements misalignment. External misalignment refers to misalignment with the customer's requirements; with established standards and guidelines; with regulatory requirements and third-party software components. They are described as external because the designated elements originate from outside the development team.

The results of the study indicate that the mobile application domain is affected by a number of shortcomings, for example, security awareness, lack of control and fragmentation which create the above-mentioned misalignment challenges. Security awareness, lack of control and fragmentation make up a category that was identified in the study as the nature of the domain. Security awareness is linked to the end users and the members of the software development team. Security awareness refers to knowledge of security and privacy standards and guidelines, and the ability to implement security and privacy requirements. The development team lacked the knowledge and skills that are necessary for dealing with the security and privacy issues that have already been mentioned. The team exhibited inadequate control of the volatile mobile environment. There are a number of factors, for example, the availability of third party application source code that the development team are not able to control. Changes that the team needed to make in the source code were controlled by the third party vendor. This limited the team on what they could add in and remove to suit their needs. Fragmentation is an aspect that is known to mobile application development. Mobile operating systems, mobile phones, programming languages and versions of the same

operating system demonstrate the aspect of fragmentation. Fragmentation is an aspect that results in misalignment.

The area of security and privacy has been identified as an area that requires expertise that a team may not always have, especially in a scrum team that has labelled roles of the team (developers and testers), the product owner (BAs) and the scrum master. As indicated by the data analysis results, the developer usually takes on the role of a security specialist who must ensure that the security requirements are documented correctly and that the correct test data are available. However, a developer may not always have the necessary skills that are essential for the role of security specialist. This type of misalignment can be addressed by identifying the necessary remedial training practices. With appropriate training, team members will be able to identify security and privacy issues and then take the necessary action to deal with them. Training should incorporate established standards and guidelines and provide practical skills for BAs, developers, testers and project managers.

6.2 Research Contribution

Lacey (2009) argues that security is a relatively new field in software development and mobile application development is also young. A great deal of research is needed in both areas. This research study will from a non-technical view add to the theory of software development in the areas of security and privacy and of mobile application development. Adolph et al., (2011) indicate that most research in software engineering typically focus on tools and negate other factors that drive the productivity in a software development team such as social factors. In this study, issues around role misalignment were identified as part of the challenges to integrating security and privacy requirements into mobile banking application development. Misalignment in roles can be reduced by effective communication within the team. Communication is a social aspect that helps in improving productivity of any team.

Findings from the study indicated four forms of misalignment that create challenges when integrating security and privacy requirements into mobile banking applications. Organisations can address the four forms of misalignment to ensure that the process of adding security and privacy requirements is less challenging. This research has pointed out that misalignment issues must be identified before commencing with a software development project, especially one in a specialist area such as security.

As indicated in the research methodology chapter, studies that follow the grounded theory methodology can potentially add to the existing coding families that were initially proposed by Glaser (Hernandez, 2009). The researcher has noted the similarities in outcome of the current study to the work of Volkoff and Strong

(2010) on types of misfits on ERP systems. The current study can potential be added to the grounded theory '*type coding family'* that was identified in the work of Glaser (1967).

6.3 Limitation of the study

The study followed the classical grounded theory methodology, often termed the Glaserian grounded theory methodology, which values the importance of theoretical sampling. However, due to time constraints, the process of theoretical sampling was not fully employed in the current study.

6.4 Future Research

It is important to expand on the current exploratory study and focus on additional research on misalignment challenges in integrating security and privacy requirements. This is necessary in order to develop descriptive and explanatory theory in the subject matter. Further research can build on the current study by developing propositions that provide a deeper explanation of the relationships between the facets of the nature of the domain (security awareness, lack of control, fragmentation) and the different forms of misalignment that were identified in this study. Subsequent studies can follow a method similar to that employed by Volkoff and Strong (2010) in their work on the lack of fit in ERP systems which use critical realism.

Misalignment of roles in software engineering has focused mostly on the roles of the developer and tester. Although the two roles are interdependent, they also come into conflict (Dhaliwal, Onita, Poston & Zhang, 2011; Mbekela & Brown, 2014; Onita & Dhaliwal, 2011). The analysis of the data indicated the importance of the role of the BA who must ensure that the developers and testers understand the requirements and how they are to be used in their particular roles. Thus, it is important that researchers studying agile teams pay attention to the role of the product owner/ BA and how it needs to align with the roles of the developer and the tester. The need for this type of research is supported by Dhaliwal et al., (2011) who pointed out the need for academics to direct more research on role alignment within an IT unit.

One of the participants, a BA, said that the guideline for selecting the third-party security vendor, more specifically, the framework for selecting security and privacy standards and guidelines needs to be an area of future research. When organisations select security and privacy guidelines, the choice is often based on popularity rather than suitability. Javed, Sattar and Faridi (2012) have studied the mechanism for selecting third-party software, but research into the mechanism for selecting security third party applications is lacking.

Complex Adaptive Systems (CAS) theory could be a useful theory used in research to provide a better explanation on the misalignment issues in mobile application development. Few researchers in IT related fields such as project management and software engineering have looked at CAS. Highsmith and Cockburn (2001) looked into CAS and its relationship with the agile methods. Meso and Jain (2006) also looked at the agile methods as a form CAS. However, there is no evidence that CAS has been applied to understand the misalignment challenges that are faced by agile software development teams. Although CAS originated as a scientific theory (Meso & Jain, 2006), there is evidence that substantial progress in research can come from adapting ideas from other fields of study (Meso & Jain, 2006). The integration of security and privacy requirements into the process of software development indicates the development of a secure software application as a complex adaptive system. The software development space is an arena with many security challenges; consequently, security needs to be adjusted constantly in order to ensure that an application can withstand any security threat. Software development teams could be encouraged to adopt CAS as a means of understanding some problems in software development. One should not reject CAS when dealing with the challenges inherent in the process of software development. Instead, researchers could explore ways that other theories might supplement the gaps in CAS theory (Meso & Jain, 2008). Meso & Jain (2006) have investigated whether the Zipf's theory of the 'path of least effort' can be applied to the concept of minimum documentation in agile development theory.

The current study mainly focused on a single case study. Possible future research can focus on multiple case studies to try and replicate the current study in different software development scenarios, which may not be limited to the development of mobile applications. In addition, future research can focus on whether the choice of a software development methodology affects secure software development.

6.5 Summary

This chapter has provided a conclusion to a study of the core challenge which is encountered when integrating security and privacy requirements into the mobile application development lifecycle. A summary of the findings is followed by a brief account of the contribution and limitations of the current study. The concluding chapter ends with some recommendations of areas of future research. The recommendations are based on the research findings.

References

Abrahamsson, P. (2007). Agile software development of mobile information systems. *Advanced Information Systems Engineering*, 1-4.

Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jäälinoja, J., Korkala, M., Koskela, J., Kyllönen, P. & Salo, O. (2004). Mobile-D: an agile approach for mobile application development. In *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications* (pp. 174-175). ACM.

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

Adolph, S., Kruchten, P., & Hall, W. (2012). Reconciling perspectives: A grounded theory of how people manage the process of software development. *Journal of Systems and Software*, 85(6), 1269-1286.

Al-Jabri, I. M., & Sohail, M. S. (2012). Mobile banking adoption: Application of diffusion of innovation theory. *Journal of Electronic Commerce Research*, 13(4), 379-391.

Andrade, A. D. (2009). Interpretive research aiming at theory building: Adopting and adapting the case study design. *The Qualitative Report*, 14(1), 42-60.

Angelakopoulos, G., & Mihiotis, A. (2011). E-banking: Challenges and opportunities in the Greek banking sector. *Electronic Commerce Research*, 11(3), 297-319.

Arhippainen, L. (2003). Use and integration of third-party components in software development. Retrieved from http://www.vtt.fi/inf/pdf/publications/2003/P489.pdf

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

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

Bickford, J., O'Hare, R., Baliga, A., Ganapathy, V., & Iftode, L. (2010). Rootkits on smartphones: Attacks, implications and opportunities. In *Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications* (pp. 49-54). ACM.

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

Botha, A., Makitla, I., Ford, F., Fogwill, T., Seetharam, D., Abouchabki, C., Oguneye, O. (2010). Mobile phone in Africa: Providing services to the masses. Retrieved from https://www.researchgate.net/publication/46063464_Mobile_phone_in_Africa_providing_services_to_the_masses.

Brancheau, J. C., Janz, B. D., & Wetherbe, J. C. (1996). Key issues in information systems management: 1994-95 SIM delphi results. *MIS Quarterly*, 20(2), 225-242.

Brechbühl, H., Bruce, R., Dynes, S., & Johnson, M. E. (2010). Protecting critical information infrastructure: Developing cybersecurity policy, 83-91.

Bryant, A., & Charmaz, K. (2007). *The sage handbook of grounded theory*. Sage.

Cao, L., & Ramesh, B. (2008). Agile requirements engineering practices: An empirical study. *Software, IEEE*, 25(1), 60-67.

Carcary, M. (2009). The research audit trial–enhancing trustworthiness in qualitative inquiry. *The Electronic Journal of Business Research Methods*, 7(1), 11-24.

Chan, Y. E., & Reich, B. H. (2007). IT alignment: what have we learned?. *Journal of Information Technology*, 22(4), 297-315.

Charmaz, K. (2014). *Constructing grounded theory*. Sage.

Coleman, G., & O'Connor, R. (2007). Using grounded theory to understand software process improvement: A study of Irish software product companies. *Information and Software Technology*, 49(6), 654-667.

Coursaris, C., Hassanein, K., & Head, M. (2003). M-commerce in Canada: an interaction framework for wireless privacy. *Canadian Journal of Administrative Sciences*, 20(1), 54-73.

Darsow, M., & Listwan, L. (2012). Corporate practitioners moving to mobile banking: Key factors driving adoption. *Journal of Payments Strategy & Systems*, 5(4), 360-372.

Daud, M. I. (2010). Secure software development model: A guide for secure software lifecycle. In *Proceedings of the international MultiConference of Engineers and Computer Scientists* (Vol. 1, pp. 17-19).

Davis, N., Humphrey, W., Redwine Jr, S. T., Zibulski, G., & McGraw, G. (2004). Processes for producing secure software. *Security & Privacy, IEEE*, 2(3), 18-25.

De Jonge, M. (2009). Developing product lines with third-party components. *Electronic Notes in Theoretical Computer Science*, 238(5), 63-80.

Degirmenci, K., Guhr, N., & Breitner, M. (2013). Mobile applications and access to personal information: A discussion of users' privacy concerns. Retrieved from http://aisel.aisnet.org/icis2013/proceedings/SecurityOfIS/6/.

Dey, I. (2007). Grounding categories. *The Sage Handbook of Grounded Theory*, (Part III), 167-190.

Dhaliwal, J., Onita, C. G., Poston, R., & Zhang, X. P. (2011). Alignment within the software development unit: Assessing structural and relational dimensions between developers and testers. *The Journal of Strategic Information Systems*, 20(4), 323-342.

Doherty, N. F., & Fulford, H. (2006). Aligning the information security policy with the strategic information systems plan. *Computers & Security*, 25(1), 55-63.

Douglas, D. (2003). Inductive theory generation: A grounded approach to business inquiry. *Electronic Journal of Business Research Methods*, 2(1), 47-54.

Elmer-Dewitt, P. (2013). Apple cracks the fortune 10. Fortune. Retrieved from http://fortune.com/2013/05/06/apple-cracks-the-fortune-10/.

Faraj, S., & Sproull, L. (2000). Coordinating expertise in software development teams. *Management Science*, 46(12), 1554-1568.

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

Fruehwirth, C., Biffl, S., Tabatabai, M., & Weippl, E. (2010). Addressing misalignment between information security metrics and business-driven security objectives. *In Proceedings of the 6th International Workshop on Security Measurements and Metrics* (pp. 6). ACM.

Frühwirth, C. (2009). On business-driven it security management and mismatches between security requirements in firms, industry standards and research work. *In Product-Focused Software Process Improvement* (pp. 375-385). Springer Berlin Heidelberg.

Ghobadi, S., & Mathiassen, L. (2015). Perceived barriers to effective knowledge sharing in agile software teams. *Information Systems Journal*, 26(2), 91–190.

Glaser, B. G. (1978). *Theoretical sensitivity*. Mill Valley, CA: Sociology Press.

Glaser, B. G., Strauss, A. L. (1967). *The discovery of grounded theory: Strategies for qualitative research. Chicago*. Aldine Pub. Co.

Glinz, M. (2007). On non-functional requirements. In *Requirements Engineering Conference, 2007. RE'07. 15th IEEE International* (pp. 21-26). IEEE.

Goadrich, M. H., & Rogers, M. P. (2011). Smart smartphone development: IOS versus Android. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (pp. 607-612). ACM.

Haddox, J. M., Kapfhammer, G. M., Colyer, R., & Tsai, T. (2009). Method for understanding and testing third-party software components. U.S. Patent No. 7,539,978. *Washington, DC: U.S. Patent and Trademark Office*.

Haley, C. B., Laney, R. C., & Nuseibeh, B. (2004). Deriving security requirements from crosscutting threat descriptions. In *Proceedings of the 3rd International Conference on Aspect-Oriented Software Development* (pp. 112-121). ACM.

Hammershoj, A., Sapuppo, A., & Tadayoni, R. (2010). Challenges for mobile application development. In *Intelligence in Next Generation Networks (ICIN), 2010 14th International Conference* (pp. 1-8). IEEE.

Harknett, R. J., & Stever, J. A. (2011). The new policy world of cybersecurity. *Public Administration Review*, 71(3), 455-460.

Hernandez, C. A. (2009). Theoretical coding in grounded theory methodology. *Grounded Theory Review*, 8(3).

Highsmith, J., & Cockburn, A. (2001). Agile software development: The business of innovation. *Computer*, 34(9), 120-127.

Javed, Z., Sattar, A. R., & Faridi, M. S. (2012). Unsolved tricky issues on COTS selection and evaluation. *Global Journal of Computer Science and Technology*, 12(10-D).

Jones, M., & Alony, I. (2011). Guiding the use of grounded theory in doctoral studies–an example from the Australian film industry. Retrieved from http://ro.uow.edu.au/commpapers/793/.

Joorabchi, M. E., Mesbah, A., & Kruchten, P. (2013). Real challenges in mobile app development. In *Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on* (pp. 15-24). IEEE.

Kim, H. K. (2011). Instrument of security assurance for mobile software development life cycle. *Journal of Security Engineering Research*, 8(2), 183-192.

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.

König-Ries, B. (2009). Challenges in mobile application development. *It-Information Technology Methoden Und Innovative Anwendungen Der Informatik Und Informationstechnik*, 51(2), 69-71.

La Polla, M., Martinelli, F., & Sgandurra, D. (2013). A survey on security for mobile devices. *Communications surveys & tutorials*, IEEE, 15(1), 446-471.

Lacey, D. (2009), *Managing the Human Factor in Information Security*, Wiley, Hoboken, NJ.

Liebenau, J., Elaluf-Calderwood, S., Hosein, G., & Kärrberg, P. (2011). Near-field communications: Privacy, regulation & business models. Retrieved from http://eprints.lse.ac.uk/39076/.

Liu, J. Y., Chen, H., Chen, C. C., & Sheu, T. S. (2011). Relationships among interpersonal conflict, requirements uncertainty, and software project performance. *International Journal of Project Management*, 29(5), 547-556.

Luftman, J., & Kempaiah, R. (2007). An update on business-IT alignment:"A line" has been drawn. *MIS Quarterly Executive*, 6(3), 165-177.

Matavire, R., & Brown, I. (2013). Profiling grounded theory approaches in information systems research. *European Journal of Information Systems*, 22(1), 119-129.

Mbekela, U., & Brown, I. (2014). Factors that influence misalignment between developers and testers in agile organizations, and alleviation strategies employed. In *Proceedings of the e-Skills for Knowledge Production and Innovation Conference 2014*, Cape Town, South Africa, 203-210. Retrieved from http://proceedings.eskillsconference.org/2014/e-skills203-210Mbekela817.pdf

McCallin, A. M. (2003). Designing a grounded theory study: Some practicalities. *Nursing in Critical Care*, 8(5), 203-208.

McGraw, G. (2006). *Software security: building security in* (Vol. 1). Addison-Wesley Professional.

Meso, P., & Jain, R. (2006). Agile software development: Adaptive systems principles and best practices. *Information Systems Management*, 23(3), 19-30.

Mouratidis, H., Giorgini, P., & Manson, G. (2005). When security meets software engineering: A case of modelling secure information systems. *Information Systems*, 30(8), 609-629.

Myers, M. D. (1997). Qualitative research in information systems. *Management Information Systems Quarterly*, 21, 241-242.

Nahavandipoor, V. (2014). *IOS 8 Swift Programming Cookbook: Solutions & Examples for IOS Apps*. O'Reilly Media, Inc.

Notario, N., Crespo, A., Martin, Y., Del Alamo, J. M., Le Metayer, D., Antignac, T., Wright, D. (2015). PRIPARE: Integrating Privacy Best Practices into a Privacy Engineering Methodology. In *Security and Privacy Workshops (SPW), 2015 IEEE* (pp. 151-158). IEEE.

Onita, C., & Dhaliwal, J. (2011). Alignment within the corporate IT unit: an analysis of software testing and development. *European Journal of Information Systems*, 20(1), 48-68.

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

Pãvãloaia, V. (2013). Methodology approaches regarding classic versus mobile enterprise application development. *Informatica Economica*, 17(2), 59.

Peeters, J. (2005). Agile security requirements engineering. In *Symposium on Requirements Engineering for Information Security.* Retrieved from https://handouts.secappdev.org/handouts/2008/abuser%20stories.pdf.

Pickard, A. (2012). *Research methods in information*. Facet publishing.

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.

Poon, W. (2007). Users' adoption of e-banking services: The Malaysian perspective. *Journal of Business & Industrial Marketing*, 23(1), 59-69.

Poslad, S., & Calisti, M. (2000). Towards improved trust and security in FIPA agent platforms. *In Autonomous Agents 2000 Workshop on Deception, Fraud and Trust in Agent Societies, Spain*. Retrieved from http://www.eecs.qmul.ac.uk/~stefan/publications/2000-trust-security.pdf.

Preston, D., & Karahanna, E. (2009). How to develop a shared vision: The key to IS strategic alignment. *MIS Quarterly Executive*, 8(1), 1-8.

Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2015). A comparison of security assurance support of agile software development methods. In *Proceedings of the 16th International Conference on Computer Systems and Technologies* (pp. 61-68). ACM.

Rotimi, E., Awodele, O., & Bamidele, O. (2007). SMS banking services: A 21st century innovation in banking technology. *Issues in Informing Science and Information Technology*, 4, 227-234.

Royce, W. W. (1970). Managing the development of large software systems. In *Proceedings of IEEE WESCON,* 26(8) 328-388.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2), 131-164.

Savin-Baden, M., & Major, C. H. (2013). *Qualitative research: The essential guide to theory and practice*. Rutledge.

Schadler, T., & McCarthy, J. C. (2012). Mobile is the new face of engagement. Retrieved from Forrester Research:Http://cdn.BlogSap.com/innovation/files/2012/08/SAP_Mobile_Is_The_New_Face_Of_Engagement.Pdf.

Schwaber, K. (2004). *Agile project management with scrum*. Microsoft Press.

Sharp, H., & Robinson, H. (2008). Collaboration and coordination in mature eXtreme programming teams. *International Journal of Human-Computer Studies*, 66(7), 506-518.

Sikolia, D., Biros, D., Mason, M., & Weiser, M. (2013). Trustworthiness of grounded theory methodology research in information systems, In *MWAIS 2013 Proceedings*. Paper 16, Retrieved from http://aisel.aisnet.org/mwais2013/16.

Siponen, M., Baskerville, R., & Kuivalainen, T. (2005). Integrating security into agile development methods. In *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on* (pp. 185a-185a). IEEE.

Spiekermann, S. (2012). The challenges of privacy by design. *Communications of the ACM*, 55(7), 38-40.

Strauss, A., & Corbin, J. (1998). *Basics of qualitative research: Procedures and techniques for developing grounded theory*. Ed: Thousand Oaks, CA: Sage,

Strong, D., & Volkoff, O. (2010). Understanding organization-enterprise system fit: A path to theorizing the information technology artefact, *MIS Quarterly*, 34(4), 731-756.

Suddaby, R. (2006). From the editors: What grounded theory is not. *Academy of Management Journal*, 49(4), 633-642.

Ullah, A., & Lai, R. (2011). Managing security requirements: Towards better alignment between information systems and business. *In PACIS* (pp. 195).

Urquhart, C. (2000). An encounter with grounded theory: tackling the practical and philosophical issues. *Qualitative research in IS: Issues and trends*, 104-140.

Venkatesh, V., Ramesh, V., & Massey, A. P. (2003). Understanding usability in mobile commerce. *Communications of the ACM*, 46(12), 53-56.

Vlaanderen, K., Jansen, S., Brinkkemper, S., & Jaspers, E. (2011). The agile requirements refinery: Applying scrum principles to software product management. *Information and Software Technology*, 53(1), 58-70.

Walker, D., & Myrick, F. (2006). Grounded theory: An exploration of process and procedure. *Qualitative Health Research*, 16(4), 547-559.

Walsham, G. (2006). Doing interpretive research. *European Journal of Information Systems*, 15(3), 320-330.

Wasserman, A. I. (2010). Software engineering issues for mobile application development. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research* (pp. 397-400). ACM.

Whitman, M., Mattord, H. (2003). *Principles of Information Security*. Thomson Course Tech.

Worku, G. (2010). Electronic banking in Ethiopia- practices, opportunities and challenges. *Journal of Internet Banking and Commerce*, 15(2), 2-8.

Yang, A. S. (2009). Exploring adoption difficulties in mobile banking services. *Canadian Journal of Administrative Sciences,* 26(2), 136-149.

Yin, R. K. (2013). *Case study research: Design and methods*. Sage publications.

Yin, Z., Yuan, D., Zhou, Y., Pasupathy, S., & Bairavasundaram, L. (2011). How do fixes become bugs? In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering* (pp. 26-36). ACM.

Zhang, D., & Adipat, B. (2005). Challenges, methodologies, and issues in the usability testing of mobile applications. *International Journal of Human-Computer Interaction*, 18(3), 293-308.

Zhang, X., Dhaliwal, J. S., Gillenson, M. L., & Stafford, T. F. (2013). The impact of conflict judgments between developers and testers in software development. *Journal of Database Management (JDM)*, 24(4), 26-50.

Zhang, X., Stafford, T. F., Dhaliwal, J. S., Gillenson, M. L., & Moeller, G. (2014). Sources of conflict between developers and testers in software development. *Information & Management*, 51(1), 13-26.

Zhu, R. (2015). An initial study of customer Internet banking security awareness and behaviour in China. *Pacific Asia Conference on Information Systems,* Paper 87, Retrieved from http://aisel.aisnet.org/pacis2015/87/.

Appendix A: Interview Questions

Open Ended Questions

The researcher will use the following questions as a guide to the interview process, especially for the first round of interviews. The researcher is making use of grounded theory methodology. Data collection and analysis process is iterative, with data collection building from previous interviews. Hence, questions are likely to emerge from the participant's interview sections. A participant's interview questions will likely be built from the analysis of the previously interviewed participants.

1. Are there checkpoints throughout the software development lifecycle (SDLC) verifying and certifying that the security requirements are being met?
2. At what points will risk management be performed throughout the SDLC?
3. At what point will vulnerability checks be performed before a product is put into production?
4. What challenges are faced when ensuring secure code in an agile environment?
5. What are the core challenges in 3rd party libraries and other software and how do they influence security integration?
6. What is the biggest challenge in ensuring secure future development?
7. How do we best reduce the challenges?
8. Do you caution consumers against transmitting sensitive or confidential data via other non-secure systems?
9. Do you provide a central system for managing security updates and information, including notification of security risks and/or breaches?
10. Do policies and procedures help in ensuring security and privacy requirements are well integrated?

Appendix B: Ethics Form

UNIVERSITY OF CAPE TOWN
FACULTY OF COMMERCE
Igniting Knowledge and Opportunity

Commerce Faculty Ethics in Research Committee

Updated Ethics Form March 2013

Any individual in the Faculty of Commerce at the University of Cape Town undertaking any research that involves the use of human subjects, or research that may hold ethical consequences for the University of Cape Town, is required to complete this form and obtain approval before conducting research. The completed form should be submitted as an electronic document to departmental Ethics Committee representatives for submission to the Commerce Faculty Ethics in Research Committee. Please also submit electronic copies of your research proposal, informed consent form or other information used to obtain consent, and any questionnaires other material shown to subjects.

3.7.1 1. PROJECT DETAILS			
Project title:	Challenges in Integrating Software Development with Security and Privacy Requirements for Mobile Applications (Apps) Development: A Mobile Banking Apps Case Study		
Principal Researcher/s:	Memory Machiridza	Email address(es):	Mchmem001@myuct.ac.za
Research Supervisor:	Prof I Brown	Email address(es):	Irwin.brown@uct.ac.za
Co-researcher(s):	n/a	Email address(es):	
Brief description of the project: Security and privacy issues are major concerns for mobile applications users, especially when making use of mobile banking applications. Mobile apps are a fairly new technology growing at an exponential rate. However, security and privacy issues have to be addressed from the start of the development phase when the concept is generated until the product is ready for launch to the customers. Focus is on the challenges faced when integrating security and privacy requirements into the mobile application software development processes.			

Data collection: (please select)

☒Interviews ☐ Questionnaire ☐Experiment ☐Secondary data ☒Observation

☐ Other (please specify): _____

Procedure: (please describe)
The grounded theory methodology will be utilised to collect and analyse data. Initial interviews will help formulate questions that will be asked in the following interviews.

Approximately 15 participants involved in the building of mobile applications will be interviews face to face and via Google Chat.

3.7.2 2. PARTICIPANTS

Characteristics of participants:

Gender:	*Male and Female*
Race / Ethnicity:	*All races*
Age range:	*21 and Above*
Location:	*Cape Town, South Africa, Russia*
Other:	*Participants are likely to be skilled IT professionals*

Race / Ethnicity:

Have you included a "Prefer not to Answer" response category in your questionnaire? (please select)

☐ Yes ☐No ☒ Not applicable

If you answered 'No' why not?

I will be using open-ended questions for the interview process. However, the participants will be informed that they may decide not answer a question.

Affiliations of participants: (please select)

☒Company employees ☐UCT staff ☐General public ☐UCT Students

☐ Other (please specify): _____

If your sample includes children (aged 18 and below), mentally incompetent persons, or legally restricted groups please explain below why it is necessary to use these particular groups. If subjects are minors or mentally incompetent, please describe how and by whom permission will be granted? If you are including children under the age of 18 and are not getting parental consent, please explain why you believe that their parents would consent if it was possible to contact them.

3.7.3 3. ORGANISATIONAL PERMISSION

If your research is being conducted within a specific organisation, please provide organisational permission or explain how permission will be obtained.

The organization Vice President has given permission for me to conduct research using the company as a case study as long as the company name remains anonymous. This was during a meeting and no formal document was provided. However, a formal invitation letter to participant in the research has been sent to him and I am awaiting his response.

Are you making use of UCT students as respondents for your research? (please select) ☐ Yes ☒No

If yes, have you contacted Executive Director: Student Affairs for permission? (please select) ☐ Yes ☐No

Was approval granted? (please select) ☐ Yes ☐No ☐Awaiting a response

Are you making use of UCT staff as respondents for your research? (please select) ☐ Yes ☒ No

If yes, have you contacted Executive Director: Human Resources for permission? (please select) ☐ Yes ☐ No

Was approval granted? (please select) ☐ Yes ☐ No ☐ Awaiting a response

Contact Emails: Executive Director: Human Resources (Miriam.Hoosain@uct.ac.za)
Executive Director: Student Affairs (Moonira.Khan@uct.ac.za)

3.7.4 4. INFORMED CONSENT

What type of consent will be obtained from study participants?

☒ written consent

☐ anonymous survey

☐ oral consent (please justify)

☐ other (please specify)

- ○ Oral Consent
- ◉ Written Consent
- ○ Anonymous survey questionnaire (covering letter required, no consent form needed)
- ○ Other (please specify)

How and where will consent/permission be recorded?

MS Word documents

3.7.5 5. CONFIDENTIALITY OF DATA

What precautions will be taken to safeguard identifiable records of individuals? Please describe specific procedures to be used to provide confidentiality of data by you and others, in both the short and long run. This question also applies if you are using secondary sources of data that is not anonymous.

All participants will complete and sign a participant consent form - Appendix 1 attached on this application. Information collected during the interviews will be tape recorded and the files will be stored on Google drive which is password protected.

3.7.6 6. RISK TO PARTICIPANTS

Does the proposed research pose any physical, psychological, social, legal, economic, or other risks to study participants you can foresee, both immediate and long range? (please select)

☐ Yes ☒No

If yes, answer the following questions:

1. Describe in detail the nature and extent of the risk and provide the rationale for the necessity of such risks
2. Outline any alternative approaches that were or will be considered and why alternatives may not be feasible in the study
3. Outline whether and why you feel that the value of information to be gained outweighs the risks

1.
2.
3.

What authorship agreement have you reached with your co-researchers or supervisor?

☐ This research is not intended for publication

☒ Standard authorship agreement (principal researcher first author, co-researcher(s) and supervisor(s) co-authors)

☐ Customised agreement (please specify below):

I certify that we have read the the UCT Authorship Policy, and Commerce Faculty Authorship Guidelines (http://www.commerce.uct.ac.za/Commerce/Information/research.asp) ☒

I certify that that the material contained herein is truthful and that all co-researchers and supervisors are aware of the contents thereof. ☒

I understand that it is my responsibility to conduct research in accordance with the ethical requirements of UCT. ☒

Signed by candidate

_____
Applicant's signature:

Date: 09/12/2014

CHECKLIST	SELECT
A full copy of a research proposal or a literature review with methodology is attached	☒
Research proposal/ interview schedules / cover letters / questionnaires / forms and other materials used in the study are attached/ consent form	☒
Organisational consent letter / UCT student or staff approval letter	☐
On your cover letter to your questionnaire have you included the following?	**NA** ☒
1. The following UCT Logo 	☐
2. A sentence explaining the aim of the research	☐
3. Sentences of a similar nature to below must be included in the cover letter or consent form:	
This research has been approved by the Commerce Faculty Ethics in Research Committee.	☐

Your participation in this research is voluntary. You can choose to withdraw from the research at any time.	☐
The questionnaire will take approximately X minutes to complete	☐
You will not be requested to supply any identifiable information, ensuring anonymity of your responses.	☐
	OR
Due to the nature of the study you will need to provide the researchers with some form of identifiable information however, all responses will be confidential and used for the purposes of this research only.	☐
Should you have any questions regarding the research please feel free to contact the researcher (insert contact details).	☐
4. Have you scanned in your signature for the last section of the form?	☒

FOR ETHICS COMMITTEE REPRESENTATIVE ONLY
Recommendation(s):
Signature:
Date:
FOR ETHICS COMMITTEE CHAIRPERSON ONLY
Recommendation:
Signature:
Date:

UNIVERSITY OF CAPE TOWN

Faculty of Commerce
Ethics in Research Committee
Courier: Room 2.21 Leslie Commerce Building Upper Campus University of Cape Town
Post: University of Cape Town □ Private Bag □ Rondebosch 7701
Email: Irwin.brown@uct.ac.za
Telephone: +27 21 650-2311
Fax No.: +27 21 689-7570

January 7, 2015

Memory Machiridza
Information Systems

Project title: Challenges in Integrating Software Development with Security and Privacy Requirements for Mobile Applications (Apps) Development: A Mobile Banking Apps Case Study

Proposal no. 4-2015

Dear Researcher,

This letter serves to confirm that this project as described in your submitted protocol has been approved contingent on adding wording to the consent form saying that participation is voluntary, subjects can withdraw at any time, and how long the interview should take.

Please note that if you make any substantial change in your research procedure that could affect the experiences of the participants, you must submit a revised protocol to the Committee for approval.

Regards,

Professor Harold Kincaid

Signed by candidate

Commerce Faculty Ethics in Research Committee

Appendix C: Invitation for Participation in Study

Department of Information Systems

Leslie Commerce Building

Engineering Mall, Upper Campus

OR

Private Bag X3 - Rondebosch - 7701

Tel: +27 (0) 21 650 2261 Fax: +27 (0) 21650 2280

Letter of Participant Consent

Title of the Research Project: Challenges in Integrating Software Development with Security and Privacy Requirements for Mobile Applications (Apps) Development: A Mobile Banking Apps Case Study

Primary Researcher: Memory Machiridza

Supervisor: Prof Irwin Brown

- You have been selected to be one of the participants in the research project on the **Challenges in Integrating Software Development with Security and Privacy Requirements for Mobile Applications (Apps) Development: A Mobile Banking Apps Case Study.** Information that you provide will be treated with confidentiality and used purely for this research **only**. The following data collection tools will be used in the research: **Interview**

Please be informed that during the interview session the researcher may find it necessary to record some of the comments that you will have made specifically for the research project **only**.

Please note that participation is **voluntary** and **subjects can withdraw at any time**.

I, the participant, was invited to participate in the above-mentioned research project that is being undertaken by Miss M. Machiridza **from the** Department of Information Systems **of the University of Cape Town**

The following aspects were explained to me:

- Purpose of the study and what the information will be used for
- Procedures in which the research will be carried out

- The risks involved in the study
- Possible benefits as result of my participation in this study
- Confidentiality of my identity
- My participation in the research is voluntary l can withdraw as and when the need arise
- Withdrawing participating will not affect my present or future care / employment / lifestyle

THE INFORMATION ABOVE WAS EXPLAINED TO ME/THE PARTICIPANT BY:

Memory Machiridza in English

- I was given the opportunity to ask questions and all these questions were answered satisfactorily.

- No pressure was exerted on me to consent to participation and I understand that I may withdraw at any stage without any penalties.

- Participation in this study will not result in any additional cost to me.

An Interview will take approximately an hour.

I HEREBY VOLUNTARILY CONSENT TO PARTICIPATE IN THE ABOVE-MENTIONED PROJECT:

Signed at……

Date……

Contact Telephone No……..

Signature……

Thank you for your participation in the project!