EXAMINING THE RELATIONSHIP BETWEEN AGILE ADOPTION MOTIVATION FACTORS AND AGILE PRACTICE CLUSTERS USED BY SOFTWARE STARTUPS IN KINGDOM OF SAUDI ARABIA

ABDULLAH MOHAMMED HUSSEIN AL-SAKKAF

**MASTER OF SCIENCE (INFORMATION TECHNOLOGY)
UNIVERSITY UTARA MALAYSIA
2016**

Perakuan Kerja Tesis/Disertasi

(Please substitute signed document for this page)

Permission to Use

Abstrak

Metodologi pembangunan perisian agil (ASDM) semakin banyak diterima pakai dalam organisasi. Walaupun banyak manfaat yang ditawarkan oleh ASDM, penggunaan ASDM yang berjaya merupakan cabaran besar bagi organisasi. Kebanyakan kajian menunjukkan bahawa kaedah yang diterima pakai ini sebahagiannya adalah dengan cara memilih satu set amalan agil. Oleh itu, adalah sukar bagi penerima baharu memilih set amalan agil yang sesuai dengan keperluan kerana ASDM mempunyai amalan atau kelompok yang meluas. Amalan agil perlu dipilih berdasarkan faktor-faktor motivasi termasuklah keperluan organisasi untuk memaksimumkan manfaat penerimaannya. Tujuan kajian ini adalah untuk mengenal pasti hubungan antara faktor motivasi penggunaan ASDM organisasi dengan amalan kelompok agil. Kajian ini menggunakan pendekatan kuantitatif untuk menilai hubungan antara pemboleh ubah. Soal selidik dengan 76 pengamal perisian daripada pemula perisian (software startups) Kerajaan Saudi Arabia (KSA) telah dijalankan. Dapatan kajian akan membantu organisasi untuk memilih kelompok amalan agil yang sesuai dengan memadankan faktor motivasi yang mempengaruhi kejayaan penggunaan ASDM. Analisis menghasilkan 4 kelompok yang mana setiap satunya dikaitkan dengan senarai amalan. Kelompok-kelompok ini dilabel sebagai pengurusan projek, jaminan kualiti, proses perisian, serta kelompok tambahan dan berterusan. Kajian ini mendapati bahawa tiga faktor motivasi penggunaan (motivasi untuk peningkatan kualiti perisian, peningkatan kecekapan, atau peningkatan keberkesanan) berkaitan dengan jaminan kualiti, proses perisian, serta kelompok tambahan dan berterusan. Dengan memahami faktor-faktor ini dari segi penggunaan ASDM dan jenis amalan, pemilihan kelompok agil yang lebih sesuai akan membantu meningkatkan kejayaan proses penerimaan amalan agil. Tambahan lagi, kajian ini akan membantu untuk memahami cara pemula memilih amalan agil yang digunakan. Selain itu, kajian itu boleh membantu syarikat baharu untuk memilih amalan agil yang sesuai dengan mudah berdasarkan motivasi dan keperluan mereka.

Kata kunci: Kaedah perisian agil, Kelompok amalan agil, Penggunaan agil, Faktor motivasi penggunaan.

Abstract

Agile software development methodology (ASDM) has been increasingly adopted in organizations. Despite many benefits offered by ASDM, successful ASDM adoption is a big challenge for organizations. Many studies show that these methods were adopted partly by selecting a set of agile practices. Therefore, it is difficult for new adopters to choose agile practice sets that fit their organization needs as ASDM has a big pool of available practices or clusters. Agile practices should be selected based on motivation factors that include the organization needs in order to maximize the benefit of adopting them. The aim of this study is to identify the relationships between organization's ASDM adoption motivation factors and the agile practices clusters. This study used a quantitative approach to evaluate the relationships between these variables. The study was conducted using a questionnaire with 76 software practitioners from software startups in the Kingdom of Saudi Arabia (KSA). The analysis generated 4 clusters; each is associated with a list of practices. These clusters are labeled as project management, quality assurance, software process, and incremental and iterative clusters. This study finds that three adoption motivation factors (a motivation for increased software quality, increased efficiency, or increased effectiveness) are associated with the quality assurance, software process, and incremental and iterative clusters. By understanding these factors in terms of ASDM adoption and which types of agile practice cluster is more suitable will help to increase the success of the agile adoption process. Furthermore, the study will help to understand how the startups selected the practices used. Also, the study could help new startups to easily choose the proper agile practices based on their motivation and needs. The findings will help the organization to select suitable agile practices cluster by matching the motivation factors that correspondingly affect the ASDM successful adoption.

Keywords: Agile software methodology, Agile practice cluster, Agile adoption, Adoption motivation factors.

Acknowledgements

Alhamdulillah, all praises to Allah for the strengths and His blessing in completing this thesis.

Special appreciation goes to my supervisor, Dr. Nor Laily Hashim, for her supervision and constant support. Her invaluable help of constructive comments and suggestions throughout the thesis works have contributed to the success of this research. Not forgotten, my appreciation to my co-supervisor, Dr. Mazni Binti Omar for her comments which help to improve my work.

I would like to express my deepest gratitude to my mother, father, brothers, sister, my wife, my son, and all other relatives, for their emotional and moral support throughout my academic career and also for their love, patience, encouragement and prayers. My goal would not have been achieved without them. I dedicate this work to my parents, my wife, and my son.

Last but not least, I wish to express my sincere thanks to all those who have one way or another helped me in making this study a success.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

ASD	Agile Software Development
ASDM	Agile Software Development Methodology
BDD	Behavior-Driven Development
CA	Cluster Analysis
CEO	Chief Executive Officer
CTO	Chief Technology Officer
DSDM	Dynamic Software Development Method
FDD	Feature-Driven Development
HCA	Hierarchical Cluster Analysis
SBIN	Saudi Business Incubator Network
SPSS	Statistical Package for the Social Sciences
SE	Software Engineering
SME	Small and Medium Enterprise
TDD	Test-Driven Development
KSA	Kingdom of Saudi Arabia
MENA	Middle East and North Africa
MVP	Minimum Viable Product
XP	eXtreme Programming

CHAPTER ONE
INTRODUCTION

1.1 Background of the Study

Agile software development methodologies (ASDM) have become very effective for software development (Campanelli & Parreiras, 2015; West & Grant, 2010). They differ from traditional methodologies in the same way that they have less documentation, fast delivery, increase customer satisfaction, accept requirement changing, improve quality, and provide more transparency to customers (Pikkarainen, Salo, Kuusela, & Abrahamsson, 2012). ASDM are also more flexible and can bring benefits such as handling requirements changes, productivity gains, and business alignment (Campanelli & Parreiras, 2015). Among well-known ASDM are Extreme Programming (XP), Scrum, Lean software development, Featured-Driven Development (FDD), Dynamic software development method (DSDM), and Crystal methodologies (Dybå & Dingsøyr, 2008). Yet they share many of the core values and principles defined in the Agile Manifesto (Beck et al., 2001).

Agile adoption is a term used to describe a process of adopting and implementing agile practices, processes and values in software development. The practices to be implemented may either correspond to just one agile method or to a combination of multiple agile methodologies (O'Connor & Duchonova, 2014). The agile adoption process is dependent on organizational environment, agile methodologies, and practices where they often have to be tailored to be integrated into existing processes (Rohunen, Rodriguez, Kuvaja, Krzanik, & Markkula, 2010). Agile adoption is a continuous and interactive activity, which includes adaptation and customization of the development method throughout the execution of the project (Krasteva, Ilieva, & Dimov, 2010). Furthermore, agile adoption is a complex process because of many factors including organization culture, resistance to change, and need the support of the high level management team (Campanelli & Parreiras, 2015).

Based on the current challenges of software development, ASDM are interesting and viable options to achieve quality, project budget control, align with organization's business strategy and deliver value frequently and continuously (Campanelli & Parreiras, 2015). The adoption of ASDM in an organization is effected by different motivations, for instance: a desire to increase the quality, increase efficiency, or increase effectiveness (Tripp & Armstrong, 2014). Agile practices selection is consistent with the organizationals values, culture, reality, needs and strategies in order to generate a tailored ASDM (Campanelli & Parreiras, 2015). Tailoring ASDM is affected by organization's cultures, objectives, and an environment.

Each ASDM defines their own processes and practices, but they share in common the same values that are addressed in the agile manifesto (Beck et al., 2001). From the perspective of software development, choosing and adopting the proper development methodology is a critical task (Nerur, Mahapatra, & Mangalaraj, 2005). Each ASDM consists of several practices inside it, which includes pair programming, stand-up meeting, short iteration, retrospective and others (Williams, 2010). A big pool of available agile practices makes it difficult to select which set of practices fit the needs of organization (Conboy & Fitzgerald, 2010). This practice is useful in its own, but it provides more values when they are working together as a cluster of practices (Tripp & Armstrong, 2014).

Globally, a remarkable number of new software startups are launched every day. More specifically, in the Middle East and North Africa (MENA) region, software startups have been an emergent trend in the last few years. With a high growth rate, the number of startups in the MENA region has increased eight times in 2011 as compared with the data of 2005 (Dubai Internet City;Frost and Sullivan, 2012). This growth is led by many factors, which includes: many success stories, big opportunities, governmental regulation, and governmental support. It is also predicted that the rate of increment will be higher in the years to come. The term software startups refers to small software companies that are trying to explore new business opportunities and working to solve a problem where the solution is not well known in a highly volatile market (Giardino, Unterkalmsteiner, Paternoster, Gorschek, & Abrahamsson, 2014). The startup has a significance impact on the economics, as it is responsible for 20% of job creation in the United States only (Giardino et al., 2014). In the next chapter, the term startup will be defined and discussed based on the relevant literature. According to several studies conducted in US and worldwide, most of the new startups are software-related companies that are based on web technology (Coleman & O'Connor, 2008).

In term of software startups, there are many characteristics that make startups a unique type of software project. These different characteristics are discussed in different studies. Furthermore, many studies treat startups as a special situation related to software engineering, because of their uniqueness, as they behave differently in practice (Coleman & O'Connor, 2008). Richardson and Von Wangenheim (2007) discussed the differences in small software companies from different perspectives. They also stated that there is a gap in studies that focus on small software firms. Many researchers defined startups as a special case of small software organization, based on common characteristics and challenges. According to Giardino et al. (2014), there are two important characteristics used to differentiate startups from established companies, which are high uncertain and rapidly evolving. Another research paper by Sutton (2000) discussed more characteristics of startups, which are: a limitation of resources, youth and maturity, multiple influences, and a great risk to failure. There are the main characteristics of startups that affect the engineering and business.

Despite many startups being created, most of them fail within first two years (Crowne, 2002). One of the major reasons for this is the startup failed to develop the product (Paternoster, Giardino, Unterkalmsteiner, Gorschek, & Abrahamsson, 2014). There are other reasons, including failing to selecting the proper development methodology, and failure to adopt and adapt the suitable software methodology (Coleman & O'Connor, 2008; Giardino et al., 2014). Moreover, there are many other factors like failures in product development due to the development team being inexperienced. Or the product is not really a product, the product has no owner, there is no strategic plan for product development, and the product platform is unrecognized (Crowne, 2002). Furthermore, the startups do not have enough resources to investigate the best development methodology (Coleman & O'Connor, 2008). Otherwise, on some cases, the software development methodology is adopted partially at late stage of the startup life cycle (Paternoster et al., 2014).

However, there are very limited studies on agile methodologies carried out in MENA region (Alnafjan, 2012; Hajjdiab & Taleb, 2011; Hajjdiab, Taleb, & Ali, 2012). Figure 1.1 shows the publication pattern on agile software development based on the country, which indicates the gap on studies on the MENA region (Dingsøyr et al., 2012). In the figure, the darker color indicates more publications.

1.2 Problem Statement

Agile software development methodology has become a more popular development method for software development, especially on web and startup companies (Giardino et al., 2014; Taipale, 2010). Hence, ASDM have been characterized differently than plan-based methods, mainly with the focus on adapting to change and delivering prod-

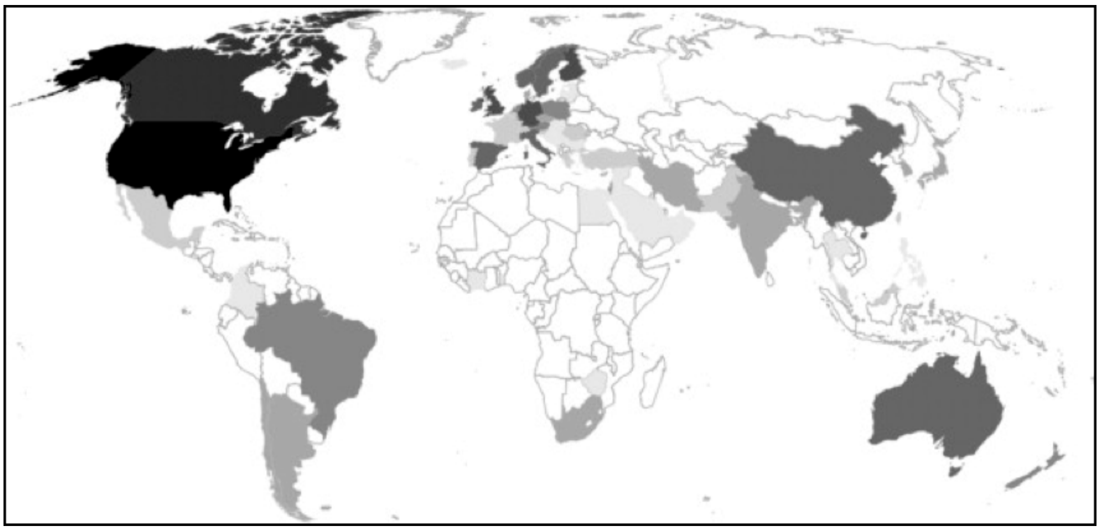
Figure 1.1. Publications on Agile Software Development by Country (Dingsøyr et al., 2012)

ucts of high quality through simple work-processes (Cockburn, 2006).

There are different types of agile methodologies, which consist of different processes, and practices. According to Yau and Murphy (2013), not all agile practices are suitable for small-scale web startups. There are some agile practices, which have a negative impact to the startups related to time or cost basis. Among the practices are pair programming and test-driven development (Yau & Murphy, 2013). In many cases, the failure of startups is mainly caused by the failure to develop the product due to a lack of adopting proper development methodology (Crowne, 2002; Murray, 2008; Shah, 2006). Agile adoption process is complex and not an easy task (Nerur et al., 2005). In practice, the organization only selects the agile practices that may be beneficial and applicable in a specific organizational context (Ayed, Vanderose, & Habra, 2014). Tripp and Armstrong (2014) asserted that there is a lack of studies done on factors that affect how agile methodology is tailored or selected. In other words, software startups need help to understand how to select a set of agile practices that fit their needs.

Moreover, the development of the software processes in general was developed in the

United States and European countries, which are fitted to their own culture (Asnawi, Gravell, & Wills, 2011). Hence, most empirical evidences come from these countries. However, a few researchers try to conduct studies on agile adoption in developing countries, such as Brazil (da Silva, Kon, & Torteli, 2005) and Malaysia (Asnawi, Gravell, & Wills, 2012). At the point this study is conducted, there were no enough studies found on agile adoption that has been done in the Kingdom of Saudi Arabia (KSA). Moreover, most of the recent studies did not focus on the startups as a medium that adopts ASDM, which currently plays a big role in software industry in the region. According to Paternoster et al. (2014), there is a global absence of studies related to the startup's software development, which gives significant importance for this research to fill up these gaps.

Based on the analysis of literature, it is clear that software startup has a big gap with the application of agile software development methodologies, from selecting the appropriate ASDM, through adoption, and applying suitable agile practice cluster. It is important to examine the relationship between the agile adoption motivation factor and agile adopted practices, in terms of understanding the whole adoption process. This at the end will affect the success of a project development. In addition, there is still a gap on studies that focuses on startups in a software engineering context.

Accordingly, this research aims to investigate the current states of agile practice of software startups in terms of agile adoption. The findings will help new research in the future to conduct a more comprehensive research in this area. Consequently, because there is a lack of studies which focus either on agile adoption on the MENA region or on software startups, this gives this investigated research an added significance.

The purpose of this study is to examine the relationship between the agile adoption motivation factors with the selected agile practice clusters for software startups in KSA. Furthermore, this study aims to explore the current status of agile practice clusters on software startups in KSA.

1.3 Research Questions

This study is set out to explore the following research questions:

RQ-1 What are the most used agile practice clusters by software startups?

RQ-2 What are the adoption motivation that impact software startups to adopt an agile methodology?

RQ-3 What is the relationship between agile adoption motivation factors and adopted agile practice clusters?

1.4 Research Objectives

Based on the background of the study and the problem statement discussed in the previous sections, the study was dedicated to achieve the following major research objectives:

1. To explore the current agile practice clusters used by software startups.
2. To investigate the adoption motivation that influence software startups to adopt ASDM.
3. To determine the relationship between agile adoption motivation factors and adopted agile practice clusters.

1.5 Scope of Study

The scope of this research is to investigate the relationship between agile adoption motivation factors and agile practice cluster. The population of this study is the software startups companies in KSA. The respondents of this study include the technical managers on the startups, which could be: chief executive officers (CEO), chief technology officers (CTO), It management, system analyst, quality assurance, tester, project manger, and the developers.

KSA is selected for two reasons. First, Saudi Arabia is treated on of biggest country to support startups with Jordan and United Arab Emirates. Second, the ecosystem in Saudi Arabia contains many business accelerator and incubator that supported by public and private sector, which makes it easier to conduct a study on it. According to Wyne and Wamda Research Lab (2014), 38 of all new startups companies will open offices in KSA within the next two years.

1.6 Significance of Study

The contributions of this study can be defined in two aspects: to the knowledge area and to the software startup industry.

Theoretically, this study contributes to the existing literature by examining the relationship between agile adoption motivation factors and agile adopted practice cluster. Understanding these factors in terms of ASDM adoption and which types of agile practice cluster is more suitable to increase the success of the method tailoring process. Also, it will help to understand how the startups selected the practices to use, as agile method tailoring is important to the successful of ASDM adopting. Furthermore, this study could help practitioners and researchers to understand the agile adoption motivation factors in the region, which will help researchers to understand the current status of the industry to improve and overcome any challenges in adopting the ASDM in the industry. Moreover, it will help practitioners to select a suitable agile practice cluster based on their needs.

Hence, this study will provide empirical evidence to the extant literature regarding agile software development in software startups, particularly in the context of agile adoption. A distinct lack of research performed in KSA area is apparent, and there is also a small number of studies that draw any comparison or connection with established product development theory. It is important to understand the motivation to adopt agile practices in order to help software startups to adopt ASDM successfully.

The industry of software startups could benefit from the study in many perspectives. The researcher tries to put a spotlight on the adoption of agile as a software development methodology on software startups. The study could help new startups to easily choose the proper agile practices based on their motivation and needs. A good understanding on how startups companies benefit from correct agile development methodology adoption is very important in order to support new startups launched every day. The startups incubators and accelerator, which on some phase provides the training to its company, will benefit from the result and provide proper agile development methodology training, which could lead to the success of the startups.

1.7 Thesis Structure

This study consists of five different chapters; this chapter elaborates the background of study, problem statement, research questions, research objective, and significance of study. The next chapter, Chapter 2, discusses agile software development methodologies, agile practices, as well as, defining software startups. Chapter 3 presents a research methodology which was used on this study. The steps of analysis of collected data from participants is then presented in detail in Chapter 4. Chapter 5 discusses the findings and results of this study.

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

This chapter presents the background and related work of the thesis. It first discusses the agile methodology. Then it describes concepts agile practices, which are the dependent variable of this study. Next, it discusses agile adoption and agile adoption motivations, which are the independent variables on this study.

The next section discusses the software startups. To study software startup, first, the term startups must be defined in a software engineering context. Finally, the related works will be discussed.

2.2 Agile Software Development

During the last two decades, agile methodology has dramatically increased its usage, which changed the way software development is performed (Diebold & Dahlem, 2014). According to a survey done by Azizyan, Magarian, and Kajko-Matsson (2011) in 35 countries, the result shows more than 66% of the surveyed companies use agile development methods. A recent study conducted globally by VersionOne Inc., found that among 3,880 respondents 58% of the surveyed companies teams use agile development methodologies in their projects (VersionOne, 2015).

Unlike traditional development methodologies characterized by sequential phases and heavy upfront planning, agile methodology deal with unpredictability and change by relying on people and close customer collaboration rather than formalized processes (Nerur et al., 2005). In 2001, the agile manifesto was written, which states that agile development core values are Individuals and interactions over processes and tools, Working software over comprehensive documentation, Customer collaboration over contract negotiation, and Responding to change over following a plan (*Manifesto for Agile Software Development*, 2001).

The main attributes of agile are short iterative, collaborative decision-making, quick feedback loops, and continuous integration of code changes into the product (Cockburn, 2006). Williams and Cockburn (2003) state that agile development is "about feedback and change", that agile methodologies are developed to "embrace, rather than reject, higher rates of change". Erickson, Lyytinen, and Siau (2005) defined agility as a means to strip away as much of the heaviness, commonly associated with the traditional software-development methodologies, as possible to promote rapid response to frequent changing in user requirements, faster project delivery and the like.

The number of agile methodologies has been grown, so now there are about 20 different agile or lean methods (Diebold & Dahlem, 2014). Table 2.1 shows the well-known agile development methodologies. Numerous studies stated that Scrum is the most used agile development methodology (Azizyan et al., 2011; Diebold & Dahlem, 2014; VersionOne, 2015). After Scrum, the other agile development methodologies were introduced, like: eXtreme Programming (XP), Crystal methodologies, Lean software development, and Feature-driven development (Dybå & Dingsøyr, 2008).

Agile software development methodology are suitable for projects with small teams or high innovative projects (D. Cohen, Lindvall, & Costa, 2004), both criteria exist on a software startup project. Another study by Nerur et al. (2005) confirmed that agile methodologies are fit for projects with a high level of requirement changing, a project that provides a high quality solution for end user, or for companies with an innovative culture.

Table 2.1
A Summary of Agile Development Methodologies

No.	Method	Description
1	Crystal methodologies	A family of methods for co-located teams of different sizes and criticality: Clear, Yellow, Orange, Red, Blue. The most agile method, Crystal Clear, focuses on communication in small teams developing software that is not life-critical. Clear development has seven characteristics: frequent delivery, reflective improvement, osmotic communication, personal safety, focus, easy access to expert users, and requirements for the technical environment (Cockburn, 2004).
2	Dynamic software development method (DSDM)	Divides projects in three phases: pre-project, project life cycle, and post project. 9 principles underlie it: user involvement, empowering the project team, frequent delivery, addressing current business needs, iterative and incremental development, allow for reversing changes, high-level scope being fixed before project starts, testing throughout the lifecycle, and efficient and effective communication (Stapleton, 2003).
3	Feature-driven development	Combines model-driven and agile development with emphasis on initial object model, division of work in features, and iterative design for each feature. Claims to be suitable for the development of critical systems. An iteration of a feature consists of two phases: design and development (Palmer & Felsing, 2001).
4	Lean software development	An adaptation of principles from lean production and, in particular, the Toyota production system to software development. Consists of seven principles: eliminate waste, amplify learning, decide as late as possible, deliver as fast as possible, empower the team, build integrity, and see the whole (Poppendieck & Poppendieck, 2003).
5	Scrum	Focuses on project management in situations where it is difficult to plan ahead, with mechanisms for ‘‘empirical process control”; where feedback loops constitute the core element. Software is developed by a self-organizing team in increments (called ‘‘sprints”), starting with planning and ending with a review. Features to be implemented in the system are registered in a backlog. Then, the product owner decides which backlog items should be developed in the following sprint. Team members coordinate their work in a daily stand-up meeting. One team member, the scrum master, is in charge of solving problems that stop the team from working effectively (Schwaber & Beedle, 2001).
6	Extreme programming (XP)	Focuses on best practice for development. Consists of twelve practices: the planning game, small releases, metaphor, simple design, testing, refactoring, pair programming, collective ownership, continuous integration, 40-h week, on-site customers, and coding standards. The revised ‘‘XP2” consists of the following practices: sit together, whole team, informative workspace, energized work, pair-programming, stories, weekly cycle, quarterly cycle, slack, 10-minute build, continuous integration, test-first programming, and incremental design. There are also 11 ‘‘corollary practices” (Beck & Andres, 2004).

There are several motivations to adopt an agile development methodology. According to VersionOne (2015) who conducted a survey on the motivation to adopt agile, the order by priority is: accelerate product delivery, enhance ability to manage changing priorities, improve business and IT alignment, increase team productivity, enhance software quality, improve project visibility, reduce project risk, simplify development process, reduce project cost, improve team morale, increase software extensibility, improve engineering discipline, better manage distributed teams, and enhance delivery predictability. Tripp and Armstrong (2014) categorized the motivation factors into three factors of motivation, based on the analysis of a principal component factor (PCF) analysis and varimax rotation with Kaiser normalization. These factors are labeled as motivation to improve software quality, motivation to improve efficiency, and motivation to improve effectiveness. Table 2.2 list these factors with a corresponding agile adoption motivation.

Table 2.2
Agile Adoption Motivation

No.	Motivations	Factors
1	Enhance Software Quality	
2	Improved/increased engineering discipline	Software Quality
3	Enhance software maintainability / extensibility	
4	Increases productivity	
5	Accelerate time-to-market	Efficiency
6	Reduce cost	
7	Enhance ability to manage changing priorities	Effectiveness
8	Improve alignment between IT and business objectives	
9	Manage distributed teams	
10	Agile Adopted Practices	
11	Reduce Risk	
12	Improve Project Visibility	
13	Manage Distributed Teams	
	(VersionOne, 2015)	(Tripp & Armstrong, 2014)

2.2.1 Agile Practices

Subsequently, each agile development methodology consists of several practices and processes where the team can choose among them to support their goals. Each practice helps the organization to achieve agile principles in a methodology (Campanelli & Parreiras, 2015). In an agile development methodology, there is a big pool of available practices to use. Diebold and Dahlem (2014) defined the agile practice as a small and very specific part of a method that addresses different aspects. Therefore, there is no common literature definition of agile practices (Diebold & Dahlem, 2014).

Agile practices aim to support agile development as well as the values proposed in the agile manifesto (Yang, Liang, & Avgeriou, 2016). Pair Programming and Retrospective are examples of agile practices. According to Williams (2010) there are 32 agile practices that are related to well known agile methodologies (i.e.: XP, Scrum, feature driven development). Table 2.3 lists the agile practices. Moreover, Jalali and Wohlin (2010) conducted a mapping study on agile practices that show there are 25 agile practices based on available literature. Furthermore, Abbas, Gravell, and Wills (2010); VersionOne (2013) conducted different surveys and identify 26, 58 agile practices, respectively. Table 2.3 shows the current agile practices by Jalali and Wohlin (2010); VersionOne (2013); Williams (2010). Some practices have different names in different books and references.

Agile methodology could be categorized into: management practices, software process practices and software development practices (Lee & Yong, 2013). There is no study that systematically summarizes, analyzes, and classifies all existing agile practices (Yang et al., 2016). An example of management practices are: on-site customer, daily stand-up meetings and open work area. The software process practices include simple design and collective code ownership. Pair programming and unit testing are examples of software development practices.

Particularly, agile methodology is not adopted entirely but it is adopted in certain practices. Furthermore, some practices are used more frequent on some business domains (Diebold & Dahlem, 2014). It is also known that project success depends on the choice of agile practices (De Souza Mariz, França, & Da Silva, 2010). A further key attribute which renders a method amenable to tailoring is the extent to which its individual component practices are independent, allowing them to be separated or combined without fear of unknown subsequent effects (Conboy & Fitzgerald, 2010). The evidence of how agile development is carried in practice, practice selection and the impact is relatively nascent and weak (Dingsøyr, Dybå, & Abrahamsson, 2008).

2.2.2 Agile Practice Cluster

The term "working set" is used to refer to the set of agile practices that bring out the positive effect in a project (Krzanik, Rodriguez, Similä, Kuvaja, & Rohunen, 2010). It was defined as "a restricted set of such top important practices, values and goals" (Krzanik et al., 2010). Also, it is known as agile practice clusters, as in Conboy and Fitzgerald (2010); Kurapati, Manyam, and Petersen (2012). However, evidence of which agile practices working set work well together, and in what contexts, is missing (Saripalli & Darse, 2011).

Most projects adopted an incremental approach, starting with a few practices but never actually getting beyond a few key ones (Conboy & Fitzgerald, 2010). In many cases only a minority of practices are actually implemented. As in the Conboy and Fitzgerald (2010) study, which found that only a quarter of the studied projects were adopting more than half the practices of the XP methodology. The previous study recommends an investigation into the practices cluster to determine if the practices are codependent.

Table 2.3
A List of Agile Development Practices

Williams (2010)	**Jalali and Wohlin (2010)**	**VersionOne (2013)**
Stand-Up Meeting	Stand-Up Meeting	Daily standup
Short Iterations	Short Iterations	Short iterations
Code and Tests	Unit Testing	Unit Testing
Retrospective	Retrospective	Retrospectives
Release and Iteration Backlog	Backlog	Prioritized backlogs
Scrum Meeting	Burndown chart	Team-based estimation
Negotiated Scope	Planning Game	Agile Games
Collective Code Ownership	Coding standards	Coding Standards
Continuous Integration	Continuous Integration	Continuous Integration
Sustainable Pace	Sprint demo	Iteration reviews
Whole Team	Scrum of scrums	Iteration reviews
Iteration Demonstration	Planning meeting	Single team
Sprint	Refactoring	Refactoring
Sit Together	Close Collaboration	Open Work area
Test-Driven Development	TDD	TDD
Informative Workspace	Virtual Scrum Wall	Digital Task board
Stories	User Stories	Digital Task board
Nightly Build	Feature-driven development	Kanban
Code Ownership	Instant message	Collective Code Ownership
Pair Programming	Pair Programming	Pair Programming
Acceptance Test	Acceptance test	Automated Acceptance Testing
Wideband Delphi Estimation	Proxy customer	Iteration Planning
Ten-Minute Build	Automated Testing	Continues Deployment
Energized Work	Sprint review	Release Planning
Features	System Metaphor	Taskboard
Incremental Design	Code Review	BDD
Inspections		
Planning Poker		
Executable Documentation		
Short Releases		
Done Criteria		

Commonly, an agile adopter need assistance in order to choose the proper combination of agile practices for them (Campanelli & Parreiras, 2015).

Many XP adopters believe that XP practices could be clustered, and have attempted to construct or figure out those clusters in their project (Conboy & Fitzgerald, 2010). Another study organizes the practices into several clusters, to try to figure out the practices which were used together (Kurapati et al., 2012). The study was a qualitative study which was conducted globally on an organization level as well as the project level. It was included in different countries, and different domains of organization. The previous study conducted a hierarchical cluster analysis to generate new practice clusters. The most common practices 'clusters' on a project level was: clusters 1 (stand-ups, Sprint/iteration), clusters 3 (testing, stories/features), clusters 6 (continuous integration, short releases), and clusters 5 (stand-ups, sprint/iteration, and sprint planning meeting).

Likewise, Abbas et al. (2010) indicated 15 agile practices clusters. The cluster was generated using principal component analysis with oblique rotation on 58 agile practices. The new cluster includes: agile quality assurance, communication (team), communication (customers), coding standards, etc. In other ways, Tripp and Armstrong (2014) categorized 12 agile practices into two clusters using qualitative methods. Three experts were interviewed in order to generate new clusters. The new clusters are: the project management cluster (include: release planning, iteration planning, velocity, daily stand-up, retrospectives, and burndown), and the software development approach (includes: TDD, refactoring, continuous integration, unit testing, coding standards, and automated builds). de O. Melo et al. (2013) categorized agile practices into three clusters which are technical, management, and collective knowledge sharing. Technical practices are pair programming, burndown chart, and automated acceptance tests.

Management practices are daily meeting, iteration development, iteration/release planning, retrospectives, checklists, one-on-one meetings, and timeboxe. Similarly Lee and Yong (2013) grouped agile practices into three groups which are management practices cluster, software process cluster, and software development practices cluster.

Conboy and Fitzgerald (2010) found that none of the project teams included in the study had managed to identify any such clusters. Based on that, they suggested conducting a quantitative study to determine correlations between adopted practice clusters and either: the effectiveness of practices within the cluster or project success. That would help to identify existing agile practices cluster on practice.

2.2.3 Agile Adoption

According to Nerur et al. (2005), the adoption of agile methodology is not an easy task, despite the implementation of most of its practices is easy. That happens because the agile adoption represents an organizational change that will affect the company's organizational structure, processes, as well as people's behavior (Campanelli & Parreiras, 2015). Therefore, it requires a carefully thought-out preparation (Nerur et al., 2005). Likewise, Ayed et al. (2014) agreed that adopting agile software development methodology needs a wide and complex organizational change that usually impacts several aspects of the organization (e.g., its structure, culture, management practices, produced artifacts, technologies in use, etc.). In order to successfully handle the several key challenges, it is crucial to understand the organization context and carefully study the transformation strategies (Ayed et al., 2014).

The biggest challenge in adopting ASDM, therefore, is choosing an appropriate method from the pool of available agile methodologies (Nerur et al., 2005). Many studies indicated that using agile methodology as defined in the literature does not work very well in real practice (Diebold & Dahlem, 2014). Because the companies do not adopt the complete methodology; they only adopt selected processes or practices and select whatever process that support their objectives (Coleman & O'Connor, 2008; Diebold & Dahlem, 2014). Adopting the agile whole practices as it is proposed in methodology will cause the organization to spend more effort and resources in adopting it (Campanelli & Parreiras, 2015).

A recent study by Diebold and Dahlem (2014) reported that the agile processes are selected based on the domain of the company. The organization factor is an important factor for successful adoption (Strode, Huff, & Tretiakov, 2009). Another study identified the success factors in adoption agile methodologies, includes: customer commitment, customer stratification, customer collaboration, corporate culture, societal culture, decision time, control, personal characteristics, and training (S. C. Misra, Kumar, & Kumar, 2010).

Nerur et al. (2005) determined the key issues and challenges to adopt agile was divided into four groups: management and organizational factors, people factors, process factors, and technology factors. Furthermore, VersionOne (2013) identified the most challenges during agile adoption, which are: inability to change organization culture, general resistance to change, management support, and lack of employee skills. Also, the study concluded the biggest pitfall for adoption agile was: lack of up-front planning, and loss of management control. Gandomani, Zulzalil, Ghani, Md. Sultan, and Sharif (2014) indicated the most important attention within agile adoption process are: focusing on people, providing an action plan, transition challenges identification during, providing prerequisites, providing facilitators, and timely assessment.

As a consequence, adoption of agile methodology should to be well planned to be successful. However, the agile adopters encounter a problem because of lack of guidance and assistance (Ayed et al., 2014)). According to (Boehm, 2002), in practice, the adoption should be only on practices that may be beneficial and applicable in a specific organizational context, which include: development effort, development team size, etc. (Ayed et al., 2014). Cao and Ramesh (2008) asserted that companies looking for implementing agile methodology and practices do not have the same level of consideration towards the existing agile practices.

2.2.4 Agile Method Tailoring

In the software development field methods tailoring is the process of adapting the method used to meet the circumstances of use (Sommerville & Ransom, 2005). Also, it could be defined as the adaptation of the method to the aspects, culture, objectives, environment and reality of the organization adopting it (Campanelli & Parreiras, 2015). Furthermore, according to Beck (1999) every project has its own processes that are tailored for its circumstances and needs. So, when adopting agile development methodology, the organization must consider tailored practices based on the current needs. In summary, the agile adoption should be iterative and the practices should be tailored and selected based on the needs of organization. Indeed, the company selects the practices they need only (Kurapati et al., 2012), Sometimes this scenario is referred as partial adoption of agile methodology (Erickson et al., 2005). As there is a long list of available agile practices, the agile teams need help in order to choose the right combination of practices based on their needs (Abbas et al., 2010).

Based on assumption that there is no software development method that's catered perfectly for specific organization, much research focuses on studies regarding the agile methods tailoring (Conboy & Fitzgerald, 2010). According to Papatheocharous and Andreou (2013) there are a few surveys conducted on agile adoption and the factors that affect it. The agile practice adoption in organizations differ from organization to another, depending on different factors which includes: the development process the organization wants to improve (Campanelli & Parreiras, 2015). Abrahamsson, Oza, and Siponen (2010) state there is no solution that fits every project, but rather the practices should be tailored to suit the needs of each projects. Beck (1999) suggested not to try to adopt all practices at once, rather just find the problem and try to solve it using XP practies. The importance of understanding agile methodology tailoring is to enable companies to selecting practices to achieve the organization needs, since full agile method adoption can be an overkill for organizations or require lots of resources (Qumer & Henderson-Sellers, 2008).

Rodríguez, Markkula, Oivo, and Turula (2012) found tailoring agile or lean practice is one of challenges facing the agile adopter. By understanding the link between ASDM motives and adopted practices, it could provide the adopter with more information on mechanism to fit their method tailoring process (Tripp & Armstrong, 2014). Abrahamsson et al. (2010) asserted that there is a lack of instruction on how to tailor XP practice during the adoption. Xu and Ramesh (2007) summarized the different factors that influence software development which includes: process goals and the specific software process activities or tasks that can be used to meet these goals.

Kalus and Kuhrmann (2013) determine 49 criteria for tailoring software development methodology based on systematic literature review (SLR) analysis, and concluded that criteria significantly impact the resulting software process. These criteria are grouped as a team, internal environment, external environment, and objectives factors. Furthermore, Diebold and Zehler (2015) listed the organization factor impact adoption, as: increasing time to market, and improving the product quality by reducing the number of defects. Additionally, Sison and Yang (2007) connected several XP practices with three factors, that includes: increased productivity, improved teamwork, and reduced overtime. There are a few studies conducted on the impact of practices as individual (e.g., pair programming) or common combination of two practices (e.g., Impact of Pair Programming and Test-Driven Development) (Diebold & Zehler, 2015). It is important to have a link between the improvement goals and agile practices (Diebold & Zehler, 2015).

2.3 Software Startups

The words 'startup' is not a well-defined term, and it is difficult to find an agreement on its definition (Giardino et al., 2014). There are different approaches to defining startups, which includes: company age, or company size. This approach is more generic. It is difficult to set boundaries for ages and size, because that varies from place to place. In this section, the focus is to define 'startup' in the software engineering context. An introductory use for startups term in the software engineering literature can be found in paper written by Carmel on 1994 titled "Time-to-completion in software package startups". This paper has conducted a study on twelve young software firms (Carmel, 1994). After that date, the use of the term startups increased slightly. There are different definitions for startups to be found, but based on different criteria. Based on the literature review the researcher found 6 different definitions for startups in the software engineering context.

Bach (1998) did an early definition for startups in 1998, where he defined startups based on two main criteria: first, he describes it as a team who has high energy and commitment. The second criterion was based on development practice, where the startups are defined to use ad-hoc process. Two years latter, Sutton (2000) defined startups based on three different criteria, as a company without established product, customer base, or revenue. In his paper "The role of process in software start-up" Sutton conducted a study on the EC Cubed company, which was more than three years old with more than 100 employees. Therefore, it is clear that based on Sutton (2000) definition, the company that is more than three years old are treated as startups. As well as the company with more than 100 employees is treated as startup. According to Sutton's definition, he did not state the limit for number of employees in startups.

In another study, Mobile Navigation is a small Swedish company, it was a case study in the study conduct by Kajko-Mattsson and Nikitina (2008). It was a two year old company in the time of study, and it was treated as a startup by the authors. The company had 16 employees only, in matter of size.

Crowne (2002) defined startups as an early phase on company life ended with the first sale. He focuses on his definition on the company age only. This definition is too simple, because it ignores other criteria. Reis (2011) defines the startups based on two new criteria; first, startups which developed new products. Second, startups that worked under a highly uncertainty environment. Also, Reis' definition ignores the size and age factors.

Giardino and Paternoster (2012) defined startups as a temporary organization that focuses on the creation of high-tech and innovative products, with little or no operating history, aiming to grow by aggressively scaling its business in highly scalable markets. This definition is interesting, because it includes different characteristic of startups. The final definition found is by Paternoster et al. (2014), where they defined startup as a small company exploring new business opportunities, working to solve a problem where the solution is not well known and the market is highly volatile. Being established recently is not enough to be a startup (Giardino et al., 2014). There are different criteria to define any company as a startup. Table 2.4 list the startup's definitions on software engineering context based on literature review.

Table 2.4
Software Startups Definitions

No.	Author (s)	Definition
1	Bach (1998)	A bunch of energetic and committed people without a bunch of defined development processes.
2	Sutton (2000)	Startups are companies without an established product, customer base, or revenue stream.
3	Crowne (2002)	The startup phase defined as the period between product conception and the first sale.
4	Reis (2011)	A startup is a human institution designed to deliver a new product or service under conditions of extreme uncertainty.
5	Giardino and Paternoster (2012)	Software startups we refer to those temporary organizations focused on the creation of high-tech and innovative products, with little or no operating history, aiming to grow by aggressively scaling their business in highly scalable markets.
6	Giardino et al. (2014)	Startup is a small company exploring new business opportunities, working to solve a problem where the solution is not well known and the market is highly volatile.

From the previous definitions, it is clear that different definitions are from different perspectives. That makes it difficult to set a clear definition. The researcher tries to analyze the current theme for a software startup to extract the main criteria of software startups, which will be helpful in setting up a proper definition for it. Based on the analysis of literature, there are 14 software startups criteria are founded.

Previous studies Bosch, Holmström Olsson, Björk, and Ljungblad (2013); Coleman and O'Connor (2008); Crowne (2002); Giardino et al. (2014); Sutton (2000); Voas (1999); Yau and Murphy (2013) showed that limited recourses is one of the primary criteria that are common for startups. A small sizes team is the second most important criteria, and many studies addressed that the team size is one of main characteristics used to define software startups. In consideration of the small team the term is vague, it is difficult to set a boundary for small team size because this is different from place to place. The team size boundary is discussed in software engineering in the organization section. The software startups themes are listed in Table 2.5.

Table 2.5
Software Startups Themes

No.	Theme	Authors
1	Lack of resources	Bosch, Holmström Olsson, et al. (2013); Coleman and O'Connor (2008); Crowne (2002); Giardino et al. (2014); Sutton (2000); Voas (1999); Yau and Murphy (2013)
2	Small teams	Bosch, Holmström Olsson, et al. (2013); Coleman and O'Connor (2008); Crowne (2002); Giardino et al. (2014)
3	Fast-growing markets	Crowne (2002); MacCormack (2001); Paternoster et al. (2014); Sutton (2000)
4	Product-driven	Coleman and O'Connor (2008); Giardino et al. (2014); Marmer et al. (2012)
5	Rapid evolution (Requirement change)	Bosch, Holmström Olsson, et al. (2013); Crowne (2002); Giardino et al. (2014); Yau and Murphy (2013)
6	Creative, dynamic, and innovative	Giardino et al. (2014); Kelly and Culleton (1999); Sutton (2000)
7	New company	Giardino et al. (2014); Sutton (2000)
8	Time-pressure	MacCormack (2001); Paternoster et al. (2014)
9	High uncertainty	Giardino et al. (2014)
10	Highly risky project	Giardino et al. (2014)
11	High commitment and energy	Bach (1998); Crowne (2002)
12	Exploratory in nature	Bosch, Holmström Olsson, et al. (2013)
13	Rapid release for product	Coleman and O'Connor (2008)
14	All employees are involved in all aspects of project	Kelly and Culleton (1999)

From the analysis of the six startups definitions in Table 2.4, and the analysis of the startups themes in Table 2.5; a software startup is defined for the purpose of this research as a new small company with limited resources that aims to find creative solutions for a problem, in order to gain a high growth rate in revenue bases.

2.3.1 Software Development in Startup

It is easy to find the relationship between startup and agile methodology as they share some key goals (Giardino et al., 2014; Murray, 2008; Rodríguez et al., 2012; Williams & Cockburn, 2003; Yau & Murphy, 2013) which includes both of them focusing on the final product rather than the process or documentation. They also focus on a fast response to change and a speedy release of products. According to Yau and Murphy (2013), in small-scale tech startups, it is clear that the nature of agile methodologies makes it fit many needs required by the startup to help it succeed. Moreover, agile methodology is developed to accept higher rates of changes as well as promote a faster response to a changeable requirement by focusing on people rather than on processes, in contrast with the traditional methodology, which makes choosing an agile methodology more appropriate for startups (Dybå & Dingsøyr, 2008; Williams & Cockburn, 2003).

Many studies indicated that the software development in startup companies is different from the traditional companies. Software startups present a unique combination of characteristic, which pose several challenges to software development activities (Paternoster et al., 2014). From a software engineering perspective startups are unique, since they develop software in a context where processes can hardly follow a prescriptive methodology (Coleman & O'Connor, 2008). Robinson and Sharp (2005) studied three types of organizations. The three types of organizations were a large, medium and small startups, respectively, where all these companies adopt XP. The study found some practices applied on the startup company may not be applied to other case studies.

The software startups merge between many different areas, including: startups, project management, and software engineering in small companies. By mixing all of these areas, the problem could be investigated. Regrettably, Sutton (2000) stated that startups are neglected on research related to software processes. After 14 years, Sutton notes a recent research that declares that the gap is not filled yet by the new researches (Giardino et al., 2014; Paternoster et al., 2014).

Coleman and O'Connor (2008) were conducting a study to figure out: the development process formation in Irish software startups, the current agile processes used on startups, and the factors that influence the make up of these processes. This study was used a grounded theory methodology in order to describe the experiences of software startups in developing processes to support their software development activities. The finding of the study was: 1) there is not enough recourse for startups to investigate for better development methodology 2) software startups depend on the earlier experience of the person tasked with managing the development to select the initial methodology for startups. Other influencers include the market sector in which the company is operating, the style of management used and the size and scale of the company operations. The researchers recommended conducting similar studies on different geographical locations.

ASDAM is used by software startups widely, because of its nature to accept the change in requirement, that always occur in startups Taipale (2010). For instance, a short release is an agile practice that enables companies to have fast and frequent delivery as well as address the challenging market K.Flora, V. Chande, and Wang (2014); Nerur et al. (2005). Frequent delivery is another reason why software startups choose agile methodology (Ambler, 2002; da Silva et al., 2005). Coleman and O'Connor (2007) found that the most used development methodology in Irish startup companies is XP, because of its light weighted process and low documentation work. Several studies found the startups do not strictly follow specific methodology, but they implement different agile practice from different methodologies (Bosch, Olsson, Björk, & Ljungblad, 2013; Coleman & O'Connor, 2007, 2008).

Most of software startup fail within first two years because of many reasons, which includes failure on the execution of sales, failure on delivery to the market, and failure in software development (Crowne, 2002). The weakness on resources leads project managers to do an ad-hoc selecting for development tools, based on their experiences, but not based on the project's current needs and growth needs (Coleman & O'Connor, 2008). Because startups suffer from lacking of resources, time, and money, the priority for startups is to develop a minimum viable product (MVP). MVP is defined as a product with only basic features and functionalities that can be released as early as possible, which could lead to lack of quality (Yau & Murphy, 2013). Also, the developers on startups have a constraint on developing the product, therefore neglecting the agile process (Yau & Murphy, 2013).

On 2000, Sutton declared there is a gap on studies that focus on startups in a software engineering context (Sutton, 2000). Recently, a study done by Paternoster et al. (2014) have mapped 1769 articles that used startups in a software engineering context, but they state the gap is only partially filled.

2.3.2 Software Engineering in Small Organizations

In 1975, the term "programming in the small" was used to differentiate the development characteristics between small and large companies (Fayad, Laitinen, & Ward, 2000). Currently, small software organizations represent up to 85% of all software organizations (Richardson & Von Wangenheim, 2007). For example 70% of software companies established in Brazil are small companies (von Wangenheim, Anacleto, & Salviano, 2006). Despite a significant portion of software companies being categorized as a small and medium enterprise (SME), they face many challenges on software development, because of their size (Coleman, 2005).

The size of the company is the primary measurement to define the small companies. The size is defined based on the industry. In general it is defined to be between 100 to 500 as a maximum (Fayad et al., 2000). According to Cater-Steel, Toleman, and Rout (2006), small companies have less than 20 employees; and medium companies have between 20 to 200 employees. Other studies defined small software companies as less than 50 employees (Fayad et al., 2000; von Wangenheim et al., 2006). Different challenges for small companies were discussed by Richardson and Von Wangenheim (2007).

Richardson and Von Wangenheim (2007) found that despite the large portion of small companies in the software industry, only a few researches were conducted in this area. Software engineering in small organizations present a lack of processes and basic documentation (Valtanen & Ahonen, 2008), although they are recognized as important aspects of software development (Martin & Hoffman, 2007). Software process improvement (SPI) is almost neglected, due to the amount of time required to establish all the configuration management and review processes (Caffery, Taylor, & Coleman, 2007). Therefore, the software engineering in a small organization shares some characteristics and challenges for startups.

2.4 Theoretical Framework

According to Dingsøyr et al. (2008) the theories that are currently in use in information systems include: task-technology fit, theory of planned behavior, contingency theory, complexity theory, social learning theory, social network theory, socio-technical theory, organizational learning theory, and the knowledge-based theory of the firm. All of the previous theories are used in information systems and relevant to agile software development. Furthermore, Tripp and Armstrong (2014) supported that contingency theory and fit theory emerge in the information system studies.

Based on the review of available literature, it is hypothesized that agile adoption motivation factors influence adoption of agile practices clusters. Specifically, this framework postulates that agile adoption motivation factors are related to agile practices which are in line with previous studies by Tripp and Armstrong (2014). In these study contingency theory is used to explain the relationship between agile adoption motivation and agile practices' clusters.

Contingency theory is categorized as a class of behavioral theory. It is an organizational theory that claims that there is no best way to structure an organization. Instead, the optimal way is dependent on numerous of internal and external variables (Islam, 2012). These variables include the organization size, adoption process, available resources, operations activities, human resources, technology, culture, etc. (Islam, 2012).

Furthermore, contingency theory is an approach used to study organizational behavior in which explanations are given as to how contingent factors influence the design and function of organizations (Islam, 2012). This theory is in used under a broad range of topics as well as in different forms and implementation (Hanisch & Wald, 2012). Recent studies postulated that organizational structure was contingent on contextual factors such as technology, dimensions of task environment and organizational size (Islam, 2012).

In software engineering, recent studies indicate that software development projects are unique depends on internal and external factors (Conboy & Fitzgerald, 2010). However, ASDM tailoring suggests that, no matter how well crafted, there is no single method that provides an exact fit for the needs of every project (Conboy & Fitzgerald, 2010). In practices, software processes need to be tailored for each specific project, so, it is essential to understand how various environmental factors affect the application of the process (Xu & Ramesh, 2007). In other words, the choice of method or method variant is dependent on organizational, technical, or human factors, and the nature of the system being developed (Conboy & Fitzgerald, 2010).

From this perspective, it clear how this theory could be applied in software engineering. When there is a fit between the environment in which the software development team operates, and the development method used to manage environmental risks, better outcomes are predicted (Tripp & Armstrong, 2014). Xu and Ramesh (2007) define these factors in software engineering context which include the following: characteristics of the customers, development team, technology, and products, as well as the business conditions in which the products will operate.

As indicated previously, contingency theory studies postulate that organizational outcomes are the consequences of a fit or match between two or more factors. Organizations attempt to fit their processes to the understand of the internal and external variables, so it is easy to modify the organization structure and practices (Donaldson, 2001). That helps to explain the relationship of agile adoption motivation factor on the output of agile adoption process (i.e., adopted agile practices). Which mean that the adoption process results are dependent on a fit or match with the adoption factors (e.g., adoption motivation factors). The fit between the environment and the software development method is recognized as a key contingency for software project success (Barki & Suzanne Rivard, 2001).

2.5 Related Works

Schindler (2008) studied the agile software development in Austria. The study focused on using agile practices, and the awareness and adoption of agile methodologies in Austrian IT companies. He conducted a survey of 42 companies. The acceptance rate was 42%, which was treated as a high response rate. The study found there is a lack of agile development method awareness in industry that affect agile adopting.

Asnawi et al. (2012) investigated the adoption and success factors of agile methodology in Malaysia. The perceptions have been identified from their awareness, the way they introduced the method and the challenges they are facing. The study found that agile was still a new and emerging method in Malaysia and the awareness was still low.

Papatheocharous and Andreou (2013) studied the level of agile adoption corresponding to the agile practices that were followed within software companies worldwide. They conducted an online survey in 2012, where its acceptance rate was 6% (377 responses over 5,900). The responses came from several countries, which includes: United States, India, UK, and others. The survey was analyzed by using statistically descriptive methods, like: frequency, and mean. The study found that agile methodology was adopted to enhance the quality of the final product developed, accelerate the time to market, and increase the productivity. Also, it found that the critical factors that affect the adoption of agile development methods, such as: finding it difficult to change the way of working, the management of the organization is opposed to change, and non-familiarity with agile methodologies.

Giardino et al. (2014) investigated the software engineering practices in startups. Their study analyzed the recent research in this area. The study found that the processes need to be tailored to software startups, to support the development of the product. In addition, applying rigid development methodology is not suitable for startups because they have a different culture and need. This study concluded that the rise of the startups phenomena is leading to the rise of the need and challenge to conduct more studies on it.

Mohamed, Baharom, and Deraman (2014) studied the practices and awareness of the agile software development for software practitioners in Malaysia. The study used a questionnaire as a data collection instrument. Purposive sampling was used as a sampling technique. The data analysis was using descriptive statistical methods like: frequency, and mean. The questionnaire is categorized into two sections: 1) Demographic background 2) agile based software development. A total of 73 responses were received, but the response rate was not mentioned. The study found that the participants from Malaysia believe that adopting agile is important to gain a high quality result on the project. The awareness of agile methodologies was high, and 75% of participants have previous experience with agile.

Recently, Tripp and Armstrong (2014) conducted a quantitative study to explore the relationship between the agile adoption motivation and the agile practices tailoring. The study focuses only on individual practices. The study used the VersionOne State of Agile 2011 survey for its source of data which includes different sizes of organization around the world. The evidence found there is a relationship between the agile practices and agile adoption motivations. Furthermore, the study finds three motives for agile adoption which are: a desire for increased software quality, increased efficiency, or increased effectiveness. Each of them are associated with different agile practices.

2.6 Conclusion

The current chapter provides reviews of literature and previous work related to the topic of the thesis which includes a theoretical and empirical review of Agile development methodology, Agile practices, agile clustering, and Agile method tailoring. Also, it discusses the definition of software startups in the context of software engineering. Finally, this chapter lists the works related to this study.

CHAPTER THREE
RESEARCH METHODOLOGY

3.1 Introduction

The purpose of this chapter is to present a research methodology which was used in this study. That includes discussing the following topics: research framework, research design, population, sampling, data collection, and data analysis.

3.2 Conceptual Framework

Based on the analysis of available literature, it is hypothesized that agile adoption motivation factors influence the agile practice clusters selection in organizations. The research framework for this study shows the relationship between agile adoption motivation factors and agile adopted practices. The agile adoption motivation factors make the independent variables for this study. The dependent variable for this study is the agile practices clusters. The framework of this study follows the framework of the Tripp and Armstrong (2014) which concentrates on examining the relationship between the effect of agile adoption motivation factors on the agile practices tailoring. However, Tripp and Armstrong (2014) do not focus on agile practice cluster. The research framework for this study is as shown in Figure 3.1. As shown on Figure 3.1, this study proposes that improved quality, efficiency, and effectiveness motivation factors have a direct relationship with agile practices clusters.

To help explaining the relationship between all factors, this study uses the contingency theory of organizations as the underpinning theory to cover the main part of the framework. Contingency theory suggests that the fit between contextual factors and the design of management control systems is relevant to superior organizational performance (Donaldson, 2001). The fit between the environment and the software development method is recognized as a key contingency for software project success (Barki & Suzanne Rivard, 2001).

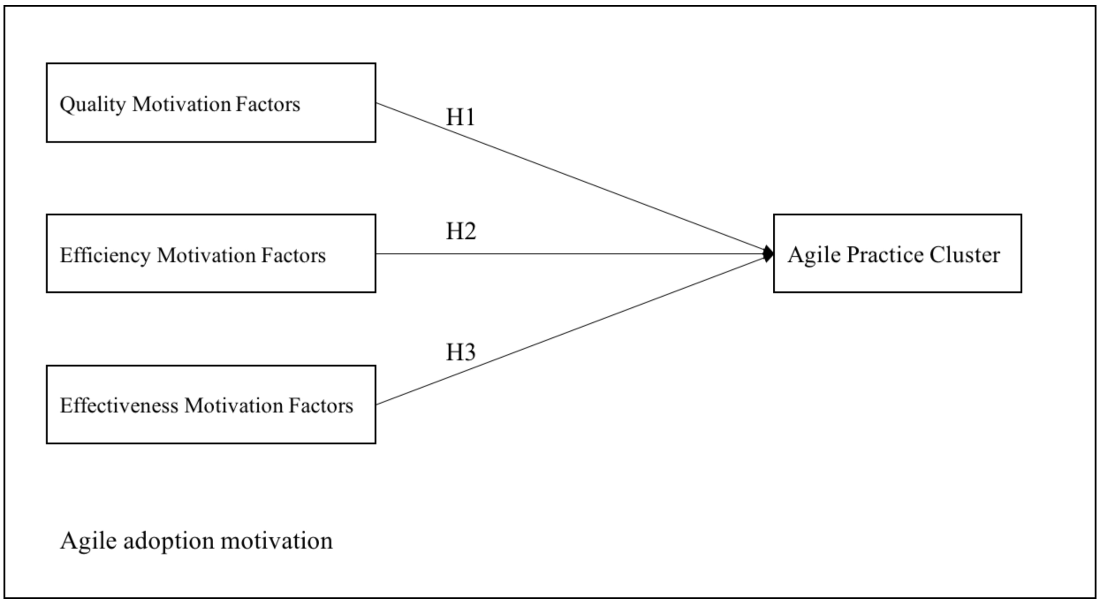
Quality Motivation Factors
Efficiency Motivation Factors
Effectiveness Motivation Factors
H1
H2
H3
Agile Practice Cluster
Agile adoption motivation

Figure 3.1. Research Framework

3.3 Research Hypotheses

This section reviews the hypotheses development regarding the relationship between quality, efficiency, and effectiveness motivation; and agile practice cluster.

H1 Motivation to improve software quality has a positive relationship with adopted agile practices clusters by organizations.

H2 Motivation to improve development efficiency has a positive relationship with adopted agile practices clusters by organizations.

H3 Motivation to improve development effectiveness has a positive relationship with adopted agile practices clusters by organizations.

3.4 Research Design

This study is a correlational study that aims to explain or describe the degree of association among two or more variables (Creswell, 2014). It is a quantitative study, used a questionnaire as the instrument. Questionnaires are one of the main approaches for collecting data to support the academic research in the fields of software engineering (Lethbridge, Sim, & Singer, 2005). As the questionnaire is a timely and cost-effective instrument (Lethbridge et al., 2005), which fit for this study constraint. In a software engineering context, the survey tries to capture what is happening broadly over large groups of projects. It reflects how the population reacts to a particular method, tool, or technique. It helps to determine the trends or relationship (Sjoberg, Dyba, & Jorgensen, 2007). The four main activities, which were used in this study, are: instrument design, defining population and sample, data collection, and data analysis.

3.5 Instrument Design

The instrument was developed as a result of an analysis of previous studies from Mohamed et al. (2014); Tripp and Armstrong (2014); VersionOne (2015). The questionnaire consists of 48 questions. The questionnaire is segregated into three sections: Section One: Demographic questions, Section Two: Agile Adoption Motivation, and Section Three: Agile Adopted Practices. The structure of the questionnaire is summarized in Table 3.1. An online survey questionnaire was developed and formatted using Google Docs.

Table 3.1
Summary of the Questionnaire Design

Section	Items	Questions	Source
One	Demographic data	1-7	Mohamed et al. (2014)
Two	Independent variable: Agile Adoption Motivation	8-22	Tripp and Armstrong (2014)
Three	Dependent variable: Agile Adopted Practices	23-48	VersionOne (2015)

3.5.1 Section One: Demographic Question

This section consists of seven questions to gather the main demographic data about respondents and their organization. The first part is about practitioners background, which includes: current position, education level, and participant experience with ASDM. The second part is about practitioners' organization, which includes: organization age, size, the average size of the development teams, and type of product developed by the organization.

3.5.2 Section Two: Agile Adoption Motivation

This section is designed to assess the agile adoption motivation on the software startups (independent variable), which consist of 15 questions. Respondents were asked to respond to the items by indicating their level of agreement with the statements in the questionnaire using a four point Likert scale (1: Not Important, 2: Somewhat Important, 3: Very Important, 4: Highest Importance) (Tripp & Armstrong, 2014).

3.5.3 Section Three: Agile Adopted Practices

The third section measures the adopted agile practices in the organization (dependent variable), which consist of 26 questions. It is used to measure the use of 25 different agile practices adopted from Tripp and Armstrong (2014); VersionOne (2015).

There are several ways to measure agile practices adoption. Salo and Abrahamsson (2005) used a measuring scale of 5 points, which was: systematically used throughout the project, mostly used throughout the project, sometimes used in the project, rarely used during the project, and never used during the project. The additional two options were added which are: 'not applicable' and 'I do not know'. Another way to measure agile adopted practice is by using a 'yes/no' option as a measurement scale as in the Tripp and Armstrong (2014) study. Based on the previous studies, the scale used for measure agile adopted practices is a five point Likert scale of importance ranging from 1 (Never Used) to 5 (Always Used).

3.6 Population and Sample

Software startup is trend now in KSA, where many startups open every year. Unfortunately, there is a lack of information about the number of existing software startup companies in KSA. This is because the startup community is still emergent in the region.

Based on Saudi Business Incubator Network (SBIN), there are 25 business incubators (Saudi Business Incubator Network, n.d.). Each incubator has on its own portfolio that includes many startups. For example, Badir Information and Communication Technology Incubator, one of the business incubators in KSA, has incubated 64 software startups (Badir Program Technology Incubator, n.d.). Flat6labs is another business incubator located in Jeddah; it has on its portfolio of 22 software startups (Flat6Labs Jeddah, 2015). Some business incubators do not provide information about how many startups are on their portfolio. Based on available incubator data, the estimated software startups are above three hundred startups.

In order to categorize the population, the startups companies were divided into two main categories:

1. Startups that join the business incubators, or business accelerators.
2. Startups, which do not join any incubator or accelerator.

Sampling is a process that represents the population of interest, which is a difficult task in software engineering studies because of the lack of enough demographic data about the population and participants (Lethbridge et al., 2005). A simple random sampling (SRS) is categorized as Probabilistic Sampling Methods, which every member of the selected population had the same probability of being included in the sample (Kitchenham & Pfleeger, 2002). Simple random sample sampling technique was used on this study. SRS has the least bias and offers the most generalizability(Sekaran & Bougie, 2010). As suggested by Sekaran and Bougie (2010), for a population size of 300 (N=300) the appropriate sample size is 175 (S=175).

The sampling unit was an organization where the respondents should be in a software development related position like CEO, CTO, technical team manager, or developer.

3.7 Data Collection Process

The survey was conducted between March and April 2016. The potential respondents were invited through email invitations. There are several reasons to choose a online as a medium for the distribution of survey. First, there is instant access to the data online. Second, it increases the ability to daily track responses. Third, it increases the survey response rates (Nardi, 2002). Moreover, the participants are familiar with the technology, which makes it easy for them to use it. The questionnaire link was sent through email to the main business incubators and accelerators on KSA (See Appendix B) to be distributed to their companies [1].

3.8 Data Analysis

The quantitative data collected was transferred to Statistical Package for the Social Science (SPSS 23.0) software to be analyzed. SPSS is a computer software commonly used by researchers to run statistical analysis. The descriptive statistics used to provide simple summaries about the sample and the measurements such as the demographic

[1]The terms “incubator” and “accelerator” should not be used interchangeably. There is a clear distinction between these two concepts, primarily based on the development stage of the new venture they are serving. see S. Cohen and Hochberg (2014).

profiles. Also, it is used to describe the trends on the instrument, and general tendencies in the data which includes the use of mean, median, mode, and frequency (RQ-1, RQ2). Hierarchical cluster analysis (HCA) was used to identify the most used agile practices' cluster, based on the collected data, to generate agile practice cluster, which answer the RQ-1. Inferential statistics was used to describe the relationship between the independent variable and dependents variable. Pearson's correlation was used for testing the correlation between variables (RQ-3).

3.8.1 Hierarchical Cluster Analysis

In order to find out the similar agile practice clusters adopted by startups, HCA was conducted on the agile practices. The main purpose of cluster analysis is to find related items in a dataset (Kaufman & Rousseeuw, 2009). Cluster analysis (CA) is a convenient method for identifying homogenous groups of objects called clusters that share many characteristics, but are very dissimilar to objects that do not belong to that cluster (Mooi & Sarstedt, 2011).

Mooi and Sarstedt (2011) suggested several steps of conducting CA, which are followed on this thesis (see Figure 3.2). The first step was to determine the variables to be used in CA. Different agile practices were used as variables in this analysis. The second step was to choose the clustering procedure. Based on literature review, this study used agglomerative hierarchical cluster analysis to determine the practices clusters (Manyam & Kurapati, 2011). The third step was to select a measure of similarity or dissimilarity. Squared Euclidean distance is used as a measure in HCA. Euclidean distance is mostly used when variables are in ratio or are interval-scaled variables (Mooi & Sarstedt, 2011).

The fourth step was to select an algorithm for clustering. There are different algorithms

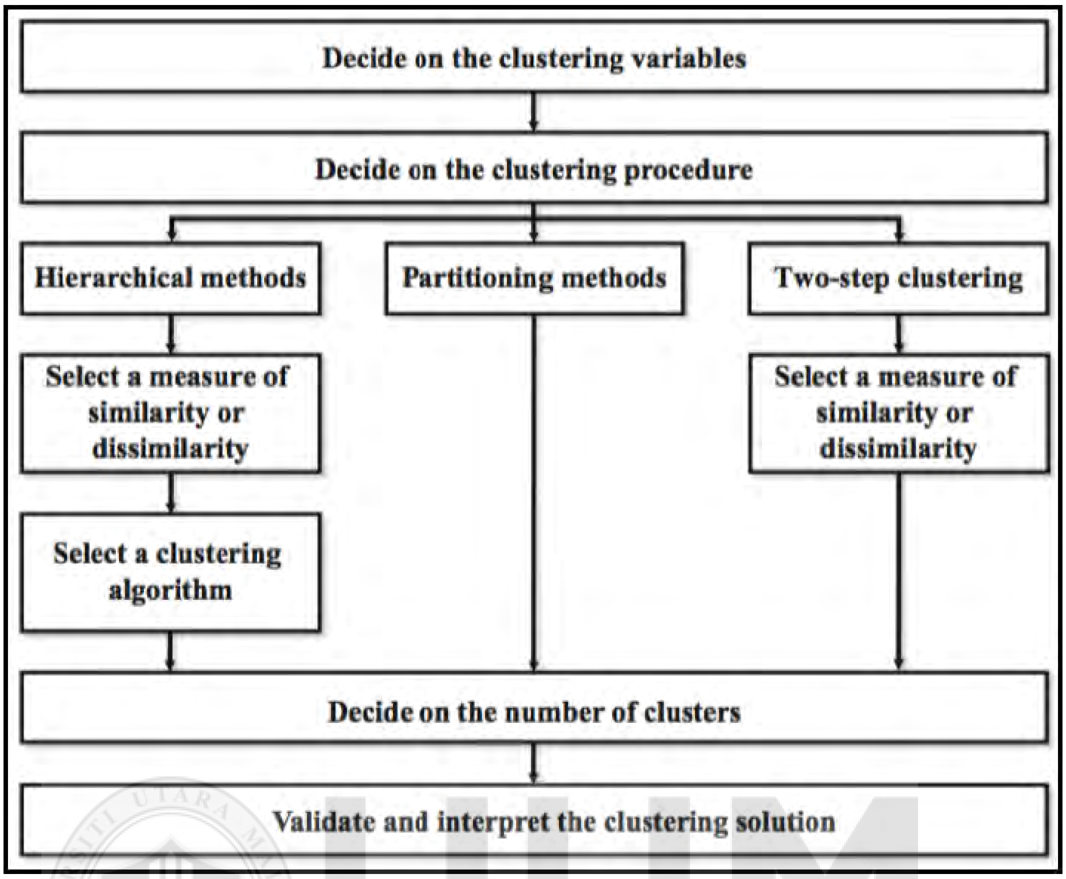
Decide on the clustering variables
Decide on the clustering procedure
Hierarchical methods
Partitioning methods
Two-step clustering
Select a measure of similarity or dissimilarity
Select a measure of similarity or dissimilarity
Select a clustering algorithm
Decide on the number of clusters
Validate and interpret the clustering solution

Figure 3.2. Steps in Cluster Analysis (Mooi & Sarstedt, 2011)

that are used for clustering, and each is used for a different purpose. The method chosen for cluster extraction was Ward's clustering algorithm as it performs well at recovering clusters (Finch, 2005) and has been very widely used (Mooi & Sarstedt, 2011; Murtagh & Legendre, 2014). Ward's clustering methodology generates somewhat equally sized clusters (Mooi & Sarstedt, 2011).

The fifth step was to decide on the number of clusters. A hierarchical algorithm yields a dendrogram that represents the nested grouping patterns and similarity levels at which the groupings change (Jain, Murty, & Flynn, 1999; Mooi & Sarstedt, 2011). Hence, agglomerative hierarchical clustering was a suitable way of composing the new clusters. A dendrogram was used to determine the number of clusters, which were named as distance-based decision rules. In SPSS, a dendrogram rescales the distances to a range of 0–25 (Mooi & Sarstedt, 2011).

3.9 Conclusion

This chapter discussed the research methodology employed in the quantitative study. It described the methodological approach used to collect and analyze the information relevant to the research objectives. In addition, this chapter elaborated on the design of research, the population and sample, measurement of variables, and data collection procedures. Besides that, this chapter discussed the description of data analysis and statistical techniques used to analyze the data.

CHAPTER FOUR
FINDINGS

4.1 Introduction

In this chapter, an analysis of the data collected will be presented. After data was collected through a survey, it was analyzed in an attempt to deduce findings. The first step was data screening to ensure that the data was clean. Second, the demographic data of respondent as well as their organization is presented. Finally, descriptive statistic and correlation analysis was conducted on in order to come up with results.

4.2 Response Rate

A questionnaire was sent to 25 business incubators on KSA. The complete list of incubators is shown in Appendix B. 175 questionnaires were distributed to software startups inside those incubators where 76 responses were received, which represents a response rate of 43%. As software engineering surveys suffer from low response rate (average of 5%), any response rate higher than 5% is acceptable for rigorous statistical analysis studies (Lethbridge et al., 2005). Also, according to Sekaran and Bougie (2010), response rate of 30% or higher is considered acceptable.

4.3 Data Preparation

Data preparation stage come after data has been collected. It includes coding data, keying in data, and editing data (Sekaran & Bougie, 2010). Before data is keyed in, variables must be coded. The total variables were 49. Next, the data collected was transferred from Google forms into SPSS 23.0. In the questionnaire required that the respondents fill all the fields before submitting them back to avoid instances of missing data. The total respondents who submitted completed questionnaires were 76. As questionnaire contains a control question to identify the software startups that adopt the agile methodology, and to exclude other software development methodologies. After removing software startups that do not adopt ASDM, the remaining subjects are 64 responses. This is in line with Rauf and AlGhafees (2015); Sison and Yang (2007); Sulayman and Mendes (2010) studies where the sampling unit was an organization. After data cleaning steps, only 64 responses were used for further analysis.

4.4 Demographic Analysis

This section discusses the profiles and demographics of the practitioners and their companies. There were 7 demographic questions in the survey. Three of the questions were about the practitioners while the others were about their organizations.

4.4.1 Profile of Respondents

This section discusses the profile of respondents who participated in the survey. There were three demographic questions focusing on the respondent's profile. These questions focused on the current positions of respondents, their education levels, and experience with ASDM.

4.4.1.1 Current Position

The first demographic question was about the current position of respondents in company. The positions of the respondents in this survey are shown in Figure 4.1. As shown in Table 4.1, the majority of the respondents were programmers or developers, representing 38% of all respondents. 20% of the respondents were working in an executive level e.g. Chief Executive Officers (CEOs) and Chief Technology Officers (CTOs). In addition, 14%, 11%, and 13% of the respondents were project managers, system analysts, and IT management (represent middle-level management) respectively. Finally, 5% of the respondents were quality assurance officers or testers.

Table 4.1
Current Position Descriptive Statistic

Position	Frequency	Percentage (%)	Mean	Std. Deviation
Programmer / Developer	24	37.5		
Project Manager	9	14.1		
Quality Assurance/ Tester	3	4.7		
System Analyst	7	10.9	3.25	2.303
IT management	8	12.5		
CTO	2	3.1		
CEO	11	17.2		
Total	64	100		

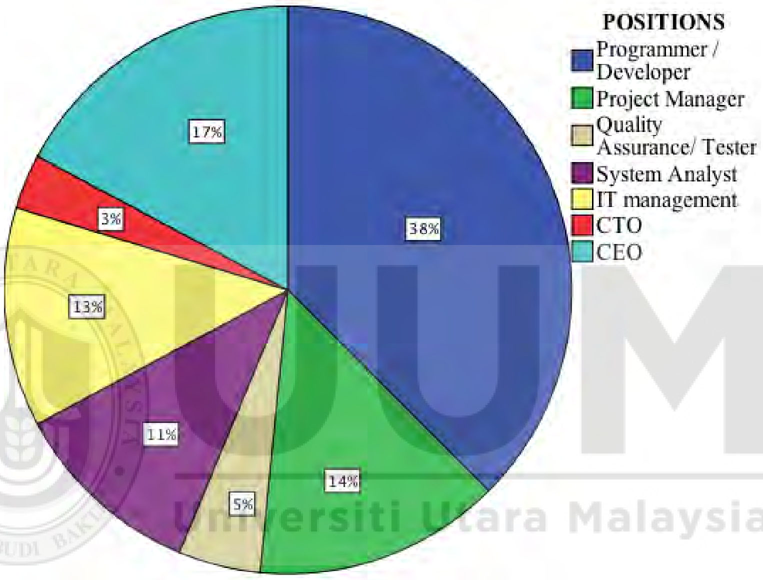

Figure 4.1. Current Position of Practitioners

4.4.1.2 Academic Level

In order to understand the education level of the respondents, participants were asked to provide their highest academic degree. As can be seen in Table 4.2, 38% of the respondents were degree holders. 41% of the respondents had master's degree. Only 6% respondents had attained a Ph.D. 16% of the respondents were diploma and certificate holders. Figure 4.2 shows the education level of respondents.

Table 4.2
Education Level Descriptive Statistics

Academic Level	Frequency	Percentage (%)	Mean	Std. Deviation
Certificate	5	7.8		
Diploma	5	7.8		
Degree	24	37.5	3.30	0.987
Master	26	40.6		
PhD	4	6.3		
Total	64	100		

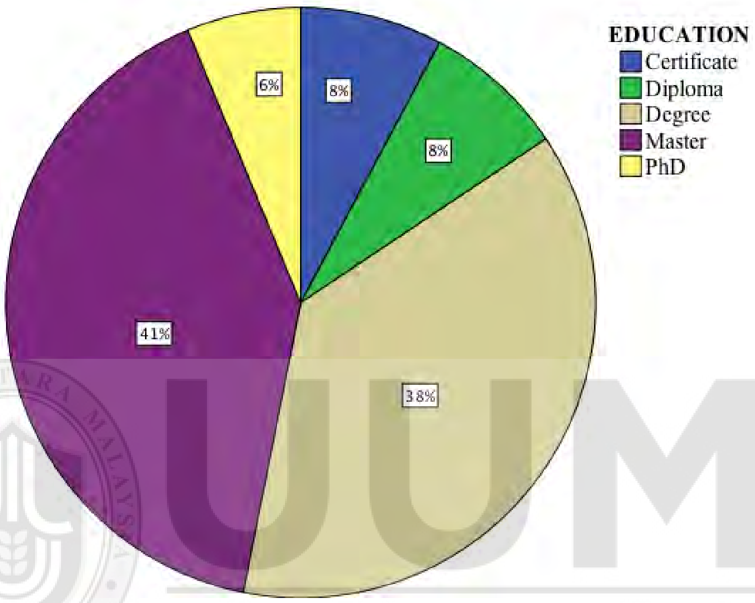

Figure 4.2. Education Level of The Practitioners

4.4.1.3 Work Experience in Agile Software Development

The last demographic question was about how long the respondents have been working with ASDM in order to understand their previous experience with agile development methodology. As can be visually observed in Figure 4.3, the majority of the respondents had little to no experience with agile development methodology. As shown in Table 4.3, 19% of the respondents do not have any previous experience with agile methodology, while 34% of the respondents had less than three years' work experience. 30% of the respondents had 3-5 years' experience while 8% of the respondents had 6-10 years' of experience. Only 9% of the respondents had more than 10 years of agile development experience.

Table 4.3
Participants' ASDM Experience

ASDM Experience	Frequency	Percentage (%)	Mean	Std. Deviation
None	12	18.8		
Less than 3 years	22	34.4		
3-5 years	19	29.7	2.55	1.167
6-10 years	5	7.8		
More than 10 years	6	9.4		
Total	64	100		

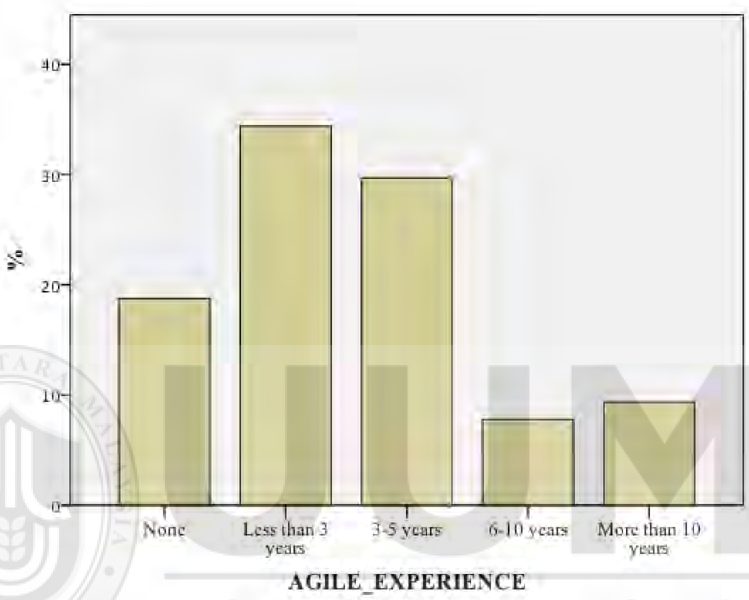

Figure 4.3. Respondents' Agile Experience

4.4.2 Organization Profile

The second section of the demographic questions was about the organization profile. This was important in order to understand the organizational background. The organizations' profiles include age, the number of employees, development team size, product type, and whether or not the organization adopts agile methods. It is important to understand the startup's profile to get insight about startup's profile in KSA.

4.4.2.1 Organization Age

Table 4.4 shows that the majority of the organizations that participated in the survey were in their early phases. 16% of the organizations were less than 1 year old. Majority of respondents (53%) of the organizations were aged between 1 and 4 years. 17% and 14% of the organizations were 5-8 and 8 years old respectively.

Table 4.4
Organization Age Descriptive Analysis

Organization Age	Frequency	Percentage (%)	Mean	Std. Deviation
Less than 1 year	10	15.6		
1-4 years	34	53.1		
5-8 years	11	17.2	2.30	0.903
9 years and above	9	14.1		
Total	64	100		

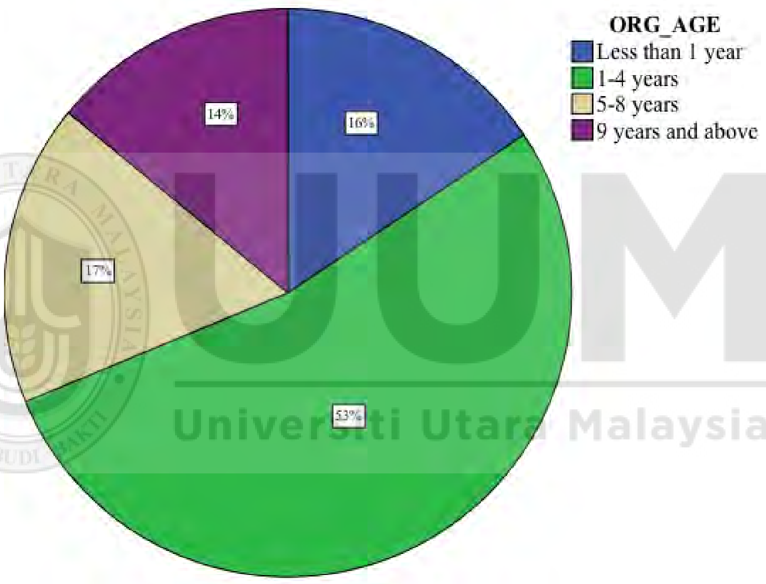

Figure 4.4. Organization Age

4.4.2.2 Organization Size

Table 4.5 shows that more than half (52%) of respondents' organizations had 10 employees or less. Also, 17% of the organizations had 10 to 20 employees, while 25% of the organizations had 21 to 40 employees. Only six percent of the respondents' organizations had more than 40 employees. These can be treated as mature startups.

Table 4.5
Number of Employees Descriptive Analysis

Organization Size	Frequency	Percentage (%)	Mean	Std. Deviation
Less than 10 employee	33	51.6	1.89	1.086
10-20 employee	11	17.2		
21-40 employee	16	25.0		
41-100 employee	2	3.1		
101-500 employee	2	3.1		
Total	64	100		

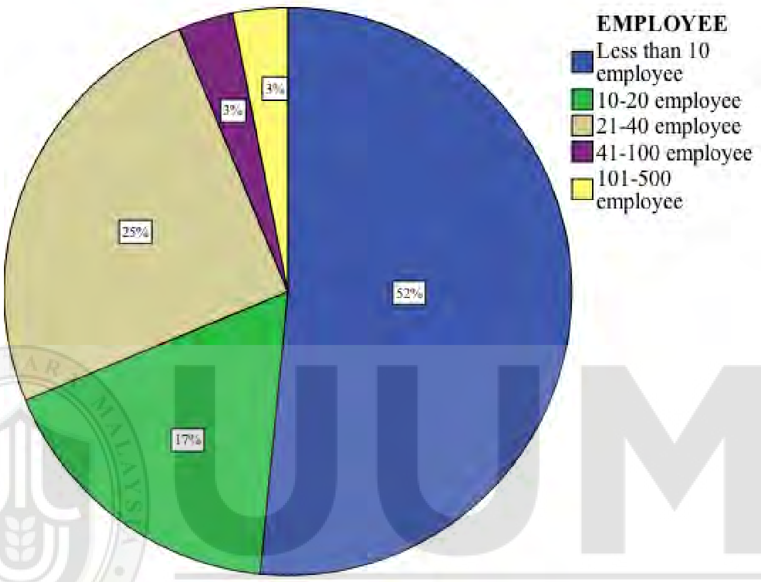

Figure 4.5. Total Employees per Organization

4.4.2.3 Development Team Size

It is important to understand the development team size in startups in order to understand how that reflects on agile adoption. From Figure 4.6, it is evident that majority (64%) of the organization had a small development team size of less than 5 members. As it is a major characteristic for a startup to have a small development team. 25% of the organizations' development teams had 6-10 members. 9% of the organizations had 11 to 20 employees in the development teams. Only one organization (2%) had a big development team with more than 21 employees on the team. Table 4.6 shows descriptive statistic for development team size.

Table 4.6
Development Team Size Descriptive

Development Team	Frequency	Percentage (%)	Mean	Std. Deviation
1-5 employees	41	64.1	1.48	0.734
6-10 employees	16	25.0		
11-20 employees	6	9.4		
21-50 employees	1	1.6		
Total	64	100		

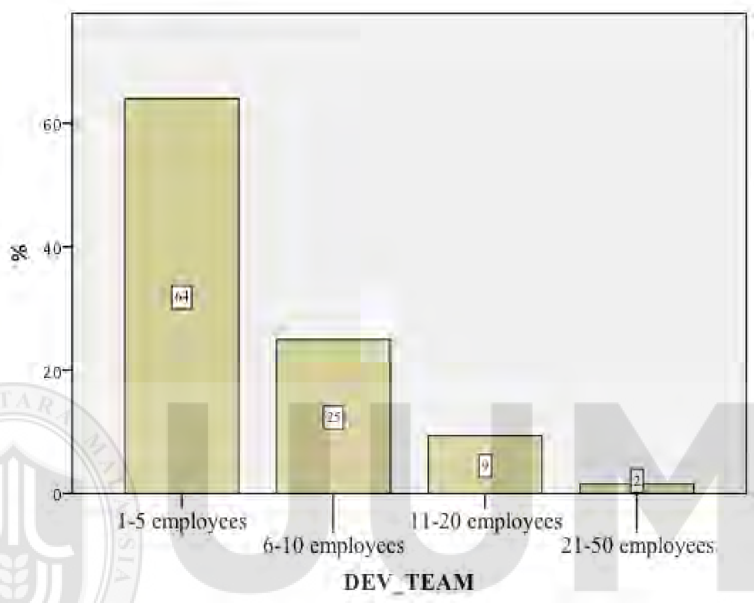

Figure 4.6. Development Team Size

4.4.2.4 Types of Developed Product

The next question was about the type of developed products. Four types of products were given as options: websites, mobile applications, desktop applications, and others. Since it was a multiple choice question, the frequency of the cases in which an organization developed more than one product type was measured. From the results shown in Figure 4.7, more than half of the organization developed only one type of product while 34% of the organizations of respondent developed two different types of products. Only 14% of the organizations developed three types of products.

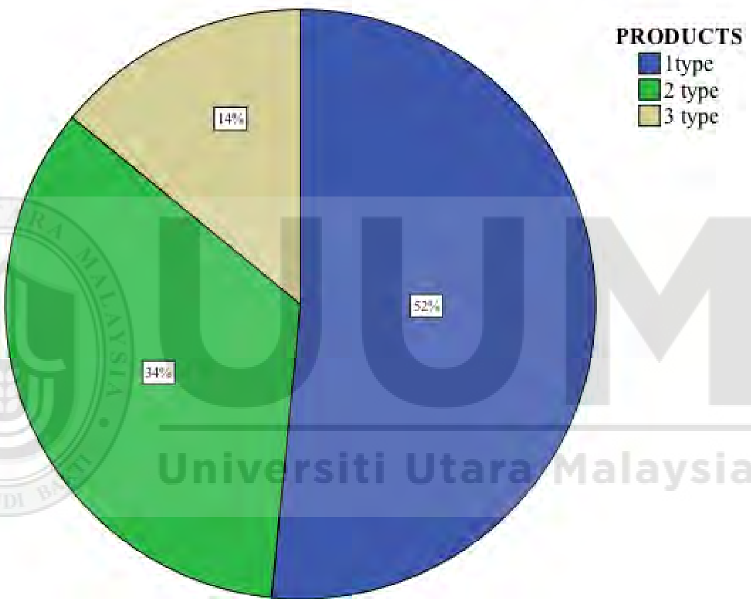

Figure 4.7. Number of Products Type Developed by Each Organization

As shown in Table 4.7, half (50%) of the organizations developed websites application, 26% of the organizations developed mobile applications and 20% of the organizations developed desktop applications. Hence, only 6% of the organizations developed other types of products(like: embedded application).

Table 4.7
Type of Product Descriptive

Product type [a]	Responses		Percent of Cases
	N	Percentage	
Websites application	52	50%	81.3%
Mobile applications	27	26%	42.2%
Desktop applications	21	20.2%	32.8%
Other	4	3.8%	6.3%
Total	104	100%	162.6%

a. Dichotomy group tabulated at value 1.

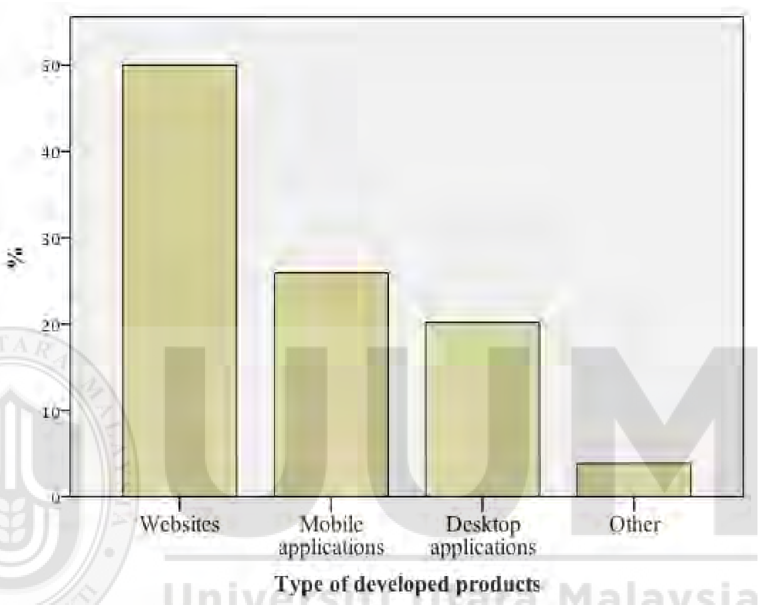

Figure 4.8. Types of Developed Product

4.4.2.5 Organization Experience in Agile Software Development

The next demographic question was asked as to understand for how long the organizations have been using agile software development methodology. Based on results shown in Table 4.8, 34% of the organizations reported having used ASDM for less than one year while 33% of the organizations reported having used ASDM for 1 to 2 years. 28% of the organizations reported having used ASDM 3 to 5 years. Only 5% of the organizations reported having used ASDM for more than five years.

Table 4.8
Organization Agile Experience

ASDM Experience	Frequency	Percentage (%)	Mean	Std. Deviation
<1 year	22	34.4	3.03	.908
1-2 years	21	32.8		
3-5 years	18	28.1		
5+ years	3	4.7		
Total	64	100		

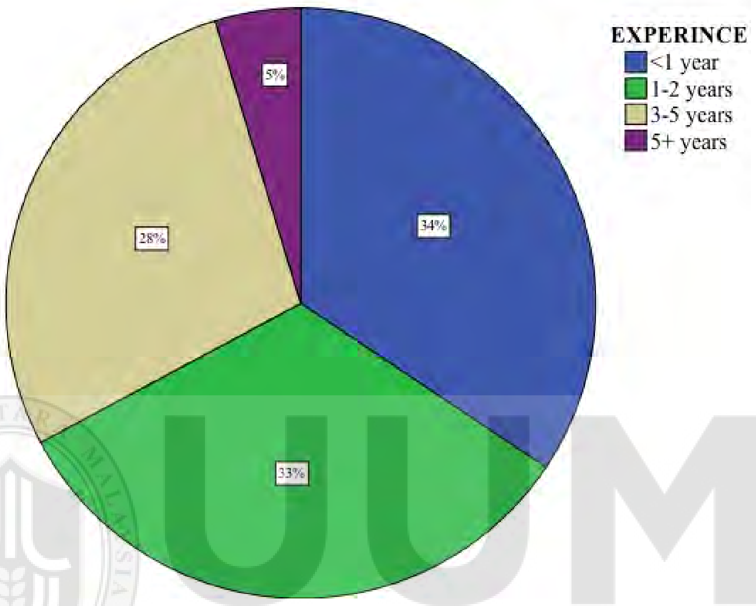

Figure 4.9. Organizations' Agile Experience

4.4.2.6 Software Development Methodologies

The final demographic question was help to understand which agile development methodology has been used by the participant software startups. This question was a multiple choice question aimed at helping understand whether the respondents use hybrid methodology. The mostly adopted agile software development methodology was scrum, where 38% of respondents used it. 15% of organization adopted XP methodology while 19% follow lean methodology. Others included FDD (7%), DSDM (13%) and other methodologies (8%); as shown in Table 4.9. Notably, none of the respondents was adopted Crystal methods methodology. On the other hand, 18% of the respondents answered that they do not follow any specific methodology.

Table 4.9
Adopted Agile methodology

Adopted Methodology [a]	Responses		Percentage of Cases
	N	Percentage	
XP	9	11.4%	14.5%
Scrum	30	38.0%	48.4%
FDD	4	5.1%	6.5%
Lean	12	15.2%	19.4%
Crystal methods	0	0.0%	0.0%
DSDM	8	10.1%	12.9%
No Methodology	11	13.9%	17.7%
Other Methodologies	5	6.3%	8.1%
Total	93	100.0%	127.5%

a. Dichotomy group tabulated at value 1.

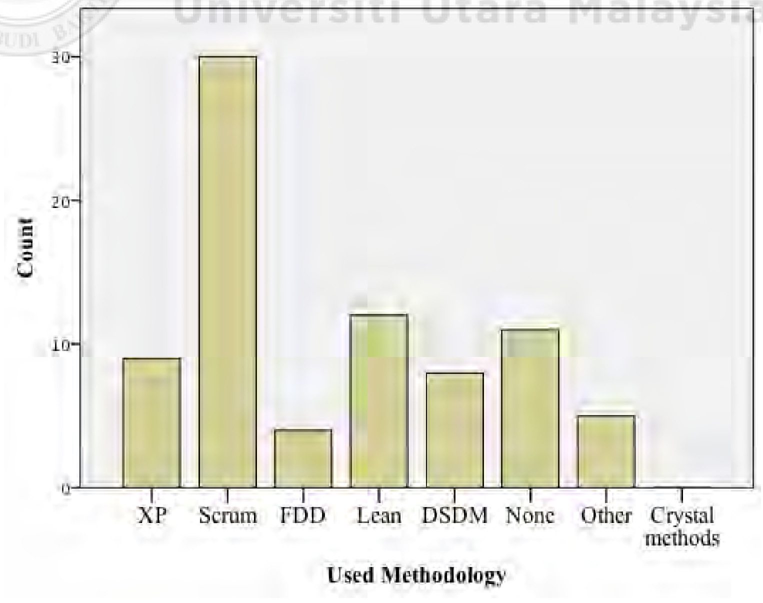

Figure 4.10. Adopted ASDM by Organization

4.5 Descriptive Statistic

4.5.1 Agile Practices

Descriptive statistic was used to understand the agile practices state in software startups. It is clear from Table 4.10 that the highest scoring practices were prioritized backlogs (a mean of 3.84 on a five-point scale, *SD* = 1.027), open work area (*M* = 3.80, *SD* = 1.184), continuous integration (*M* = 3.75, *SD* = 1.168), coding standards (*M* = 3.75, SD = 1.127), and story mapping (*M* = 3.64, *SD* = 0.982). As well as the Table 4.11 shows the frequencies of agile practices usages answers.

Table 4.10
Most Used Agile Practices

	Practices	**N**	**Mean**	**Std. Deviation**
1	Prioritized backlogs	64	3.84	1.027
2	Open Work area	64	3.80	1.184
3	Continuous integration	64	3.75	1.168
4	Coding standards	64	3.75	1.127
5	Story mapping	64	3.64	0.982
6	Release planning	64	3.61	1.203
7	Refactoring	64	3.58	1.081
8	Taskboard	64	3.55	1.425
9	Single team	64	3.55	1.221
10	Team-based estimation	64	3.48	1.054

Table 4.11
Frequency Distribution of Agile Practice

Practice	Never	Rarely	Sometimes	Very Often	Always
Daily Meeting	11 (17.20%)	5 (7.80%)	22 (34.40%)	11 (17.20%)	15 (23.40%)
Short iterations	6 (9.40%)	3 (4.70%)	27 (42.20%)	16 (25.00%)	12 (18.80%)
Unit testing	9 (14.10%)	11 (17.20%)	22 (34.40%)	14 (21.90%)	8 (12.50%)
Retrospectives	9 (14.10%)	5 (7.80%)	30 (46.90%)	12 (18.80%)	8 (12.50%)
Prioritized backlogs	3 (4.70%)	2 (3.10%)	15 (23.40%)	26 (40.60%)	18 (28.10%)
Team-based estimation	2 (3.10%)	8 (12.50%)	24 (37.50%)	17 (26.60%)	13 (20.30%)
Coding standards	4 (6.30%)	3 (4.70%)	17 (26.60%)	21 (32.80%)	19 (29.70%)
Continuous integration	4 (6.30%)	7 (10.90%)	8 (12.50%)	27 (42.20%)	18 (28.10%)
Iteration reviews	6 (9.40%)	8 (12.50%)	20 (31.30%)	21 (32.80%)	9 (14.10%)
Dedicated product owner	8 (12.50%)	5 (7.80%)	15 (23.40%)	26 (40.60%)	10 (15.60%)
Single team	7 (10.90%)	2 (3.10%)	20 (31.30%)	19 (29.70%)	16 (25.00%)
Refactoring	6 (9.40%)	1 (1.60%)	17 (26.60%)	30 (46.90%)	10 (15.60%)
Open Work area	7 (10.90%)	0 (0.00%)	10 (15.60%)	29 (45.30%)	18 (28.10%)
TDD	8 (12.50%)	8 (12.50%)	24 (37.50%)	18 (28.10%)	6 (9.40%)
Story mapping	3 (4.70%)	3 (4.70%)	19 (29.70%)	28 (43.80%)	11 (17.20%)
Collective code ownership	3 (4.70%)	10 (15.60%)	19 (29.70%)	18 (28.10%)	14 (21.90%)
Pair programming	20 (31.30%)	10 (15.60%)	18 (28.10%)	13 (20.30%)	3 (4.70%)
Automated acceptance testing	13 (20.30%)	10 (15.60%)	21 (32.80%)	14 (21.90%)	6 (9.40%)
Iteration planning	8 (12.50%)	9 (14.10%)	20 (31.30%)	15 (23.40%)	12 (18.80%)
Continues deployment	9 (14.10%)	2 (3.10%)	20 (31.30%)	19 (29.70%)	14 (21.90%)
Release planning	6 (9.40%)	6 (9.40%)	9 (14.10%)	29 (45.30%)	14 (21.90%)
Taskboard	10 (15.60%)	6 (9.40%)	7 (10.90%)	21 (32.80%)	20 (31.30%)
BDD	14 (21.90%)	18 (28.10%)	16 (25.00%)	11 (17.20%)	5 (7.80%)
User Story	10 (15.60%)	5 (7.80%)	12 (18.80%)	20 (31.30%)	17 (26.60%)

4.5.2 Agile Adoption Motivation

As shown in Table 4.12, the main motivation for software startups to adopt ASDM was to enhance their abilities to manage the changing in priorities the projects (a mean of 3.03 on a four-point scale, *SD* = .835). The second motivation to adopt ASDM by participants was to accelerate their product deliverability (*M* = 3.00, *SD* = .992) by adopting the ASDM. The rest motivations are ordered as following: to increase software maintainability (*M* = 2.91, *SD* = .868), enhance delivery predictability (*M* = 2.88, *SD* = .882), simplify development process (*M* = 2.86, *SD* = .889), increase team productivity (*M* = 2.83, *SD* = .969), reduce project risk (*M* = 2.80, *SD* = .979), enhance software quality (*M* = 2.77, *SD* = .955), improve business and IT alignment (*M* = 2.73, *SD* = .930), better manage distributed teams (*M* = 2.69, *SD* = .957), improve team morale (*M* = 2.64, *SD* = .861), reduce project cost (*M* = 2.48, *SD* = .891), improve project visibility (*M* = 2.47, *SD* = .835), and improve engineering discipline (*M* = 2.41, *SD* = .849).

Table 4.12
Motivation to Adopt ASDM

	Agile methodology adopted to...	**Mean**	**Std. Deviation**
1	Enhance ability to manage changing priorities	3.03	.835
2	Accelerate product delivery	3.00	.992
3	Increase software maintainability	2.91	.868
4	Enhance delivery predictability	2.88	.882
5	Simplify development process	2.86	.889
6	Increase team productivity	2.83	.969
7	Reduce project risk	2.80	.979
8	Enhance software quality	2.77	.955
9	Improve business and IT alignment	2.73	.930
10	Better manage distributed teams	2.69	.957
11	Improve team morale	2.64	.861
12	Reduce project cost	2.48	.891
13	Improve project visibility	2.47	.835
14	Improve engineering discipline	2.41	.849

4.6 Hierarchical Cluster Analysis

In order to perform cluster analysis, the agile practice variables were analyzed in SPSS using the described methods. The analysis produced a total of 19 clusters (count of variables-1) (see Table 4.13). Also, a dendrogram was generated as shown in Figure 4.11. Dendrogram was used to determine the clusters numbers; as shown in Figure 4.11, the dendrogram at level 10 show four clusters of agile practices. From the dendrogram, four new clusters were identified. They are shown in Table 4.14.

Table 4.13
Agglomeration Schedule

Stage	**Cluster Combined**		**Coefficients**	**Stage Cluster First Appears**		**Next Stage**
	Cluster 1	**Cluster 2**		**Cluster 1**	**Cluster 2**	
1	7	8	23.00	0	0	15
2	5	11	47.50	0	0	12
3	18	19	76.50	0	0	13
4	12	13	105.50	0	0	12
5	2	4	134.50	0	0	821
6	6	9	164.50	0	0	76
7	6	10	205.17	6	0	14
8	1	2	248.17	0	5	11
9	15	16	291.67	0	0	13
10	3	14	335.17	0	0	15
11	1	17	383.17	8	0	17
12	5	12	435.42	2	4	16
13	15	18	494.67	9	3	17
14	6	20	560.00	7	0	16
15	3	7	627.75	10	1	19
16	5	6	706.38	12	14	18
17	1	15	821.50	11	13	18
18	1	5	949.88	17	16	19
19	1	3	1134.05	18	15	05

As shown in Table 4.14, the first cluster contains 4 agile practices which are: daily meeting, short iterations, retrospectives, and iteration planning. The second cluster has also 4 agile practices which are: unit testing, coding standards, continues integration, and test-driven development. The third cluster has four agile practices which are: story mapping, collective code ownership, continues deployment, and release plaining. The final cluster has 8 agile practices which are: prioritized backlogs, team-based

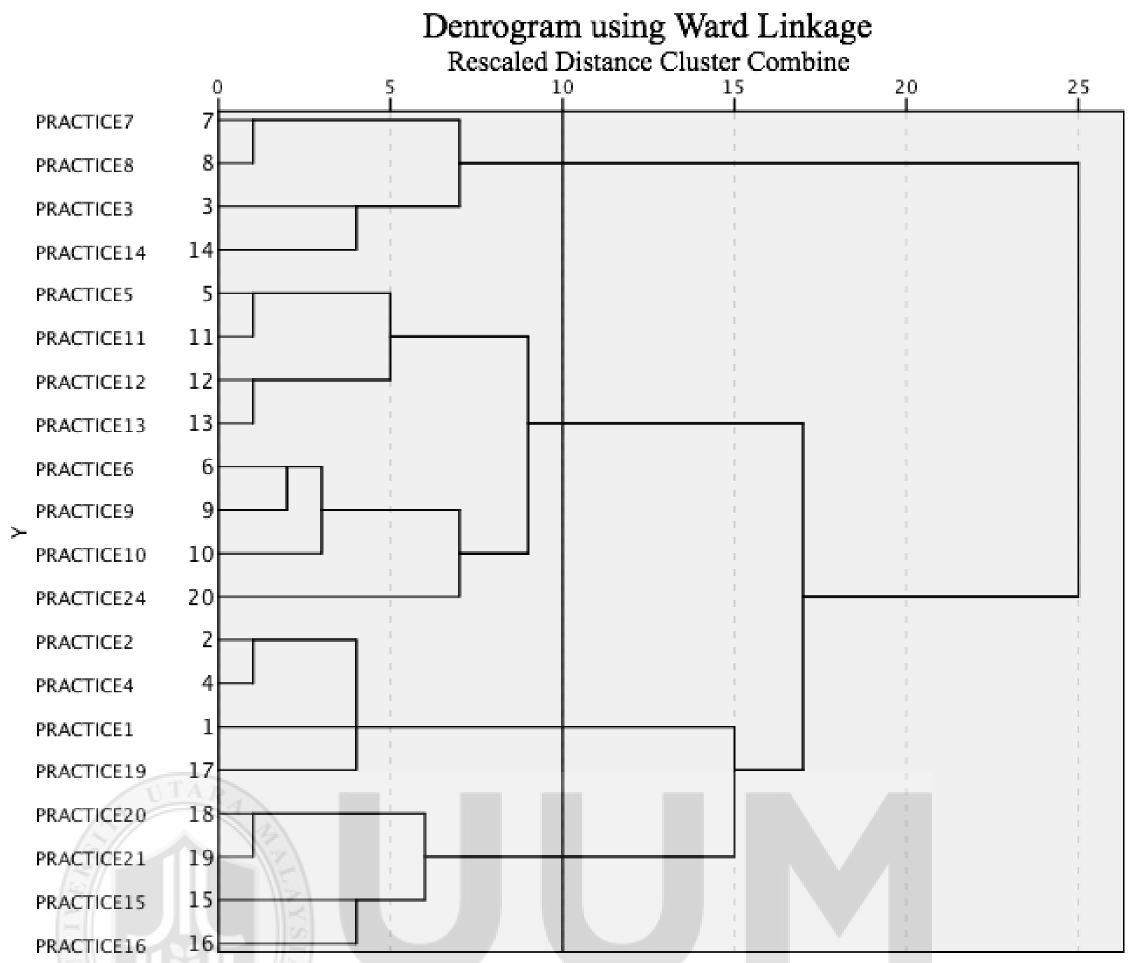

Figure 4.11. Dendrogram from HCA

estimation, iteration reviews, dedicated product ownership, single team, refactoring, open work area, and user story. Therefore, three over four clusters have the same size, which are four agile practices, thus is a feature of Ward's algorithm to create a similar size cluster as possible.

The new clusters were analyzed using descriptive statistics to figure out the most agile practices clusters' used by software startups. As shown in Table 4.15, cluster 3 (*M* = 3.55, *SD* = 0.873) and cluster 4 (*M* = 3.54, *SD* = 0.904) are the most commonly used clusters by software startup. Although cluster 2 (*M* = 3.40, *SD* = 0.907) and cluster 1 (*M* = 3.29, *SD* = 0.917) are the least adopted clusters in software startups.

Table 4.14
Cluster's Membership

No.	Cluster 1	Cluster 2	Cluster 3	Cluster 4
1	PRACTICE1 Daily meeting	PRACTICE3 Unit testing	PRACTICE15 Story mapping	PRACTICE5 Prioritized back-logs
2	PRACTICE2 Short iterations	PRACTICE7 Coding standards	PRACTICE16 Collective code ownership	PRACTICE6 Team-based esti-mation
3	PRACTICE4 Retrospectives	PRACTICE8 Continuous inte-gration	PRACTICE20 Continues de-ployment	PRACTICE9 Iteration reviews
4	PRACTICE19 Iteration planning	PRACTICE14 Test-driven development	PRACTICE21 Release planning	PRACTICE10 Dedicated prod-uct ownership
5				PRACTICE11 Single team
6				PRACTICE12 Refactoring
7				PRACTICE13 Open work area
8				PRACTICE24 User story

Table 4.15
Most Used Agile Practices' Cluster

No.	Cluster	N	Mean	Std. Deviation
1	CLUSTER3	64	3.55	0.873
2	CLUSTER4	64	3.54	0.904
3	CLUSTER2	64	3.40	0.907
4	CLUSTER1	64	3.29	0.917

4.7 Reliability Analysis

A reliability analysis was conducted in order to measure the internal consistency across items using Cronbach's Coefficient Alpha. Cronbach's alpha indicates how well the items measure a concept as a set. According to Sekaran and Bougie (2010), the value of 0.60 is generally considered sufficient to be acceptable for research. Furthermore, a reliability more than 0.80 is considered good (Hair Jr, Hult, Ringle, & Sarstedt, 2013).

Based on the factor analysis results, the Cronbach's alpha for dimensions of adoption motivation factors are above 0.6, which meets the minimum accepted reliability (see Table 4.16).

Table 4.16
Reliability Coefficients of Agile Adoption Motivation Factors Variables

Variables Name	No. of Items	Cronbach's Alpha
Quality	3	0.769
Efficiency	3	0.736
Effectiveness	2	0.525

In the same, agile practices' clusters alpha value is above 0.8, which is considered as good (see Table 4.17). Overall, the reliability analysis undertaken on the items showed that all measurements were reliable and internally consistent.

Table 4.17
Reliability Coefficients of Agile Practices Clusters Variables

Variables Name	No. of Items	Cronbach's Alpha
Cluster1	4	0.760
Cluster2	4	0.855
Cluster3	4	0.790
Cluster4	8	0.887

4.8 Research Hypotheses

Based on the results of HCA analysis, the clusters of the generated agile practices were used to state sub-hypotheses of the three main research hypotheses stated earlier. The results of hypotheses testing are as follows:

H1 Motivation to improve software quality has a positive relationship with adopted agile practices clusters by organizations.

H1a Motivation to improve software quality has a positive relationship with adopted agile practices cluster one by organizations.

H1b Motivation to improve software quality has a positive relationship with adopted agile practices cluster two by organizations.

H1c Motivation to improve software quality has a positive relationship with adopted agile practices cluster three by organizations.

H1d two by organizations.

H1c Motivation to improve software quality has a positive relationship with adopted agile practices cluster four by organizations.

H2 Motivation to improve development efficiency has a positive relationship with adopted agile practices clusters by organizations.

H2a Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster one by organizations.

H2b Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster two by organizations.

H2c Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster three by organizations.

H2d Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster four by organizations.

H3 Motivation to improve development effectiveness has a positive relationship with adopted agile practices clusters by organizations.

H3a Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster one by organizations.

H3b Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster two by organizations.

H3c Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster three by organizations.

H3d Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster four by organizations.

4.9 Hypotheses Testing

Correlation analysis is used to describe the direction, strength, and significance of the relationship among variables measured at an interval or ratio level (Sekaran & Bougie, 2010). A hypothesis that postulates a significant positive/negative relationship between two variables can be tested by examining the correlation between the two (Sekaran & Bougie, 2010). Table 4.18 illustrates the correlation between study variables.

Table 4.18
Correlation of Variables

		Quality	**Efficiency**	**Effectiveness**
CLUS1	Pearson Correlation	.016	.123	.031
	Sig. (2-tailed)	.900	.331	.809
CLUS2	Pearson Correlation	.244	.143	.273*
	Sig. (2-tailed)	.052	.259	.029
CLUS3	Pearson Correlation	.235**	.362**	.311*
	Sig. (2-tailed)	.004	.003	.012
CLUS4	Pearson Correlation	.098	.163	.262*
	Sig. (2-tailed)	.443	.199	.036

**. Correlation is significant at the .01 level (2-tailed).
*. Correlation is significant at the 0.05 level (2-tailed).

All the hypotheses were tested. The correlation matrix provided the answers to all the hypotheses. Motivation to improve software quality was found to have a positive correlation with single cluster over 4 clusters. In other words, improvement in software quality had no correlation with cluster one (r(64) = 0.120, p = 0.300), cluster two (r(64) = 0.244, p = 0.052), In contrast, cluster three had a positive correlation with quality motivation factor, r(64) = 0.235, $p < 0.05$.

Similarly, motivation to improve development efficiency had a positive correlation with one clusters. It was a strong positive correlated with cluster three (r(64) = 0.362, $p < 0.05$). However, cluster one (r(64) = 0.123, p = 0.331), cluster two (r(64) = 0.143, p = 0.259), and cluster four (r(64) = 0.163 p = 0.199) do not have a relationship with this factor.

Despite this, motivation to improve development effectiveness had a positive correlation with 3 of 4 clusters. The 3 correlation is very significance, but the cluster one is not significance correlated. In other hand, improved development effectiveness was strongly positively correlated to cluster two (r(64) = 0.273, $p < 0.05$), cluster three (r(64) = 0.311, $p < 0.05$) and cluster four (r(64) = 0.262, $p < 0.05$).

4.10 Summary of Hypotheses Testing

From the previous analysis, a summary of the hypotheses testing results are presented in Table 4.19. It has been shown that 5 over 12 hypotheses were supported. Next chapter discussed the result of hypothesis in details.

Table 4.19
Summary of Hypothesis Testing Results

No.	Hypothesis	Result
H1a	Motivation to improve software quality has a positive relationship with adopted agile practices cluster one by organizations.	Not supported
H1b	Motivation to improve software quality has a positive relationship with adopted agile practices cluster two by organizations.	Not supported
H1c	Motivation to improve software quality has a positive relationship with adopted agile practices cluster three by organizations.	Supported
H1d	Motivation to improve software quality has a positive relationship with adopted agile practices cluster four by organizations.	Not supported
H2a	Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster one by organizations.	Not supported
H2b	Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster two by organizations.	Not supported
H2c	Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster three by organizations.	Supported
H2d	Motivation to improve development efficiency has a positive relationship with adopted agile practices cluster four by organizations.	Not supported
H3a	Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster one by organizations.	Not supported
H3b	Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster two by organizations.	Supported
H3c	Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster three by organizations.	Supported
H3d	Motivation to improve development effectiveness has a positive relationship with adopted agile practices cluster four by organizations.	Supported

4.11 Conclusion

This chapter presented and discussed the statistical analyses and results. The analyses carried out include frequency analysis, descriptive analysis, cluster analysis, and correlation analysis. Hierarchical cluster analysis generates four clusters of agile practice. The results reveal that five of the study's hypotheses are valid. These findings will be used for discussions, implications and contributions of the research as will be presented in the next chapter.

CHAPTER FIVE
DISCUSSION AND CONCLUSION

5.1 Introduction

This chapter discusses the findings reported in chapter four. Subsequently, the findings elaborate the implications for software startups and software engineering in a small organization. In addition, it provides implication guides to software startup companies in KSA in terms of adopting proper agile practices. It also includes limitations for this study, and finally suggestions for future research.

5.2 Recapitulation of the Study Findings

The purpose of this study is to investigate the relationship between agile adoption motivation factors and the agile practices' clusters. As the hierarchical cluster analysis (see Table 4.14) found four agile practices' clusters. Those four clusters were correlated with the agile motivation factors which improve software quality, improve efficiency, and improve effectiveness. In addition to that, this study investigates motivations that influence software startups to adopt ASMD. In order to test the hypotheses, a correlation analysis was performed. Twelve hypotheses were tested. The results are discussed in details below.

5.3 Discussing of Findings

5.3.1 Most Used Agile Practice in Software Startups

The first objective of the current study was to determine the most used agile practices, as well as the most used agile practices' cluster. Based on the practices adopted by the respondents, the most adopted practices by software startups are shown on Table 4.15 (see Appendix C for complete list). The highest scoring practices were prioritized backlogs, open work area, continuous integration, coding standards, and story mapping. These practices support different goals, like: project management, continuous improvement, and team's management. For example, open work area adopted to improve the communication between stakeholder. While prioritized backlogs practice help to manage the tasks.

In contrast, a recent study by VersionOne (2015) found the five most used agile practices are daily standup, prioritized backlogs, short iterations, retrospective, and iteration planning. Only one practice that is shared with the VersionOne result which is prioritized backlogs practice. The difference occurs for two main reasons: the different population in each survey; since our survey was conducted in Saudi Arabia whereas the VersionOne survey was a global study. The second reason is that this study focuses only on software startups companies yet the VersionOne study focuses on all domain of software companies. To put it differently, the difference occurs based on several factors including geographic and type of organization. The data appears to suggest that it is important to study the relationship between the adopted practices with different variables, like location, the number of developer, type of project, and others, to get the full insight about used agile practices.

Furthermore, agile practices are studied as clusters, and the HCA yielded that there are four new practices' clusters, as shown on Figure 4.11. Also, Table 4.15 shows the most used agile practice cluster in software startups. Cluster 3 and Cluster 4 where the most adopted agile practices clusters in software startups. Whereas most of the participants adopt those clusters fully or partly. The clusters are discussed in detail below.

Based on the literature review, the new clusters could be labeled based on several criteria as following. Cluster 1 (see Table 4.14) is labeled as a project management cluster, since three to four practices are similar to those on the project management cluster identified by Tripp and Armstrong (2014), as most of their practices support project management activity. As shown in Table 4.15, the project management cluster is less adopted by software startups, because startups implement a loose organizational structure and avoid traditional management (Giardino et al., 2014). This is caused by the fact that startups work under pressure with limited time and small size team.

In the second cluster, it is clear its practices support testing and quality responsibilities. This cluster was labeled as a quality assurance cluster. This cluster shares 75% of quality assurance cluster defined by Abbas et al. (2010). According to Stolberg (2009) in continuous integration, “each integration is verified by an automated build (including test) to detect integration errors as quickly as possible”, which makes it clear that it continues integration codependent with other testing practices in the same cluster. As mentioned earlier when agile practices are adopted together that give it more benefits. It is clear that continuous integration is adopted more by software startups (see Table 4.10) because it helps them to release products quickly. In contrast, continuous integration is only adopted by 50% of the participants in the VersionOne (2015) survey, where the organization category is different. It can be seen that the difference in motivation results in differences in adopted agile practices.

The third cluster is labeled as a software process clusters. In contrast to other clusters, this cluster’s practices support different goals for that reasons it labeled using a general name. Despite this, this cluster was the most adopted agile practices cluster by software startups in KSA. This cluster’s practices, contribute to the main adoption motivations, which enhance the ability to manage change, and accelerate product delivery. The link between story mapping and release planning, where both of them deal with a product backlog, and both of these tasks support better management for changing the product.

In fact, software startups suffering from rapid changing in requirements that explains why this cluster is the most adopted in software startups. By adopting these practices organization can enhance their abilities to manage change. Equally important, collective code ownership enable team members to work directly on any part of the code in response to new changes in requirement, that also support the previous adoption motivations. Finally, continuous deployment accelerates the deliverability of product to the end-user. It has been shown that is why this cluster is the most adopted agile practices' cluster among others because it greatly supports the main adoption motivations for software startups.

The final cluster is labeled as the incremental and iterative cluster. It was difficult to label this cluster because it has eight practices while other clusters have only four practices. Most of these practices support the iterative process like iteration review, prioritized backlog, team-based estimation and user story. As startups focus on fast move from idea conception to production by using iterative and incremental approach; these approaches help them to reach this goal (Giardino et al., 2014), which explain why this cluster is the second most used agile practice clusters software startups in KSA (see Table 4.15). Iterative development is an important for software startups on account of software startups are product-driven company and focusing on product rapid release. This is in contrast to traditional software development methodology where the product release at the end. In the same way, this will cause to reduce the cost of change, as software startups suffer from rapidly changing.

5.3.2 Software Startups Motivation to Adopt ASDM

The second objective of this study is to explain why software startups adopt agile as development methodologies for their projects. Understanding the motivations to adopt ASDM is important so it helps to explain all steps during the adoption process. Based on data analysis shown in Table 4.12, the software startups aim to enhance their abilities to manage the changing in priorities the projects by adopting the ASDM. The second reason to adopt ASDM by participants was to accelerate their product deliverability. Without a doubt these results make sense, since the software startups are product-driven companies that suffer from a rapid changing of requirements; where many software startups adopt ASDM to overcome this challenge (Taipale, 2010). The next goals are to increase software maintainability, simplify development process, and enhance delivery predictability, since startups have a small team with lack of other recourses, it is important to manage these resources wisely and avoid wasting it on complex management activities that exist on a rigid software development methodology. The comprehensive list of motivation is presented by order in Table 4.12. These results reflect on the software startups attribute discussed in Chapter 2.

Previous studies found the most important motivation for startups to adopt agile methodology is to produce a ready product that can be brought to market as soon as possible (Coleman & O'Connor, 2008; Giardino et al., 2014). A recent study by VersionOne (2015) found the most important motivation for an organization to adopt ASDM was the accelerated product delivery, the enhanced ability to manage changing priorities, increase productivity, and enhance software quality. In contrast, software startups looking for an increase in their team's productivity that help utilize their limited resource to build their product with less hassle. Agile methodologies assist and simplify the processes of developing new products. Moreover, ASDM help software startups utilize their resources.

5.3.3 Relationship Between Agile Adoption Motivation Factors and Adopted Agile Practice Cluster

In order to evaluate the relationship between agile adoption motivation factors (i.e.: quality, efficiency, and effectiveness) and agile practice clusters, a Pearson's correlation was conducted. A correlation analysis was used to help understand the degree of relationship among motivation variables and all practices clusters. Twelve hypothesis were tested; the test was conducted at level 0.05. The results indicates five of the hypothesis were supported, as shown on Table 4.19.

First, the hypotheses stated that motivation to improve software quality has a positive relationship with adopted agile practices clusters by organizations. The results supported only one hypothesis (H1c). Whereas, hypotheses H1a, H1b, and H1d were not supported by the results. These results are consistent with the result found by Tripp and Armstrong (2014) where they did not find a relationship between enhanced software quality and the iteration planning practice, which is a member of cluster one (project management cluster). The positive relationship between software process clusters and motivation to improve software quality elucidate why software startups adopt this cluster. The adopter considers this cluster's practices has a positive effect on the software quality. As explained by Shore and Warden (2007) ”with collective code ownership, everyone shares responsibility for code quality”. In addition, with collective code ownership, all the programmers are in a way or another responsible for maintaining the code.

Second, the hypotheses stated that motivation to improve development efficiency has a positive relationship with adopted agile practices clusters by organizations. Only one of four sub-hypotheses was supported by the results which is H2c. The correlations was strong and positive. However, the result did not find any significance relationship in

H2a, H2b, and H2c. Further evidence showed there is no relationship between cluster 4 (incremental and iterative cluster), and the efficiency that is aligned with the findings of Tripp and Armstrong (2014), who did not find a relationship with refactoring practice, where refactoring practice is a member with cluster 4. Considering a software process clusters has a relationship with motivation to improve development efficiency it can be concluded that agile adopter believes that by adopting the group of practice that support manages requirement change, and accelerate delivery of product will improve their development efficiency.

Third, hypothesis stated that motivation to improve development effectiveness has a positive relationship with adopted agile practices clusters by organizations. The results were positive and supporting the following hypotheses: H3b, H3c, and H3d. Although, hypothesis H3a was not supported by the result. Therefore, cluster 3 had a positive relationship with all studied motivation factors. This is in contrast to cluster 2, and cluster 4 which are only associated with effectiveness motivation factors.

The evidence found that cluster one (project management cluster) is not significant with any of the three adoption motivation factors included in this study. That is to say the studied motivation factor does not explain the motivation to adopt this cluster. To explain the nature of those practices included in the cluster need to study it with other motivation factors. Also due to startups and small-team organizations do not focus on a project management task in response to their resources limitation, as Table 4.15 shows the project management cluster is the least adopted over other clusters. As startups have a very small and co-located development team, that enables members to operate with high coordination, control and communication (Giardino & Paternoster, 2012), without the need for traditional management roles on a development team. In other words, startups development teams are self-organized. Self-organizing teams help organizations to increase the interactions and communication between the team members (S. Misra, Kumar, Kumar, Fantazy, & Akhter, 2012).

In total, the correlation result in this study is almost in the same direction but stronger than the one found in Tripp and Armstrong (2014). That implies that the study of a group of practices explains the relationship more than the studies of the practice as individual because of the codependent on agile practices. The other reason is the measurement of using agile practice; while Tripp and Armstrong (2014) used yes/no, our study uses the 4-point scale. This helps to understand exactly the level of adoption for each agile practice.

5.4 Implication of Study

The result of this study had provided several theoretical implications for future research and also some practical implications for software startups.

5.4.1 Theoretical Implication

In general, this study aims to explore the relationship between the agile adoption motivation factor and selected agile practices' clusters. This relationship will help researcher to conduct more studies to understand tailoring of agile software development methodology in early stage adoption. Also, it will help to understand how agile practices are tailored in software startups. These findings provide evidence for the relationship between organizational motivations for the adoption of agile development methodology and the agile practices clusters used. More research is needed to understand the early agile adoption and figure out the relationships between different variables such as customer satisfaction, and adoption success.

This study brings about a method of tailored ASDM through the alignment of agile adoption motivation factors. As the method tailoring area is still new, especially in the context of agile methodology (Conboy & Fitzgerald, 2010). The tailoring of ASDM is a key part of the successful methodology implementation process. The perception of why organizations adopt the agile methodology, and, which types of agile practices clusters fit with that motives will lead to a successful tailoring of the process. By understanding the organization's overall goal in terms of adoption of agile methodology (e.g., increase efficiency) and which types of agile practices cluster are better to align with that goal (e.g., quality assurance cluster) may increase the success of the method tailoring process.

Also, this study contributes to the literature of software startups as it is a unique case of small and medium-sized enterprises (SME). Furthermore, startups are a growing phenomenon globally, where the foregoing discussion implies that software startups still need to be studied and explored in the context of software engineering (Giardino et al., 2014). Therefore, it is important to help startups successfully adopt ASDM. Moreover, by understanding adopted agile practices/clusters, it will help to understand how startups develop their product, and will reduce the failure of development, as match the agile motivation factors with proper agile practices. In total that will help researchers in the future to study software startups. This study found that software startups adopt ASDM to accelerate their product deliverability, to manage the changing in project, and increase software maintainability. This knowledge will help to build a customize solution to support software startups success with agile software development.

Finally, existing literature recommends identifying practices clusters which are interdependent between practices (Conboy & Fitzgerald, 2010). This research found statistically four practices clusters. It was constructed using a hierarchical cluster analysis from a total of twenty-four agile practices and four clusters were generated. The new clusters that adopted by software startups are a project management cluster, quality assurance cluster, team communication cluster, and incremental and iterative cluster.

5.4.2 Practical Implication

A key implication for project managers, CTO, or CEO is the need to match their needs to the contextual agile practices of the project. In other words, they are advised to match their need with agile practices clusters to be adopted. That will help startups have a fast formation of their agile practices and maximize their potential by using the practices together. This will help startups to tune their method tailoring process.

In practice, understanding agile practices clusters will help companies to maximize their benefits from adopting these clusters as some agile practices work better when they are adopted together (Conboy & Fitzgerald, 2010). Startups need to choose the appreciate process quickly (Giardino et al., 2014), and startups do not have available resources to figure out the best way to develop the product. On the contrary adopting the practices proposed by an ASDM will lead the organization to spend more effort and resources to adopting it; without evaluation, each proposed agile practice could bring more value to an organization or not (Abbas et al., 2010; Kurapati et al., 2012). This study can be used as a guideline to decision making in software startups to adopt agile practices (or clusters). Consequently, that will help organizations to overcome issues found on the adoption process (Campanelli & Parreiras, 2015).

5.5 Limitations

As with all studies, there are a few limitations for this study. The study investigated only three motivation variables which are enhance quality, improve efficiency, and improve effectiveness, because the time limit of this research and availability of re-sources to investigate further. Yet there are other adoption motivations' variables that need further investigation in the future. In addition to that, the current study is limited to software startups organization. However, the future studies should cover more organization types, for example, small and medium enterprises, large corporation, and government sector, in order to compare the result from different sectors.

5.6 Future Research

Although the findings of this study find new agile practices clusters. Further investigation using quantitative research is suggested to determine the correlations between the adopting of agile practices clusters and the effectiveness of the individual practices within that cluster. Also, it is important to figure out the effect of adoption that clusters have on the project success. The research recommends to conduct mixed methods (combining qualitative and quantitative) study in order to understand the agile practices cluster, and moreover, to identify clusters statistically (quantitative) then interview (qualitative) with experts to refine the new clusters.

Future studies are needed to discover and understand more about those clusters, in order to understand how each cluster member support one another, which will help to understand how to maximize the benefit from fully adopting a cluster. Also, there is a need to discover the relationship between these clusters and other variables, e.g.: quality, project success, customer satisfaction and others. Furthermore, there is a need to identify the impact of adopting these clusters after successfully adopting agile methodology. Since there is still a lack of empirical studies, that focus should be on agile practices (Rauf & AlGhafees, 2015).

As a future study, the researcher aims to extend the scope of this study to include other sectors like government and large enterprises. It is important to conduct these studies and compare the generated clusters on a different geographic location as well as in different company sectors. That will help to understand more about agile methods tailoring and generalize the result. Further research might be needed to connect the finding of this study with the success of adoption as well as the success of the project.

5.7 Conclusion

As a final note, this study takes an important step toward understanding the relationship between agile practices clusters and the motivation to adopt agile methodology. The results found a relationships between agile adoption motivation factors (quality, efficiency, and effectiveness) with the adopted agile practices. This knowledge will help ASDM adopter to select the proper agile practice cluster based on their needs. That will affect the success of the agile adoption process.

The results from the survey showed the current agile practices clusters and motivations used by software startups in KSA. The overall results of the research confirm that startups possess unique characteristics of uncertainty, lack of resources and time-pressure. These factors influence directly how the software startups adopting agile software development methodology. As the result indicate that software startup motivations to adopt ASDM is different to other software engineering domain. Furthermore, the adopted agile practices by software startups asserted the uniqueness of startups in software engineering context.

REFERENCES

Abbas, N., Gravell, A. M., & Wills, G. B. (2010). Using Factor Analysis to Generate Clusters of Agile Practices (A Guide for Agile Process Improvement). In *2010 agile conference* (pp. 11–20). IEEE. doi: 10.1109/AGILE.2010.15

Abrahamsson, P., Oza, N., & Siponen, M. T. (2010). Agile Software Development Methods: A Comparative Review. In *Agile software development* (pp. 31–59). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-12575-1_3

Alnafjan, K. (2012). An empirical investigation into the adoption of Software Engineering Practice in Saudi Arabia. *IJCSI International Journal of Computer Science*, *9*(3), 328–332.

Ambler, S. (2002). Lessons in agility from Internet-based development. *IEEE Software*, *19*(2), 66–73. doi: 10.1109/52.991334

Asnawi, A. L., Gravell, A. M., & Wills, G. B. (2011). Empirical investigation on agile methods usage: Issues identified from early adopters in Malaysia. In *Lecture notes in business information processing* (Vol. 77 LNBIP, pp. 192–207). doi: 10.1007/978-3-642-20677-1_14

Asnawi, A. L., Gravell, A. M., & Wills, G. B. (2012, feb). Emergence of agile methods: Perceptions from software practitioners in Malaysia. In *Agile india (agile india), 2012* (pp. 30–39). doi: 10.1109/AgileIndia.2012.14

Ayed, H., Vanderose, B., & Habra, N. (2014). Supported approach for agile methods adaptation: an adoption study. In *Proceedings of the 1st international workshop on rapid continuous software engineering - rcose 2014* (pp. 36–41). doi: 10.1145/2593812.2593820

Azizyan, G., Magarian, M. K., & Kajko-Matsson, M. (2011, aug). Survey of Agile Tool Usage and Needs. In *2011 agile conference* (pp. 29–38). IEEE. doi: 10.1109/AGILE.2011.30

Bach, J. (1998, feb). Microdynamics of process evolution. *Computer*, *31*(2), 111–113. doi: 10.1109/2.652976

Badir Program Technology Incubator. (n.d.). *Information and Communication Technology Incubator | Badir Program Technology Incubator.* Retrieved 2015-05-26, from `InformationandCommunicationTechnologyIncubator|BadirProgramTechnologyIncubator.(n.d.).RetrievedMay26,2015,http://www.badir.com.sa/en/incubator/information-and-communication-technology-incubator`

Barki, H., & Suzanne Rivard, J. T. (2001). An integrative contingency model of software project risk management. *Journal of Management Information Systems*, *17*(4), 37–69. doi: 10.1080/07421222.2001.11045666

Beck, K. (1999). Embracing change with extreme programming. *Computer*, *32*(10), 70–77. doi: 10.1109/2.796139

Beck, K., & Andres, C. (2004). *Extreme Programming Explained: Embrace Change* (2nd ed.). Addison-Wesley.

Beck, K., Beedle, M., Bennekum, A. V., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). *Manifesto for Agile Software Development.* Retrieved 2014-12-01, from `http://agilemanifesto.org`

Boehm, B. (2002). Get ready for agile methods, with care. *Computer*, *35*(1), 64–69. doi: 10.1109/2.976920

Bosch, J., Holmström Olsson, H., Björk, J., & Ljungblad, J. (2013, jan). The Early Stage Software Startup Development Model: A Framework for Operationalizing Lean Principles in Software Startups. In B. Fitzgerald, K. Conboy, K. Power, R. Valerdi, L. Morgan, & K.-J. Stol (Eds.), *Lean enterprise software and systems* (Vol. 167, pp. 1–15). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-44930-7_1

Bosch, J., Olsson, H. H., Björk, J., & Ljungblad, J. (2013). *Lean Enterprise Software and Systems* (Vol. 167; B. Fitzgerald, K. Conboy, K. Power, R. Valerdi, L. Morgan, & K.-J. Stol, Eds.). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-44930-7

Caffery, F., Taylor, P., & Coleman, G. (2007, jan). Adept: A Unified Assessment Method for Small Software Companies. *IEEE Software*, *24*(1), 24–31. doi: 10.1109/MS.2007.3

Campanelli, A. S., & Parreiras, F. S. (2015, dec). Agile methods tailoring – A systematic literature review. *Journal of Systems and Software*, *110*, 85–100. doi: 10.1016/j.jss.2015.08.035

Cao, L., & Ramesh, B. (2008, jan). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software*, *25*(1), 60–67. doi: 10.1109/MS.2008.1

Carmel, E. (1994). Time-to-completion in software package startups. In *Proceedings of the twenty-seventh hawaii international conference on system sciences hicss-94* (Vol. 4, pp. 498–507). IEEE Comput. Soc. Press. doi: 10.1109/HICSS.1994.323468

Cater-Steel, A., Toleman, M., & Rout, T. (2006, may). Process improvement for small firms: An evaluation of the RAPID assessment-based method. *Information and Software Technology*, *48*(5), 323–334. doi: 10.1016/j.infsof.2005.09.012

Cockburn, A. (2004). *Crystal clear: A human-powered methodology for small teams* (first ed.). Addison-Wesley Professional.

Cockburn, A. (2006). *Agile Software Development: The Cooperative Game* (second ed.). Addison-Wesley Professional.

Cohen, D., Lindvall, M., & Costa, P. (2004). An introduction to agile methods. In (Vol. 62, p. 1 - 66). Elsevier. doi: 10.1016/S0065-2458(03)62001-2

Cohen, S., & Hochberg, Y. V. (2014). Accelerating Startups: The Seed Accelerator Phenomenon. *SSRN Electronic Journal*. doi: 10.2139/ssrn.2418000

Coleman, G. (2005, jan). An Empirical Study of Software Process in Practice. In *Proceedings of the 38th annual hawaii international conference on system sciences* (pp. 315c–315c). IEEE. doi: 10.1109/HICSS.2005.86

Coleman, G., & O'Connor, R. (2007, jun). Using grounded theory to understand software process improvement: A study of Irish software product companies. *Information and Software Technology*, *49*(6), 654–667. doi: 10.1016/j.infsof.2007.02.011

Coleman, G., & O'Connor, R. V. (2008, oct). An investigation into software development process formation in software start-ups. *Journal of Enterprise Information Management*, *21*(6), 633–648. doi: 10.1108/17410390810911221

Conboy, K., & Fitzgerald, B. (2010, jun). Method and developer characteristics for effective agile method tailoring. *ACM Transactions on Software Engineering and Methodology*, *20*(1), 1–30. doi: 10.1145/1767751.1767753

Creswell, J. W. (2014). *Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research* (4th ed.). Pearson.

Crowne, M. (2002). Why software product startups fail and what to do about it. Evolution of software product development in startup companies. In *Ieee international engineering management conference* (Vol. 1, pp. 338–343). IEEE. doi: 10.1109/IEMC.2002.1038454

da Silva, A. F., Kon, F., & Torteli, C. (2005). XP South of the Equator: An eXPerience Implementing XP in Brazil. In *Lecture notes in computer science* (Vol. 3556, pp. 10–18). doi: 10.1007/11499053_2

de O. Melo, C., Santos, V., Katayama, E., Corbucci, H., Prikladnicki, R., Goldman, A., & Kon, F. (2013, nov). The evolution of agile software development in Brazil. *Journal of the Brazilian Computer Society*, *19*(4), 523–552. doi: 10.1007/s13173-013-0114-x

De Souza Mariz, L. M. R., França, A. C. C., & Da Silva, F. Q. B. (2010, sep). An empirical study on the relationship between the use of agile practices and the success of software projects that use scrum. In *Proceedings - 24th brazilian symposium on software engineering, sbes 2010* (pp. 110–117). New York, New York, USA: IEEE. doi: 10.1109/SBES.2010.17

Diebold, P., & Dahlem, M. (2014). Agile practices in practice. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering - ease '14* (pp. 1–10). New York, New York, USA: ACM Press. doi: 10.1145/2601248.2601254

Diebold, P., & Zehler, T. (2015). The agile practices impact model: idea, concept, and application scenario. In *Proceedings of the 2015 international conference on software and system process - icssp 2015* (pp. 92–96). New York, New York, USA: ACM Press. doi: 10.1145/2785592.2785609

Dingsøyr, T., Dybå, T., & Abrahamsson, P. (2008). A preliminary roadmap for empirical research on agile software development. In *Proceedings - agile 2008 conference* (pp. 83–94). doi: 10.1109/Agile.2008.50

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, *85*(6), 1213–1221. doi: 10.1016/j.jss.2012.02.033

Donaldson, L. (2001). *The contingency theory of organizations*. Sage.

Dubai Internet City;Frost and Sullivan. (2012). *The Role of Entrepreneurship and Small amd Medium Enterprises (SME) in the Development of the ICT Industry* (Tech. Rep.). Dubai: Dubai Internet City; Frost and sullivan. Retrieved from www.in5.ae/resources/download/dic.pdf

Dybå, T., & Dingsøyr, T. (2008, aug). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, *50*(9-10), 833–859. doi: 10.1016/j.infsof.2008.01.006

Erickson, J., Lyytinen, K., & Siau, K. (2005, jan). Agile Modeling, Agile Software Development, and Extreme Programming. *Journal of Database Management*, *16*(4), 88–100. doi: 10.4018/jdm.2005100105

Fayad, M. E., Laitinen, M., & Ward, R. P. (2000, March). Thinking objectively: Software engineering in the small. *Communications of the ACM*, *43*(3), 115–118. doi: 10.1145/330534.330555

Finch, H. (2005). Comparison of Distance Measures in Cluster Analysis with Dichotomous Data. *Journal of Data Science*, *3*, 85–100.

Flat6Labs Jeddah. (2015). *Companies | Flat6Labs Jeddah.* Retrieved 2015-06-14, from http://www.flat6labsjeddah.com/en/companies

Gandomani, T. J., Zulzalil, H., Ghani, A. A. A., Md. Sultan, A. B., & Sharif, K. Y. (2014). An Exploratory Study on Managing Agile Transition and Adoption. In (Vol. 265, pp. 177–188). doi: 10.1007/978-3-319-06538-0_18

Giardino, C., & Paternoster, N. (2012). *Software development in startup companies* (Doctoral dissertation). Retrieved from `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.390.208&rep=rep1&type=pdf`

Giardino, C., Unterkalmsteiner, M., Paternoster, N., Gorschek, T., & Abrahamsson, P. (2014, sep). What Do We Know about Software Development in Startups? *Software, IEEE*, *31*(5), 28–32. doi: 10.1109/MS.2014.129

Hair Jr, J. F., Hult, G. T. M., Ringle, C., & Sarstedt, M. (2013). *A primer on partial least squares structural equation modeling (PLS-SEM)*. Sage Publications.

Hajjdiab, H., & Taleb, A. S. (2011). Agile adoption experience: A case study in the U.A.E. In *Icsess 2011 - proceedings: 2011 ieee 2nd international conference on software engineering and service science* (pp. 31–34). doi: 10.1109/ICSESS.2011.5982247

Hajjdiab, H., Taleb, A. S., & Ali, J. (2012). An industrial case study for Scrum adoption. *Journal of Software*, *7*(1), 237–242. doi: 10.4304/jsw.7.1.237-242

Hanisch, B., & Wald, A. (2012, jun). A Bibliometric View on the Use of Contingency Theory in Project Management Research. *Project Management Journal*, *43*(3), 4–23. doi: 10.1002/pmj.21267

Islam, J. (2012, apr). A review of literature on contingency theory in managerial accounting. *AFRICAN JOURNAL OF BUSINESS MANAGEMENT*, *6*(15). doi: 10.5897/AJBM11.2764

Jain, a. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys*, *31*(3), 264–323. doi: 10.1145/331499.331504

Jalali, S., & Wohlin, C. (2010, aug). Agile Practices in Global Software Engineering - A Systematic Map. In *2010 5th ieee international conference on global software engineering* (pp. 45–54). IEEE. doi: 10.1109/ICGSE.2010.14

Kajko-Mattsson, M., & Nikitina, N. (2008, dec). From Knowing Nothing to Knowing a Little: Experiences Gained from Process Improvement in a Start-Up Company. In *2008 international conference on computer science and software engineering* (pp. 617–621). IEEE. doi: 10.1109/CSSE.2008.1370

Kalus, G., & Kuhrmann, M. (2013). Criteria for software process tailoring: a systematic review. In *Proceedings of the 2013 international conference on software and system process - icssp 2013* (p. 171). New York, New York, USA: ACM Press. doi: 10.1145/2486046.2486078

Kaufman, L., & Rousseeuw, P. J. (2009). *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons.

Kelly, D., & Culleton, B. (1999). Process improvement for small organizations. *Computer*, *32*(10), 41–47. doi: 10.1109/2.796108

K.Flora, H., V. Chande, S., & Wang, X. (2014, may). Adopting an Agile Approach for the Development of Mobile Applications. *International Journal of Computer Applications*, *94*(17), 43–50. doi: 10.5120/16454-6199

Kitchenham, B., & Pfleeger, S. L. (2002, sep). Principles of survey research. *ACM SIGSOFT Software Engineering Notes*, *27*(5), 17. doi: 10.1145/571681.571686

Krasteva, I., Ilieva, S., & Dimov, A. (2010). Experience-based approach for adoption of agile practices in software development projects. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture*

Notes in Bioinformatics), *6051 LNCS*, 266–280. doi: 10.1007/978-3-642-13094-6_22

Krzanik, L., Rodriguez, P., Similä, J., Kuvaja, P., & Rohunen, A. (2010). Exploring the transient nature of agile project management practices. In *Proceedings of the annual hawaii international conference on system sciences* (pp. 1–8). doi: 10.1109/HICSS.2010.204

Kurapati, N., Manyam, V. S. C., & Petersen, K. (2012). Agile Software Development Practice Adoption Survey. In C. Wohlin (Ed.), *Agile processes in software engineering and extreme programming* (pp. 16–30). Springer Berlin Heidelberg. doi: 10.1007/978-3-642-30350-0_2

Lee, S., & Yong, H.-S. (2013, mar). Agile Software Development Framework in a Small Project Environment. *Journal of Information Processing Systems*, *9*(1), 69–88. doi: 10.3745/JIPS.2013.9.1.069

Lethbridge, T. C., Sim, S. E., & Singer, J. (2005, jul). Studying Software Engineers: Data Collection Techniques for Software Field Studies. *Empirical Software Engineering*, *10*(3), 311–341. doi: 10.1007/s10664-005-1290-x

MacCormack, A. (2001). Product-development practices that work: How internet companies build software. *MIT Sloan Management Review*, *42*(2), 75.

Manifesto for Agile Software Development (Vol. 2009) (No. December 14). (2001). Retrieved 2015-01-10, from `http://agilemanifesto.org/`

Manyam, V. S. C., & Kurapati, N. (2011). *Empirical investigation on adoption and adaptation of agile practices* (Unpublished doctoral dissertation).

Marmer, M., Herrmann, B. L., Dogrultan, E., Berman, R., Eesley, C., & Blank, S. (2012). *Startup Ecosystem Report 2012* (Tech. Rep.). Abingdon, UK: Technical report, Startup Genome. Retrieved from `http://www.tandfebooks.com/action/showBook?doi=10.4324/9780203165829_PART_ONE` doi: 10.4324/9780203165829_PART_ONE

Martin, K., & Hoffman, B. (2007, jan). An Open Source Approach to Developing Software in a Small Organization. *IEEE Software*, *24*(1), 46–53. doi: 10.1109/MS.2007.5

Misra, S., Kumar, V., Kumar, U., Fantazy, K., & Akhter, M. (2012, oct). Agile software development practices: evolution, principles, and criticisms. *International Journal of Quality & Reliability Management*, *29*(9), 972–980. doi: 10.1108/02656711211272863

Misra, S. C., Kumar, V., & Kumar, U. (2010). Identifying some critical changes required in adopting agile practices in traditional software development projects. *International Journal of Quality & Reliability Management*, *27*(4), 451–474. doi: 10.1108/02656711011035147

Mohamed, S., Baharom, F., & Deraman, A. (2014, may). An Exploratory Study on Agile based Software Development Practices. *International Journal of Software Engineering and its Applications*, *8*(5), 85–114. doi: 10.14257/ijseia.2014.8.5.09

Mooi, E., & Sarstedt, M. (2011). *A Concise Guide to Market Research*. Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-12541-6

Murray, C. (2008). *Lean and agile software development: a case study* (Doctoral dissertation, Massachusetts Institute of Technology). Retrieved from `http://dspace.mit.edu/handle/1721.1/43176`

Murtagh, F., & Legendre, P. (2014). Ward's Hierarchical Agglomerative Clustering

Method: Which Algorithms ImplementWard's Criterion? *Journal of Classification*, *31*(October), 274–295. doi: 10.1007/s00357-

Nardi, P. M. (2002). *Doing survey research* (1st ed.). Pearson.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005, may). Challenges of migrating to agile methodologies. *Communications of the ACM*, *48*(5), 72–78. doi: 10.1145/1060710.1060712

O'Connor, R. V., & Duchonova, N. (2014). Assessing the Value of an Agile Coach in Agile Method Adoption. In *Communications in computer and information science* (Vol. 425, pp. 135–146). Springer Berlin Heidelberg. doi: 10.1007/978-3-662-43896-1_12

Palmer, S. R., & Felsing, M. (2001). *A Practical Guide to Feature-Driven Development* (1st ed.). Pearson Education.

Papatheocharous, E., & Andreou, A. S. (2013). Evidence of Agile Adoption in Software Organizations: An Empirical Survey. In *Communications in computer and information science* (Vol. 364 CCIS, pp. 237–246). doi: 10.1007/978-3-642-39179-8_21

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2014, apr). Software Development in Startup Companies: A Systematic Mapping Study. *Information and Software Technology*. doi: 10.1016/j.infsof.2014.04.014

Pikkarainen, M., Salo, O., Kuusela, R., & Abrahamsson, P. (2012). Strengths and barriers behind the successful agile deployment-insights from the three software intensive companies in Finland. *Empirical Software Engineering*, *17*, 675–702. doi: 10.1007/s10664-011-9185-5

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

Qumer, A., & Henderson-Sellers, B. (2008, nov). A framework to support the evaluation, adoption and improvement of agile methods in practice. *Journal of Systems and Software*, *81*(11), 1899–1919. doi: 10.1016/j.jss.2007.12.806

Rauf, A., & AlGhafees, M. (2015, aug). Gap Analysis between State of Practice and State of Art Practices in Agile Software Development. In *2015 agile conference* (pp. 102–106). IEEE. doi: 10.1109/Agile.2015.21

Reis, E. (2011). *The Lean Startup: How Constant Innovation Creates Radically Successful Businesses*. Portfolio Penguin.

Richardson, I., & Von Wangenheim, C. (2007, jan). Guest Editors' Introduction: Why are Small Software Organizations Different? *IEEE Software*, *24*(1), 18–22. doi: 10.1109/MS.2007.12

Robinson, H., & Sharp, H. (2005, jul). Organisational culture and XP: three case studies. In *Agile development conference (adc'05)* (pp. 49–58). IEEE Comput. Soc. doi: 10.1109/ADC.2005.36

Rodríguez, P., Markkula, J., Oivo, M., & Turula, K. (2012). Survey on agile and lean usage in finnish software industry. In *Proceedings of the acm-ieee international symposium on empirical software engineering and measurement - esem '12* (p. 139). New York, New York, USA: ACM Press. doi: 10.1145/2372251.2372275

Rohunen, A., Rodriguez, P., Kuvaja, P., Krzanik, L., & Markkula, J. (2010). Approaches to agile adoption in large settings: A comparison of the results from a literature analysis and an industrial inventory. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *6156 LNCS*, 77–91. doi: 10.1007/978-3-642-13792-1_8

Salo, O., & Abrahamsson, P. (2005, nov). Integrating agile software development and software process improvement: a longitudinal case study. In *2005 international symposium on empirical software engineering, 2005.* (pp. 187–196). IEEE. doi: 10.1109/ISESE.2005.1541828

Saripalli, P. S., & Darse, D. H. P. (2011). *Finding common denominators for agile software development: a systematic literature review* (Master's thesis, Blekinge Institute of Technology). Retrieved from http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A830452&dswid=-1885

Saudi Business Incubator Network. (n.d.). *Business Incubators.* Retrieved 2015-05-26, from http://sbin.org.sa/en/members/incubators

Schindler, C. (2008, Dec). Agile software development methods and practices in austrian it-industry: Results of an empirical study. In *Computational intelligence for modelling control automation, 2008 international conference on* (p. 321-326). doi: 10.1109/CIMCA.2008.100

Schwaber, K., & Beedle, M. (2001). *Agile Software Development with Scrum* (1st ed.). Upper Saddle River, NJ, USA: Prentice Hall PTR.

Sekaran, U., & Bougie, R. (2010). *Research methods for business* (fifth ed.). Wiley.

Shah, D. (2006). *On Startups : patterns and practices of contemporary software entrepreneurs* (Doctoral dissertation, Massachusetts Institute of Technology). Retrieved from http://hdl.handle.net/1721.1/37253

Shore, J., & Warden, S. (2007). *The Art of Agile Development* (1st edition ed.). O'Reilly Media.

Sison, R., & Yang, T. (2007, dec). Use of Agile Methods and Practices in the Philippines. In *14th asia-pacific software engineering conference (apsec'07)* (pp. 462–469). IEEE. doi: 10.1109/ASPEC.2007.35

Sjoberg, D. I. K., Dyba, T., & Jorgensen, M. (2007, may). The Future of Empirical Methods in Software Engineering Research. In *Future of software engineering (fose '07)* (pp. 358–378). Washington, DC, USA: IEEE. doi: 10.1109/FOSE.2007.30

Sommerville, I., & Ransom, J. (2005, jan). An empirical study of industrial requirements engineering process assessment and improvement. *ACM Transactions on Software Engineering and Methodology*, *14*(1), 85–117. doi: 10.1145/1044834.1044837

Stapleton, J. (2003). *DSDM: Business focused development* (2nd ed.). London: Addison-Wesley.

Stolberg, S. (2009, aug). Enabling Agile Testing through Continuous Integration. In *2009 agile conference* (pp. 369–374). IEEE. doi: 10.1109/AGILE.2009.16

Strode, D. E., Huff, S. L., & Tretiakov, A. (2009, jan). The Impact of Organizational Culture on Agile Method Use. In *2009 42nd hawaii international conference on system sciences* (pp. 1–9). IEEE. doi: 10.1109/HICSS.2009.436

Sulayman, M., & Mendes, E. (2010). Software and Web Process Improvement – Predicting SPI Success for Small and Medium Companies. In *Advances in software engineering* (pp. 120–129). Springer. doi: 10.1007/978-3-642-17578-7_13

Sutton, S. (2000, jul). The role of process in software start-up. *IEEE Software*, *17*(4), 33–39. doi: 10.1109/52.854066

Taipale, M. (2010). Huitale – A Story of a Finnish Lean Startup. In P. Abrahamsson & N. Oza (Eds.), *Lean enterprise software and systems se - 16* (Vol. 65, pp. 111–114). Springer Berlin Heidelberg. doi: 10.1007/978-3-642-16416-3_16

Tripp, J. F., & Armstrong, D. J. (2014, jan). Exploring the Relationship between Organizational Adoption Motives and the Tailoring of Agile Methods. In *2014 47th hawaii international conference on system sciences* (pp. 4799–4806). IEEE. doi: 10.1109/HICSS.2014.589

Valtanen, A., & Ahonen, J. J. (2008). Big Improvements with Small Changes: Improving the Processes of a Small Software Company. In *Product-focused software process improvement* (Vol. 5089 LNCS, pp. 258–272). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-540-69566-0_22

VersionOne. (2013). *8th Annual State of Agile Survey* (Tech. Rep.). VersionOne Inc. Retrieved from `http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf`

VersionOne. (2015). *10th Annual State of Agile Report* (Tech. Rep.). VersionOne Inc. Retrieved from `https://versionone.com`

Voas, J. (1999). Advice for those bitten by the startup bug [IT business]. *IT Professional*, *1*(3), 38–44. doi: 10.1109/6294.774952

von Wangenheim, C., Anacleto, A., & Salviano, C. (2006, jan). Helping small companies assess software processes. *IEEE Software*, *23*(1), 91–98. doi: 10.1109/MS.2006.13

West, D., & Grant, T. (2010, jan). *Agile development: Mainstream adoption has changed agility* (Tech. Rep.). Forrester Research.

Williams, L. (2010). Agile Software Development Methodologies and Practices. In *Advances in computers* (Vol. 80, pp. 1–44). doi: 10.1016/S0065-2458(10)80001-4

Williams, L., & Cockburn, A. (2003, jun). Agile software development: it's about feedback and change. *Computer*, *36*(6), 39–43. doi: 10.1109/MC.2003.1204373

Wyne, J., & Wamda Research Lab. (2014). *The Next Step: Breaking barriers to scale for MENA's entrepreneurs* (Tech. Rep.). Retrieved from `https://s3-eu-west-1.amazonaws.com/wrl-reports/english/wrl-nextstep-for-scale.pdf`

Xu, P., & Ramesh, B. (2007, oct). Software Process Tailoring: An Empirical Investigation. *Journal of Management Information Systems*, *24*(2), 293–328. doi: 10.2753/MIS0742-1222240211

Yang, C., Liang, P., & Avgeriou, P. (2016, jan). A systematic mapping study on the combination of software architecture and agile development. *Journal of Systems and Software*, *111*, 157–184. doi: 10.1016/j.jss.2015.09.028

Yau, A., & Murphy, C. (2013). *Is a Rigorous Agile Methodology the Best Development Strategy for Small Scale Tech Startups?* (Tech. Rep.).

APPENDIX A
QUESTIONNAIRE

Dear Respected Respondent,

You are invited to participate in this survey about agile adoption on software startups. It will take around 15 minutes to complete the questionnaire. All answers will be treated as strictly confidential and will be used for the purpose of the study only.

Thank you for your cooperation and the time taken in answering this questionnaire. If you have any questions regarding this research, you may address them to us at the contact details below.

Abdullah Mohammed Al-Sakkaf
School of Computing
Universiti Utara Malaysia

A.1 Section One: Demographic

A.1.1 Background of Practitioners

1. What best describes your current position in your organization?

   a) Programmer / Developer

   b) Project Manager

   c) Quality Assurance / Tester

   d) System Analyst

   e) IT management

   f) CTO

   g) CEO

   h) Other .....

2. What is your Education Level?

   a) Certificate

   b) Diploma

   c) Degree

   d) Master

   e) PhD

   f) Other .....

3. How long have you been participating in agile development methodologies?

   a) None

   b) Less than 3 years

   c) 3-5 years

   d) 6-10 years

   e) More than 10 years

A.1.2 Organization’s Background

1. What is the age of your organization?

   a) Less than 1 year

   b) 1-4 years

   c) 5-8 years

   d) 9 years and above

2. What is the number of employee in your organization?

   a) Less than 10 employee

   b) 10-20 employee

   c) 21-40 employee

   d) 41-100 employee

   e) 101-500 employee

   f) More than 500 employee

3. What is the number of employees in the development team in your organization?

   a) 1-5 employees

   b) 6-10 employees

   c) 11-20 employees

   d) 21-50 employees

   e) More than 50 employees

4. What is your organization main product type?

   a) Website

   b) Mobile application

   c) Desktop application

   d) Other .....

A.2 Section Two: Agile Adoption Motivation

1. Does your organization use agile development methodology?

    a) Yes

    b) No (If yes, continue to next question. If no, stop here)

2. How important were the following in your company's decision to initially adopt agile development methods in your organization? (Please circle your answer)

	Agile methodology adopted to	Not Important	Somewhat Important	Very Important	Highest Important
1	Accelerate product delivery	1	2	3	4
2	Enhance ability to manage changing priorities	1	2	3	4
3	Improve business and IT alignment	1	2	3	4
4	Increase team productivity	1	2	3	4
5	Enhance software quality	1	2	3	4
6	Improve project visibility	1	2	3	4
7	Reduce project risk	1	2	3	4
8	Reduce project cost	1	2	3	4
9	Simplify development process	1	2	3	4
10	Improve team morale	1	2	3	4
11	Increase software maintainability	1	2	3	4
12	Improve engineering discipline	1	2	3	4
13	Better manage distributed teams	1	2	3	4
14	Enhance delivery predictability	1	2	3	4

A.3 Section Three: Agile Adopted Practices

1. Which agile software development methodology has your organization adopted?

    a) Extreme programming (XP)

    b) Scrum

    c) Feature-driven development (FDD)

    d) Lean Development

    e) Crystal methods

    f) Dynamic systems development method (DSDM)

    g) Combination of Agile and other methods

    h) None

    i) Other ......

2. How many years has your company been using agile methodology?

    a) Not practicing agile

    b) <1 year

    c) 1-2 years

    d) 3-5 years

    e) 5+ years

    f) I don't know

3. How frequent do you apply the following agile practice on your startup? (Please circle your answer)

	Agile Practice	Never Used	Rarely Used	Sometimes Used	Very Often Used	Always Used
1	Daily Meeting Each day at the same time, the team meets so as to bring every-one up to date on the information that is vital for coordination.	1	2	3	4	5
2	Short iterations An iteration is a timebox during which development takes place, is in most cases fixed for the du-ration of a given project	1	2	3	4	5
3	Unit testing A unit test is a short program fragment written and maintained by the developers on the product team, which exercises some nar-row part of the product's source code and checks the results.	1	2	3	4	5

4	Retrospectives A time boxed meeting held at the end of an iteration, or at the end of a release, in which the team examines its processes to determine what succeeded and what could be improved.	1	2	3	4	5
5	Prioritized backlogs A backlog is a list of features or technical tasks which the team maintains and which, at a given moment, are known to be necessary and sufficient to complete a project or a release.	1	2	3	4	5
6	Team-based estimation The development team estimates stories together.	1	2	3	4	5
7	Coding standards Coding standard is an agreed upon set of rules that the entire development team agree for source code to adhere to throughout the project.	1	2	3	4	5
8	Continuous integration (CI) Integrating the newly produced code to system baseline frequently	1	2	3	4	5

	CI is a practice that requires developers to integrate code into a shared repository several times a day.					
9	Iteration reviews The iteration review meeting is held on the final day of the iteration that aims to successfully demonstrate the features and functions completed during the iteration.	1	2	3	4	5
10	Dedicated product owner The Product Owner leads the development effort by conveying his vision to the team, outlining work in the Product Backlog, and prioritizing it based on business value.	1	2	3	4	5
11	Single team There are only one team for development and testing. That means the developers testing the code themselves.	1	2	3	4	5

12	Refactoring Refactoring consists of improving the internal structure of an existing program's source code, while preserving its external behavior.	1	2	3	4	5
13	Open Work area The whole team has the use of a dedicated space for the duration of the project, set apart from other groups' activities.	1	2	3	4	5
14	Test-driven development (TDD) TDD refers to a style of programming in which three activities are tightly interwoven: coding, testing and design.	1	2	3	4	5
15	Story mapping Story Mapping is a technique, which helps teams visualize the bigger picture while keeping all user story elements in perspective. It also helps to stay updated about the project progress.	1	2	3	4	5

16	Collective code ownership All team members have authority to make changes at any part of the code.	1	2	3	4	5
17	Pair programming Pair programming consists of two programmers sharing a single workstation.	1	2	3	4	5
18	Automated acceptance testing Acceptance tests (also called Customer tests) describe black-box requirements, identified by your project stakeholders, which your system must conform to.	1	2	3	4	5
19	Iteration planning Performed at the beginning of each iteration to plan iterations and break up stories into smaller tasks.	1	2	3	4	5
20	Continues deployment Continuous deployment is aiming to minimizing the time elapsed between development writing one new line of code and this new code being used by live users, in production.	1	2	3	4	5

21	Release planning The release plan specifies which user stories are going to be implemented for each system release and dates for those releases,	1	2	3	4	5
22	Taskboard A task board can be drawn on a whiteboard or even a section of wall, which is divided, into three columns labeled ”To Do”, ”In Progress” and ”Done”.	1	2	3	4	5
23	Behavior-driven development BDD is a software development methodology in which an application is specified and designed by describing how its behavior should appear to an outside observer.	1	2	3	4	5
24	User Story An Agile requirement, stated as a sentence or two of plain English. A user story is often expressed from the user’s point of view, and describes a unit of desired functionality.	1	2	3	4	5

APPENDIX B

STARTUPS INCUBATORS AND ACCELERATORS LIST

This list is collection technology incubator and accelerator located in Saudi Arabia.

All listed companies were invited to participate in this study.

	Name	Website
1	Saudi Business Incubator Network	http://www.sbin.org.sa
2	Badir Incubator for Information and Communication Technology	https://badir.com.sa/ar/incubator/information-and-communication-technology-incubator
3	UQU Business & Innovation	https://uqu.edu.sa/bi
4	Innovation and Entrepreneurship Institute	http://iei.uqu.edu.sa
5	GIS Technology Innovation Center	http://www.gistic.org
6	IE Community	http://www.iecommunity.org
7	Programize Me	http://programize.me
8	Makkah Techno Valley Company	http://www.mtvc.com.sa
9	KAUST Innovation Fund	http://innovation.kaust.edu.sa/entrepreneurs/
10	Prince Salman Institute for Entrepreneurship, King Saud University	https://alriyadah.ksu.edu.sa
11	Center of Creativity and Entrepreneurship, King Abdulaziz University	http://alliance.kau.edu.sa/Pages-الرئيسية-لحاضنات-الأعمال.aspx
12	Flat6 labs Jeddah	http://www.flat6labsjeddah.com/en
13	Endeavor Saudi Arabia	http://www.endeavor.sa
14	the centennial fund	http://www.tcf.org.sa/
15	InspireU	http://iu.stc.sa/ar/contact-us/
16	Qotuf	http://www.qotuf.com
17	Oqal angel investor group in Saudi Arabia	http://oqal.org/
18	Aramco Entrepreneurship Center	http://waed.net
19	Entrepreneurship Institute at KFUPM	http://ei.kfupm.edu.sa
20	Mobily Ventures	http://mobilyventures.com
21	Badir – Oasis500	http://www.badiroasis500.com
22	SIRB	http://sirb.sa/
23	Prince Sattam bin Abdulaziz University Incubator	https://badir.com.sa/en/incubator/prince-sattam-bin-abdulaziz-university-incubator
24	Taibah University Technolgoy Incubator	https://www.taibahu.edu.sa
25	Badir Technology Incubator in Taif	https://badir.com.sa/en/incubator/taif-incubator

APPENDIX C

AGILE PRACTICES DESCRIPTIVE STATISTICS

The full descriptive statistics for agile practices order by the highest mean.

Practice	N	Minimum	Maximum	Mean	Std. Deviation
Prioritized backlogs	64	1	5	3.84	1.027
Open work area	64	1	5	3.80	1.184
Continuous integration	64	1	5	3.75	1.168
Coding standards	64	1	5	3.75	1.127
Story mapping	64	1	5	3.64	0.982
Release planning	64	1	5	3.61	1.203
Refactoring	64	1	5	3.58	1.081
Taskboard	64	1	5	3.55	1.425
Single team	64	1	5	3.55	1.221
Team-based estimation	64	1	5	3.48	1.054
Collective code ownership	64	1	5	3.47	1.140
User Story	64	1	5	3.45	1.379
Continues deployment	64	1	5	3.42	1.270
Dedicated product owner	64	1	5	3.39	1.217
Short iterations	64	1	5	3.39	1.136
Iteration reviews	64	1	5	3.30	1.150
Iteration planning	64	1	5	3.22	1.266
Daily Meeting	64	1	5	3.22	1.362
TDD	64	1	5	3.09	1.137
Retrospectives	64	1	5	3.08	1.159
Unit testing	64	1	5	3.02	1.215
Automated acceptance testing	64	1	5	2.84	1.250
BDD	64	1	5	2.61	1.229
Pair programming	64	1	5	2.52	1.260
Valid N (listwise)	64				