UNIVERSITY OF NAIROBI

COLLEGE OF BIOLOGICAL AND PHYSICAL SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS

Investigate the Software Development Methodologies in Practice: A Case of Software Producing Firms in Nairobi

CONSERAY S MABEYA

REG NO. P54/6372/2017

A RESEARCH PROJECT REPORT SUBMITTED TO THE SCHOOL OF COMPUTING AND INFORMATICS IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF MASTER OF SCIENCE IN INFORMATION TECHNOLOGY MANAGEMENT, UNIVERSITY OF NAIROBI

2019

Declaration

This is my original work that has not been presented for a research project report in any other university by any other person in part or whole.

Name: **CONSERAY S. MABEYA** REG NO. **P54/6372/2019**

Signature……..... Date……

This Project report has been presented with my approval as the university supervisor.

DR. AGNES WAUSI

Signature……. Date……

Abstract

The role of software development methodology firms and products is increasingly becoming more important in developing countries. This informs the need for software development firms operating (or intending to) in such context to adopt software development methodology/practices that fits well with the context of developing countries in order to stay competitive and ensure their long-term sustainability. The very nature of software developing firms being small in size, and operating in markets such as developing countries puts pressure on them to determine the software development methodologies that they can effectively use in such context, determine the benefits that they can derive from using the given software development methodologies, and also determining the factors influencing software development methodologies in such contexts. This research aimed at investigating the software development methodologies in practice in Kenya.

The research revealed that software development firms in Kenya largely use the agile methodology, this is mainly focused on perception that it can derive the benefits related to having enhanced controlled and offer helpful guidance. Additionally, the study also established that there is a mix of target market for software developing firms in Kenya mainly led by Banking financial institutions and governments (or public enterprises); Java programming language is the most commonly used language; the software firms largely engaging in new software development, closely followed by system integration.

The research recommends that in order to be successful in the Kenyan software market, firms should use Java application and agile methodology. However, they need to adapt to the dynamic of the market and customers' needs. Secondly, the research recommends that more needs to be done to establish attitude that influence software methodology usage. Thirdly, the research recommends that more studies needs to carried out to establish depth of usage of software engineering practices.

Dedication

I am grateful to the Almighty God for virtues of patience, tolerance, strength and peace of mind that has helped me to come this far. I am indebted to my project supervisors, for their valuable time, suggestions, ideas, criticism and guidance in the preparation of this document. With their years of expertise and experience, they have been instrumental in the formulation of the work contained herein. I would like to acknowledge the input of the departmental lecturers, students, and staff who have molded me, in a way, to have the ability to think critically and look beyond the obvious. Finally, I would like to thank my wife, Faith and my two sons Ray And Remy for the encouragement and moral support.

Definition of terms

Term	Definition
ICT	Information communication and technology
CMMI	Capability Maturity Model Integration a model for software process improvement
SPICE	Software process improvement Capability Determination
SEI	Software Engineering Institute
TAM	Technology Acceptance Model
SDLC	Software Development Life Cycle
SDM	System Development methodologies
SWEBOK	Software Engineering Body of Knowledge
PLS SEM	Partial Least Square Structured Equation Modelling
SD	Software Development

TABLE OF CONTENTS

Declaration i

Definition of terms iii

TABLE OF CONTENTS iv

List of Tables vi

CHAPTER 1: INTRODUCTION 1

1.1 Background of the Study 1

1.2 Problem Statement 2

1.3 Research Objectives 3

1.4 Research questions 3

1.5 The significance of the research 4

CHAPTER 2: LITERATURE REVIEW 5

2.1 Introduction 5

2.2 Software practices concepts 5

2.2.1 Software Process 5

2.2.2 Software Development Life Cycle 6

2.2.3 Software development methodologies 6

2.2.3.1 Waterfall Model 7

2.2.3.2 Spiral Model 7

2.2.3.3 Prototyping Model 7

2.2.3.4. Incremental model 7

2.2.9. Agile methodologies 8

2.3 Factors that influence selection of a software development model 8

2.3 Trends in Software development methodology adoption and usage 9

2.3.1 Capability Maturity Model Integration (CMMI) 13

2.3.2 ISO/IEC 15504 standard 14

2.3.3 Software Engineering Body of Knowledge 14

2.4 Theoretical Models of acceptance 15

2.5 Unified Methodology Adoption Model (UMAM) 15

2.6 Conceptual Framework 16

2.6.1 Hypothesis Development 18

2.7 Summary of Literature Review. 18

CHAPTER 3: RESEARCH METHODOLOGY 20
3.1. Introduction 20
3.2. Research Design 20
3.3. Sample Population and Size 20
3.4 Sampling Technique 20
3.5. Data collection 21
3.5.1. Data Sources 21
3.5.2. Research Instruments 21
3.5.3. Data collection procedures 22
3.6. Operationalization of Variables 22
3.5. Reliability and validity of Research instrument 23
3.6. Pilot testing 23
3.9 Data Analysis 24
3.10 Model Estimation and Hypotheses Testing 24
3.11 Ethical Considerations 25
CHAPTER 4: FINDINGS AND DISCUSSIONS 26
4.1 Introduction 26
4.2 Response Rate 26
4.3 Descriptive statistics 26
4.3.1 Size of the organization 26
4.3.2 Role of the respondents 27
4.3.3 Target customers of the software product 28
4.2.4 The characteristics of the project (s) undertaken by the organizations 30
4.2.5 Average Project Duration 31
4.2.6 Programming Language used 32
4.2.7 Standard process used 33
4.2.8 Analysis of approaches employed on various Software development Project-related activities 34
4.2. 9 Perceived Benefits frequencies 37
4.3 Hypothesis testing 39
4.3.1 SmartPLS Bootstrapping Direct Effect Results 39
4.3.2 Hypothesis testing summary 41

CHAPTER 5: CONCLUSION AND RECOMMENDATIONS 45
5.0 Overview 45
5.1 Conclusions 46
5.2 Recommendations 47
5.3 Limitations 48
6. REFERENCES 49

List of Tables

Table 2. 2 Literature review summary 19

Table 3 1 Operationalization of Variables 23

Table 4. 1 Size of the organisation 27
Table 4. 2 Primary role in the organisation 28
Table 4. 3 Customer target 28
Table 4. 4 Type of Projects 30
Table 4. 5 Average duration of the projects 31
Table 4. 6 programming language used 32
Table 4. 7 standard process 33
Table 4. 8 Summary of Software development approaches adopted by firms 35
Table 4. 9 Approaches employed per each SWEBOK practice 36
Table 4. 10: Methodologies used 37
Table 4. 11 Perceived benefits on software development methodologies frequencies 39
Table 4. 12 Hypothesis testing summary 41

List of Figures

Figure 2 1: Software Process 5
Figure 2 2: SLDC phases 6
Figure 2 3: Chaos resolution by software methodology 11
Figure 2 4: UMAM Model 16
Figure 2 5: Conceptual Framework 17
Figure 4. 2: Reduced Model 40

Figure 4. 1: SDMU Research Model and PLS-SEM results 40

CHAPTER 1: INTRODUCTION

1.1 Background of the Study

Software runs enterprises today by playing a role that is critical in the day-to-day business operations, which is crucial in giving organizations a competitive edge. Organizations overly rely on software; thus any software failure may result in big losses. In recent times, organizations have become aware of the consequences any failure of customer fronting applications; whereby once such occurrences are noticed and shared on social media, they may go viral in a short period and cause significant financial losses to enterprises (Otieno, 2017).

According to Kenya ICT master plan 2014-2017, the third pillar is to develop ICT enterprises that can generate and/or offer exportable quality products and services, which are equivalent to the best in the world. However, participating in global software development require incorporating best-in-class practices and standards in areas such as software development. According to Putta (2016), in order to participate and be successful in the global software development space, there is need for local software development organizations to understand industry trends, regarding common practices and the factors that drive high acceptance.

Software development methodologies(SDM) emerged from frustrations that characterized software projects in 1970s, through the need to structure, simplify, standardize, and improve the management and control of the software development process(Wynekoop & Russo, 1995). SDM provides a "framework for planning, executing, and managing the process of developing software systems"(Vijayasarathy & Butler, 2016).

Despite the importance placed on software development methodology usage, not many organizations have universally embraced them(Griffin & Brandyberry, 2010).

The 2015 Chaos report by Standish Group paints a sorry state in software engineering, by revealing that only 36% of the software projects surveyed were successful. The report also indicates 45% of the projects were challenged; implying cost or time overruns, and not meeting customer requirements. Moreover, 19% of the projects were also marked as failed. The Chaos Report also indicates that Agile projects had nearly four times success rate than waterfall projects (The Standish Group, 2015).

Software process improvement (SPI) is concerned with making quality software products within set timelines and budget (Anees, 2017). The most notable models include IS0 9001, CMMI, TQM, and Six Sigma. The Capability Maturity Model Integration (CMMI) is the de-facto SPI model, which was developed by the Software Engineering Institute (SEI) and administered by the CMMI Institute. CMMI is based on a set of best practices that address productivity, performance, costs, and stakeholder satisfaction, and is aimed at supporting the integration of processes, procedures, standards, and elevation of organizational performance. The CMMI Institute maturity profile report (2017) records year over year increase in a number of appraisals per year with 2017 marking an 18% increase. The report further asserts that countries such as China, United States, India, and Mexico are increasingly adopting the model. On the other hand, despite its expressed value, the same maturity report indicates that Africa still lags behind in a number of appraisals; reporting only 151 for over a decade. For instance, by examining the CMMI appraisal data accessible from CMMI institute's portal, Kenya has only one organization appraised, against 16,151 organizations across the world. Therefore, this suggests that software companies in Africa (Kenya) may not really be able to compete globally since organization that adopts CMMI tend to achieve greater performance (Margarida, Vidal, & Vieira, 2012). This forms the basis for this research in investigating the software development methodologies in practice, in the context of Software Producing firms in Nairobi, Kenya.

1.2 Problem Statement

Enterprises have become more dependent on information systems in the day to day running of their business. Development of these information systems has remained, to a considerable extent, a creative endeavor whose management is difficult. Many a time, software projects overrun their budgets, especially due to failure to deliver within the timeline or are failure to sign off because of quality issues. To solve this, IT professionals are often tempted to solve issues in isolation, thus leading to short-term achievements. However, in the long run, this poses a risk of these issues reoccurring, hence causing the same resolution cycle to continue, which eventually leads to crisis. Globally, software organizations have recognized the need to follow software development methodologies(Kuhrmann *et al*.,2017). But, there is little literature on software development practices in the Kenyan context. Therefore, it is important to identify practices and techniques mainstreamed in the software industry and factors that influence adoption of such in order to advise on software process improvement. This will allow improvement of software

development processes instead of bit by bit improvement of individual software systems or projects. Research problem can be summarized as:

1. Previous studies in Nairobi have not dealt in detail with usage of software methodology in software producing firms in Nairobi,
2. Factors that influence usage of software development methodology have not been well investigated.

1.3 Research Objectives

This study was aimed at examining software development methodologies in practice and factors influencing their usage in software development firms in Nairobi. Therefore, the main objectives of this research were:

1. To identify software development methodologies used by software development firms in Nairobi
2. To get determine what software development organization consider as benefits of using software development methodologies,
3. To determine factors influencing usage of software development methods in the Kenyan context.

1.4 Research questions

RQ 1: Which software development methodologies are currently used in Kenya and the frequency of their application?

RQ 2: what is the level of usage of the current methodology based on perceived ease of use in Kenya?

RQ 3: What factors influence the selection of software development methodology in Kenyan firms?

1.5 The significance of the research

To the best of researchers' knowledge, no study has been conducted locally to specifically investigate software development methodologies in Kenya. Therefore, this makes the study significance in;

- Contributing to the understanding of issues involved in system development methodology use/adoption
- Offering solutions to quality problems related to software methodologies and issues with methodology through the finding from this study.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter assesses a range of literature related to software development methodologies. This involves an overview of the theoretical models related to acceptance of software development methodologies, and software practices and concepts. Additionally, empirical studies relating to software practices are also reviewed. This will form a good basis for establishing the conceptual framework, and the hypothesis for the research in investigating the software development methodologies in practice, in the context of Software Producing firms in Nairobi, Kenya.

2.2 Software practices concepts

2.2.1 Software Process

Software process is a sequence of related activities that a software product has to undergo before it is released. Software processes are both technical and managerial in nature. Jalote (2008) argues that while developing software, there are several processes that are involved (as illustrated in figure 1). According to Jalote (2008), like business processes, social processes, training among others are not software development related, but they have direct impact on software development.

Figure 2 1: Software Process

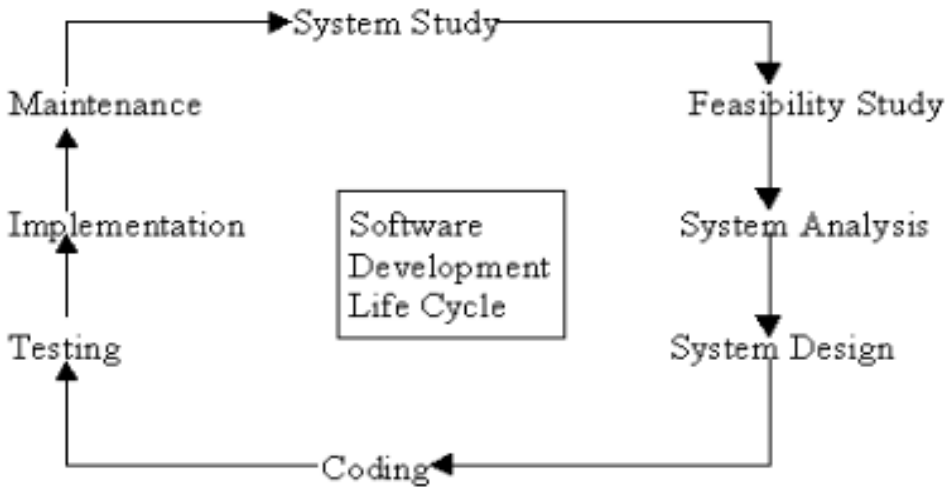
Adapted from Jalote (2008)

2.2.2 Software Development Life Cycle

The process of producing Software products usually evolve from one stage to another in a systematic way. Software Development Lifecycle (SDLC) is a framework which consists of various activities that software development stakeholders follow to produce software (as illustrated in figure 2). The context under which these activities are executed is largely dependent on user's needs (Apoorva & Deepty, 2013).

The phases are illustrated below:

Figure 2 2: SLDC phases



Adapted from Apoorva & Deepty (2013)

2.2.3 Software development methodologies

Several software development methodologies have emerged over the last decades in an effort to solve efficiency and management challenges in the software industry. Researchers and software practitioners have debated for a long time on how to classify these methods, with a general agreement to broadly group these models into two categories; the classical software engineering methodologies and agile methodologies (Jiang & Eberlein, 2009). The Classical software engineering methodologies are also referred to as traditional methodologies: These are often plan-driven, and require heavy upfront requirement specification, comprehensive plans, and advance documentation. Examples of these methodologies include Waterfall, Spiral, incremental and prototyping. On the other hand, agile methodologies include extreme programming XP and Scrum (Jiang & Eberlein, 2009).

2.2.3.1 Waterfall Model

The origin of this model can be traced back to 1970 in a study by Winston Royce (Macías-Escrivá *et al*., 2013). The Waterfall approach proposes a structured step by step predictive and sequential method, whereby requirements are planned, analyzed, followed by designing, coding, integration, and finally maintenance (Popli, Anita & Chauhan, 2012). According to Popli, Anita & Chauhan (2012), this methodology is usually document-driven, with deliverables being mandatory at each phase. The Waterfall model bears advantages from its simplicity, perceived ease of use, proper documentation, and upfront planning, which enhances the ease to track project status. However, the model also has its own shortcomings as it does not allow rigorous user involvement, assumes that user requirements do not change with time, and takes a long time for the project to be completed (Dasoriya, 2017). According to Jiang & Eberlein (2009), the Waterfall model is ideal for projects that have clear development phases, and follows software engineering principles that include customer involvement and quality requirements. Therefore, the model is suitable for large, long-term, and fixed contract projects.

2.2.3.2 Spiral Model

This method emerged in 1988, as a result of in the changing requirement challenges realized in the waterfall model during development (Macías-Escrivá *et al*., 2013). The phases in this model are similar to those in the waterfall model, but are carried out in an iterative manner. The advantage of this model is that it is risk conscious; in the sense that each cycle kicks off with risk analysis, and any outstanding risk issues are resolved before the next iteration. The downside of the model is that spiral projects require risk experts, and efforts towards risk analysis tend to have negative implications on project completion, hence consuming more time and cost(Kaur, 2015; Liviu DESPA, 2014; Wallin & Rikard, 2010).

2.2.3.3 Prototyping Model

This model is based on the idea that end users and developers can clearly determine critical requirements of the product from which a prototype is quickly developed. On user feedback, the prototype undergo constant modification until the user is satisfied(Dasoriya, 2017; Yu, 2018).

2.2.3.4. Incremental model

This model integrates sequential elements of the waterfall with the rapid iterative aspects of the prototype model (Yu, 2018). The model requires the product to be designed, implemented, and

tested in a sequence of iterations. It also follows the same upfront requirement definition approaches, similar to waterfall model(Popli *et al*., 2012; Yu, 2018) .

2.2.9. Agile methodologies

According to Popli *et al*. (2012) agile approaches emerged as a result of the challenges experienced in the traditional software development approaches, where those advocating for the former considered it for being heavyweight in processes and in documentation. Companies continue to embrace agility for the purposes of achieving quality as well as reducing costs and time associated with projects(Popli *et al*., 2012). Agile model consist of several development processes that share agility characteristics(Mall, 2014). The major models under agile umbrella include:

1. Crystal
2. Atern
3. Scrum
4. Extreme Programming
5. Lean Development
6. Unified process

2.3 Factors that influence selection of a software development model

Dasoriya(2017) suggest a criterion that can be used to select a model based on the project characteristics, which include;

1. Documentation- involves how demanding the project documentation is
2. Team Size- involves the size of the team required for a specific methodology. For instance, heavy methodologies like waterfall require large team as compared to light methodologies like agile and prototyping
3. Planning- projects that demand upfront documentation often require pre-planning as opposed to agile ones
4. Requirements- changing requirements is not considered a major issue on agile projects, while it is a crucial element in traditional methodology. Additionally, in traditional methods, system requirements are frozen before design phase begins.
5. Communication-There is close interaction between the user and the developer in agile models as compared to traditional models

Software development process is wholly dependent on situational factors in software development setting. Such factors characterize the project and environment, and include the requirements of the application under development, project stakeholders, the size of the team and project, personnel experience and uncertainty of the requirements (Clarke & O'connor, 2012; Peng Xu & Balasubramaniam, 2007).

According to Ozturk (2016), requirement stability, clarity, development time, project /product size, and complexity of the system should be taken in consideration, when selecting appropriate SDLC for a project. Griffin (2008a) also suggests that customer aspects can also be used in determining whether a software development methodology can be used or not. Griffin (2008a) further argues that besides software development methodologies, tools such as CASE tools, visual programming tools, and debuggers, play a significant role in ensuring the ease of software development process. Additionally, in their study to assess programming language impact on development and maintenance, Bhattacharya & Neamtiu(2011) concluded that using C++, as opposed to C programming language, significantly improved software quality and reduced the effort required to maintain resulting application.

2.3 Trends in Software development methodology adoption and usage

In a study to examine the factors that influence the use of software development methodologies, Khalifa & Verner (2000) developed a model to test a number of usage factors. They found out facilitating conditions (organizational support and development team size) and developer's beliefs (process and product quality), as the significant variables influencing usage of software development methodology. To operationalize the variables, they depended on previous literature that adopted the Triandis Model, as well as traditional software development methodologies, such as waterfall and prototyping. According to Khalifa & Verner (2000), measuring usage variable is better done by using depth of usage and breath of usage. Additionally, they established that the size of development team can be measured by the number of staff. Besides these, other dimensions that formed part of Khalifa & Verner's (2000) research included product quality, whose indicators are software quality and software maintainability, and process quality, measured by communication with the users, project control, cost of development, and detection of problems in earlier stages.

On the other hand, Geambaşu *et al*. (2011) indicate that there are a series of factors that influence the choice of a software development methodology. These factors include the precision of the initial requirements, the accuracy of the original estimation of costs and time of development, integration of changes in requirements during process of software development, attaining functional forms of the system during the development process, the usefulness of the software, the costs of development, the period of the delivery time of the ultimate system, the complexity of the system, the nature of communication between developers and users/customers, and size of the development team.

Moreover, studies in best practices in software development adoption have been carried out in many countries across the world including Europe(Dutta, Lee, & Van Wassenhove, 1999), Malaysia(Zainol & Mansoor, 2008), Turkey(Garousi, Coşkunçay, Betin-Can, & Demirörs, 2015), Saudi Arabia(Alanezi, 2018), Botswana(Ayalew & Motlhala, 2014), New Zealand(Kirk & Tempero, 2012) and South Africa(Tighy, 2012). Most study suggests that a majority of software development firms are small firms, especially in developing countries, including Botswana (Ayalew & Motlhala, 2014). As expressed by Griffin & Brandyberry (2010), the complexity of the software methodologies influenced the formation of light-weight methodologies, including the agile approach, which stresses on the generation of early functioning products with incremental capabilities, rather than prototyping. The 2015 Standish chaos indicates that agile projects outperformed waterfall projects by nearly four times, in terms of success rate, while waterfall projects had a failure rate of three times that of agile projects. The table below shows FY2011-2015 report segmented by agile and waterfall method.

Figure 2 3: Chaos resolution by software methodology

SIZE	METHOD	SUCCESSFUL	CHALLENGED	FAILED
All Size Projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Large Size Projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium Size Projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small Size Projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

Source: The Standish Group (2015)

A survey by Vijayasarathy & Butler (2016) on the use of software development methodologies among project managers and team members established that agile approaches, such as scrum and unified process, are dominant, and that classical methodologies were still popular. According to the survey, the factors that influence usage of methodology include; organization factors (such as revenue, number of employees), project factors (such as project criticality, and budget) and team factors (such as team size).

In their study to understand software engineering practices in Turkish firms, Garousi *et al*., (2015) adopted Software Engineering Body of Knowledge (SWEBOK) to design the online survey. The study assessed several practices, including software requirements, design, development, testing, maintenance, configuration management, release planning, and support practices. The findings from the study showed that among the software industries surveyed, Defense and Military were strong, and most practitioners were in these companies. Additionally, it was also noted that development phase takes most effort averaging at 31%, while software testing, requirements, design and maintenance averaged at 14%, 12%, 12% and 11%

respectively. According to the study, the Waterfall method was still preferred; with 53%, and incremental and agile/lean usage rate at 38% and 34%, respectively.

In the study to evaluate Software Engineering Management Best Practices based in Western Cape, Tighy (2012) focused on best practices in seven software development life cycle, namely Requirement and design, Code and test, Configuration management, Estimates and schedules, Project management, and Risk management. The study revealed that most organizations comply at 60% with configuration management, and 90% with project management reporting compliance. However, other practice categories reported worst performance; with design and documentation recording 20% compliance, controlling of change requirements, estimation and scheduling, and training also recording 20% compliance.

On the other hand, In Europe, a study by European Software Institute (ESI) to establish the extent to which software development organizations were using best practices, surveyed nearly 400 companies across several business sectors in 20 countries (Dutta *et al*., 1999). In designing the research instrument, the research examined CMM, ISO 9000 and SPICE software process improvement frameworks, and mapped out best practices by focusing on five key areas, which include:

1. Organizational issues: these relates to issues on whether project management function is executed appropriately, change control aspects, and training new appointee managers,
2. Standards and procedures involving existence of standardized processes for estimation, Quality, coding, independent audits, appropriate handover, and planning on test related activities,
3. Metrics, such as issue or error tracking, efficiency, project tracking, root cause analysis,
4. Control of development processes, which include project managers' authority to schedule, estimates, and over change control activities,
5. Tools and Technology for planning and scheduling, testing, requirement tracing, resource assignment, and prototyping.

The findings from the European Software Institute (ESI)'s study revealed that 51% of all practices had been adopted by the respondents. Adoption per each practice area showed organizational issues, standards and procedures, metrics, control of development process, Tools and technology, at 58%, 51%, 45%, 58%, and 45% respectively.

2.3.1 Capability Maturity Model Integration (CMMI)

CMMI is a process and behavioral model containing a set of best practices aimed at assisting businesses to improve their processes (Soydan & Kokar, 2012). The model was developed by SEI (software engineering Institute), and is currently being administered by CMMI institute and ISACA organization. Raninen (2014) cites that CMMI is used to provide guidance on processes that should be improved across projects, organization unit or the organization at large. Raninen (2014) further argues that the model helps in integrating silo organization functions, define goals and priorities for process improvement, offering direction for quality processes, and baseline for assessing current processes.

However, CMMI is either applied as continuous or staged (Chrissis, Konrad, & Shrum, 2011). According to Chrissis, Konrad, & Shrum (2011), Continuous representation enables users to concentrate on the specific processes considered critical to organization's immediate business goals, hence achieving capability levels. on the other hand, staged representation allows organizations to improve a set of related processes, and can serve as a guideline for comparing the maturity of different projects and organizations).By applying staged representation, organization can achieve five levels, where each level defines maturity of the process with corresponding process areas(Chrissis e*t al*., 2011). Maturity levels serve as a roadmap for ensuring organizations improve the consecutive pool of process areas incrementally. Staples & Niazi (2010) argues that organizations graduate, in terms of maturity, when they meet goals of process areas for the particular level and subsequent level.

According to Chrissis *et al*. (2011), CMMI comprises of several process areas, which in turn consists of related activities. In general, it consists of 22 process areas; with its best practices published in models which are designed to improve different business needs(Chrissis *et al*., 2011). The models include:

- Product and service development—CMMI for Development (CMMI-DEV)
- Service establishment, management—CMMI for Services (CMMI-SVC)
- Product and service acquisition—CMMI for Acquisition (CMMI-ACQ).

2.3.2 ISO/IEC 15504 standard

ISO/IEC 15504, also referred as SPICE (Software Process Improvement and Capability Determination), is an International Standard (ISO) model for Process Assessment, which was established in 1993. The process was advanced by JTC1/SC7's Working Group. Its origin can be traced to other maturity models, such as Bootstrap, Trillium and the CMM (Kozina & Kirinic, 2013). SPICE provide a document for process management, guidelines for carrying out an assessment and rating the target process, construction, selection, and use of assessment instruments and tools, and training for assessors. According to Kozina & Kirinic (2013), the SPICE model includes forty process areas classified into four categories; that is, primary lifecycle process, organizational processes, and support lifecycle processes. According to Ayalew & Motlhala (2014), ISO/IEC 15504 has been customized to match with the needs of small SMD firms.

2.3.3 Software Engineering Body of Knowledge

The Guide to the Software Engineering Body of Knowledge (SWEBOK), in its initiative to characterize content of software engineering discipline, identified 15 knowledge areas in its SWEBOK V3. The guide presents the knowledge area in hierarchical approach, in attempt to details activities associated with a particular process. These knowledge areas include (Pierre Bourque & Fairley, 2014):

1. Software Requirements
2. Software Design
3. Software Construction
4. Software Testing
5. Software Maintenance
6. Software Configuration Management
7. Software Engineering Management
8. Software Engineering Process
9. Software Engineering Models and Methods
10. Software Quality
11. Software Engineering Professional Practice

12. Software Engineering Economics
13. Computing Foundations
14. Mathematical Foundations
15. Engineering Foundations

2.4 Theoretical Models of acceptance

Researchers have used several theories to frame software development approach usage. Some of the theories include:

1. Technology Acceptance Model (TAM)
2. Technology Acceptance Model 2 (TAM 2)
3. The Theory of Planned Behavior (TPB)
4. Perceived Characteristics of Innovating (PCI),
5. Model of Personal Computer Utilization (MPCU)

Riemenschneider, Hardgrave & Davis (2002) utilized the five models in an attempt to explain individual developer acceptable of a methodology. The study found out that intentions to adopt a methodology were driven by; an organizational mandate to utilize the methodology, the methodology compatibility with how developers undertake their work, and the supervisors' and developers' coworkers' opinions on using the methodology.

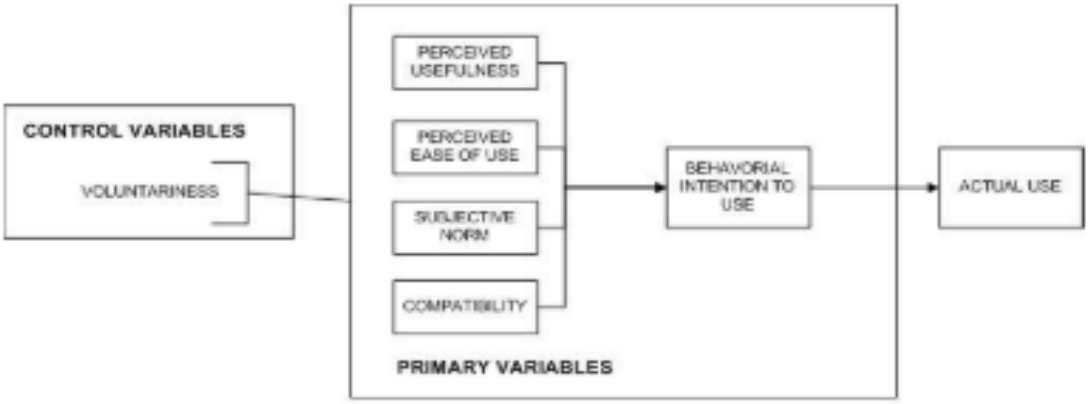
Davis(1989) introduced the Technology Acceptance Model (TAM) to predict and explain variables that influence use of system. He posits that the perceived usefulness and ease of use, jointly determine the intentions to use the systems. The ability to Work more quickly, job performance, increase in productivity, and the effectiveness in making the job easier and useful are assumed as the measurement scales for the perceived usefulness construct, while the ease to learn, controllability, clarity and understandability, flexibility, and ease to acquire skills and ease to use are assessed as the valid measurement scales for perceived ease of use variable (Chen, Shing-Han & Chien-Yi, 2011).

2.5 Unified Methodology Adoption Model (UMAM)

UMAM was developed by Diéguez, Sepúlveda, & Cachero (2012). Diéguez, Sepúlveda, & Cachero (2012) suggest that the UMAM model is simple enough to be understood by non-experts. This model was developed from the analysis of several proposals explaining factors that

influence methodology adoption (Diéguez, Sepúlveda, & Cachero, 2012). It consists of the following variables as illustrated below:

Figure 2 4: UMAM Model



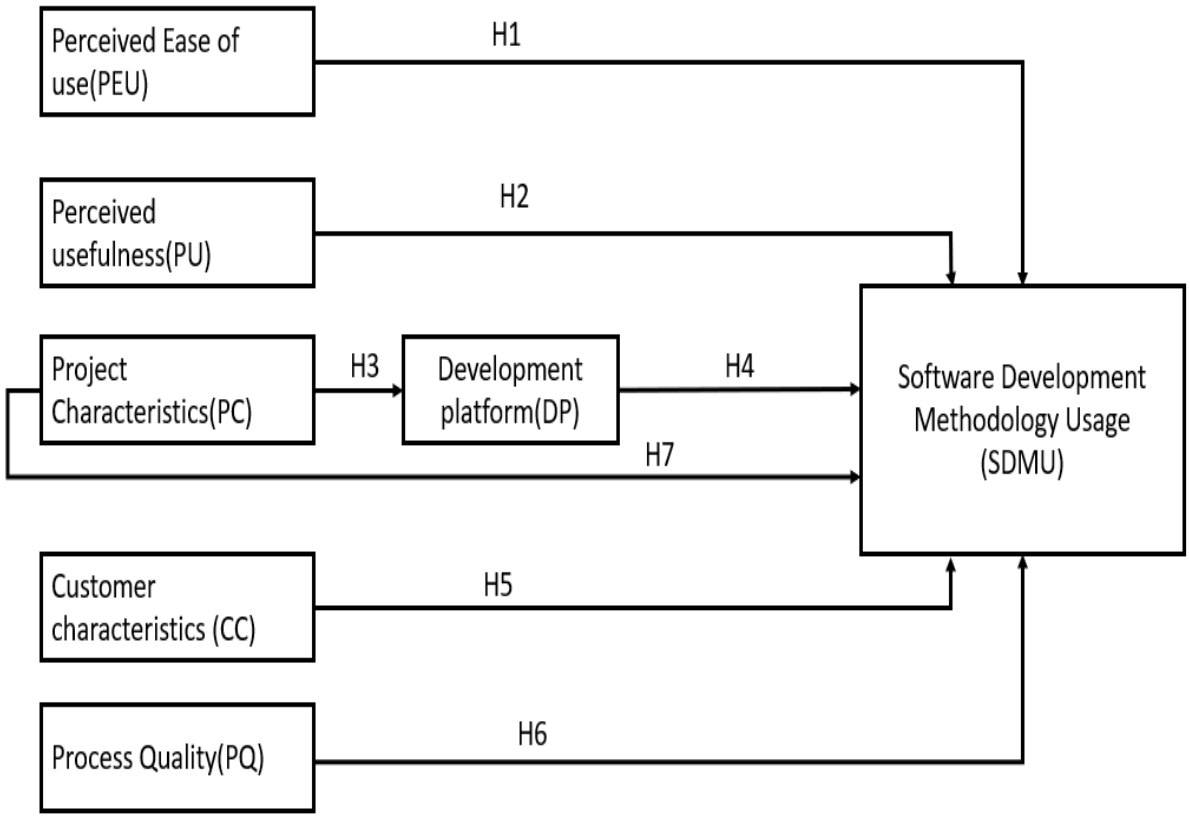
Adapted from Diéguez et al. (2012)

- **Perceived usefulness**: the extent to which a person observes using a particular method will improve performance of a task
- **Perceived ease of use**: belief that use of methodology will be free of effort
- **Subjective Norm**: extent to which developers believe that others who are important to them command that they should use the system
- **Compatibility:** the extent to which the methodology is perceived as coherent with existing values, practices, needs, and past experience of the potential adopter
- **Voluntariness:** degree to which potential adopter of the methodology discern the adoption decision is not mandatory

2.6 Conceptual Framework

According to Jabareen (2009) a conceptual framework represents a network of interlinked concepts that together offer a comprehensive understanding of phenomenon/phenomena. Drawing from literature review, a conceptual model is developed based on the relationship between the key variables as illustrated as illustrated below;

Figure 2 5: Conceptual Framework



Source: Researcher (2018)

In this study, the dependent variable is the software development methodology usage, which is the variable of primary interest. The study will attempt to explain the construct using three independent variables; that is the perceived usefulness, perceived ease of use, project characteristics, Process Quality, and customer characteristics.

2.6.1 Hypothesis Development

Hypothesis

H1: Individuals Perceived ease of use has a positive correlation on methodology usage,

H2: Individuals Perceived usefulness has a positive correlation with methodology usage,

H3: There is a positive relationship between the characteristics of the project and development platform being used,

H4: There is no relationship between the development platform being used and the decision to use software development methodology,

H5: There is a positive relationship between the characteristics of the project being undertaken and the decision to use software development methodology,

H6: There is a positive correlation between process quality of software producing organization and the decision to use software development methodology

H7: There is a positive correlation of the customer characteristics the software is being developed for and the decision to use software development methodology

2.7 Summary of Literature Review.

This chapter reviewed a range of literature related to software development methodology, which formed a good basis for developing the conceptual framework for the study. This will form a good basis for establishing a suitable research methodology for the research. The summary of the reviewed sources of literature is illustrated in the table below.

Summary of Literature Review

Variable	Measures	Supporting Literature
Perceived ease of use (PEU)	Learning Curve Level of control Understandable Flexibility	(Davis, 1989); (Chen, Shing-Han & Chien-Yi, 2011)
Perceived usefulness (PU)	Maintainability Productivity Performance Effectiveness	(Davis, 1989); (Chen, Shing-Han & Chien-Yi, 2011)
Project Characteristics	• Project type • Requirement clarity • Development time	(Vijayasarathy & Butler, 2016; Ozturk, 2016) (Geambaşu *et al*., 2011)
Development platform	• Development tools • programming language	(Bhattacharya & Neamtiu, 2011; Griffin, 2008b)
Customer Characteristics	Sector Customer collaboration Customer commitment	(Hassan, 2000); (Griffin, 2008a); (Ayalew & Motlhala, 2014) (Griffin, & Brandyberry, 2010)
Process quality	Project control Communication with the users	(Khalifa & Verner, 2000); (Geambaşu *et al*., 2011); (Ayalew & Motlhala, 2014)
Software Development Methodology usage (SDMU)	Breath of use Depth of use	(Hassan, 2000); (Griffin & Brandyberry, 2010); (Ayalew & Motlhala, 2014)

Table 2. 1 Literature review summary

CHAPTER 3: RESEARCH METHODOLOGY

3.1. Introduction

This chapter details the research methodology that was adopted to achieve the research objectives. This involves the research design, the population and sample of study, and data collection instruments, as well as data analysis techniques.

3.2. Research Design

The study used a descriptive survey research design in collecting data from the respondents. Descriptive survey design portrays an accurate profile of persons, events, or account of the characteristics of a individual, situation or a group (Burns & Grove, 2003). The descriptive survey research design was preferred because it ensures complete description of the situation, thus making sure that there is minimum bias in the collection of data (Kothari, 2003).

3.3. Sample Population and Size

An appropriate sampling procedure does enhance the ability of a study to represent the whole targeted population under investigation, which consequently improves the accuracy and the ability of the research to establish a meaningful view of the specific population (Levy & Lemeshow, 2013). Organizations undertaking Software development in Nairobi, Kenya were considered as the target population for this study. This population is characterized by firms of different sizes and developing software for a wide variety of industries. There are approximately four hundred and fifty-five software producing firms within the Nairobi City. The samples were selected from practitioners in software development firms who are involved in software development processes.

3.4 Sampling Technique

According to Orodho & Kombo (2002), a sampling technique involves the procedure a researcher uses to gather people, places or things to being studied. Kothari (2003) defines sampling as the selection of parts of an aggregate or totality, on whose basis a judgment or inference about the aggregate or totality is made. It is the process of obtaining information about an entire population by examining only a part of it. Cooper and Schindler (2011), argue that sampling is commonly used in inferential statistics to make predictions on the behavior of the

population. Using sampling techniques, a researcher is guaranteed that the characteristics of the population are accurately reproduced in the sample (Oso & Onen, 2011). The study adopted a simple random sampling technique to obtain the final sample size. This methodology was preferred because the target population was homogeneous. The research picked a sample size of two hundred and three respondents, taken from the population of four hundred and fifty five software companies. This consisted of the practitioners working at the software firms in Nairobi City, Kenya.

Using Yamane Formula:

$$n = \frac{N}{(1 + N(e)^2)}$$

where n is your Yamane sample size, N is underlying population size and e is determined from the confidence n therefore,

$$= \frac{425}{1 + 425(0.05)^2}$$

$$= \frac{425}{2.14}$$

$$= \qquad n = 198.6 \cong 199$$

3.5. Data collection

3.5.1. Data Sources

Data source describes where the data is driven and gathered from. This research adopted the primary source of data, which involved collection of data through a first-hand approach. Primary data collection requires the researcher to be in direct contact with the research participants. Normally, in primary-based sources of data, such data is collected in its original state, and provides the researcher the platform to manipulate the data as desired, hence permitting the research to meet the intended research purpose (Seale, 2004; Bryman & Bell, 2011).

3.5.2. Research Instruments

The researcher developed structured questionnaires in order to have control and manipulate the research towards the intended purpose. In doing so, the questionnaire used closed-ended

questions, which consisted of a list of predetermined answers from which participants choose from. Online questionnaires were also used since by using them, it was not necessary to undertake physical visits to the firms, by instead dispatching to participants via emails.

3.5.3. Data collection procedures

The study used structured questionnaires as the main instruments for collecting primary data from respondents. McMillan and Schumacher (2001) define a questionnaire as a set of questions or statements that assess attitudes, opinions, beliefs, biographical information or other forms of information. The questionnaires were used because they allowed the respondents to give their responses in a free/more convenient environment, and help the researcher get information that would not have been given out had interviews been used.

3.6. Operationalization of Variables

Variable	Measures	Related Question	Supporting Literature
Software Development Methodology usage (SDMU)	Breath of use Depth of use	Q. 4-6	(Khalifa & Verner, 2000)
Perceived ease of use (PEU)	Learning Curve Level of control Understandable	Q7-11	(Davis, 1989); (Chen, Shing-Han & Chien-Yi, 2011)
Perceived usefulness (PU)	Maintainability Development effort Productivity Performance	Q12-17	(Davis, 1989); (Chen, Shing-Han & Chien-Yi, 2011)
Project Characteristics (PC)	• Type of Project • Project Size • Requirement clarity • Development time	Q18-22	(Vijayasarathy & Butler, 2016)(Ozturk, 2016) (Geambaşu et al., 2011)

Development platform	• Programming language • Tools		(Griffin, 2008a) (Bhattacharya & Neamtiu, 2011)
Customer Characteristics (CC)	Sector Customer collaboration Customer commitment	Q23,3	(Griffin, 2008a)
Process quality (PQ)	Project control Communication with the users	Q24-25	(Khalifa & Verner, 2000)(Geambaşu et al., 2011)
Software Development Methodology usage (SDMU)	Breath of use Depth of use	Q. 4-6	(Khalifa & Verner, 2000)

Table 3 1 Operationalization of Variables

3.5. Reliability and validity of Research instrument

The questionnaire was pre-tested to discover any possible problems related to the design of the questionnaire, in terms of the degree of reliability and validity. In statistical terms, reliability is the ability of an instrument to measure something consistently and repeatedly, while validity can be described as the extent to accuracy of the research findings in reflecting the phenomena being studied (Munro, 2005).

3.6. Pilot testing

A pilot testing was performed, prior to rolling out the survey to the targeted population, in order to establish if the format of the online survey was user-friendly, if the instructions and questions were clear and understandable. This was necessary in order to ensure optimal response rate by anticipating and reducing potential issues early enough. The test samples were presented to project managers and developers from few software companies. The questionnaires were sent to their emails, along with instructions. Their feedback was incorporated in the survey instruments before actual roll out.

3.9 Data Analysis

This describes the criteria through which the data gathered from the questionnaire was organized, assessed, and interpreted. The Data collected was be edited, formatted and organized for coding into the Statistical Package for Social Sciences (SPSS) data viable table. According to Cohen *et al.* (2013), missing data may increase the risk of bias and minimize generalizability of the results. Therefore, data entered in SPSS was verified and missing data was deleted. Assumptions underlying the multivariate analysis was conducted using descriptive and structured equation Modelling. The statistical parameters generated from the software were presented in tables and charts for easier interpretation. Based on the statistical findings the researcher was able to draw useful conclusions from the responses.

3.10 **Model Estimation and Hypotheses Testing**

This is study utilized the Partial Least Squares Structured Equation Modelling (PLS SEM) as the main technique to examine the link between the independent and dependent variables. As expressed by Avkiran & Ringle (2018), PLS SEM is a non-parametric, multivariate technique that is based on the iterative OLS regression for estimating models that have latent variables and their relationships. The latent elements can be directly viewed, but can be indirectly measured via various variables, such as measuring the quality of various indicators that can be observed from questionnaire responses (Avkiran & Ringle, 2018). The study decided to use this model for various reasons. Studies suggests that the PLS has been used due to its advantages that it has over other models, such as the structural equation model (Chapman & Kihn, 2009; Lee *et al*., 2011). First, as established by Ringle, Sarstedt & Straub (2012) and Haier *et al.* (2012), this model is increasingly being used in the marketing, as well as management information systems areas of study. Additionally, according to Hair Jr *et al*.'s (2014) review, the top three reasons for the application of PLS-SEM are sample sire, data distribution, and ability to use formative indicators. For instance, Smith and Langfield-Smith (2004) argue that the PLS model is especially useful in cases where the sample size is limited, and can also accommodate non-normal data, based on the less rigorous nature of the assumptions in the technique. As explained by (LEI & LOM), PSL-SEM ability to accommodate non-normal data springs from the fact that in most cases, data gathered from social sciences studies tend to fail in adopting a multivariate normal distribution. For instance, while trying to assess path models based on other models (such as the CB-SEM), non-normal data may result into standard errors that are underestimated, and

goodness-of-fit measures that are inflated, but PLS-SEM tends to be less stringent when using non-normative data since it transforms the non-normal data according to the central limit theorem (Hair Jr *et al*., 2014). However, the main shortcoming for PLS-SEM offering wholesome results when using non-normal data is the fact that highly skewed data may lower the statistical capacity of the analysis. Therefore, based on this assessment, the study adopted the PLS model to test the hypothesis.

While using the PLS-SEM to test the hypothesis, the research undertook a multi-staged process that included the specification of the outer and inner models, gathering of data and its examination, the real model estimation, and lastly the assessment of the outcomes.

3.11 Ethical Considerations

As expressed earlier, participants tend to participant more in a research that offers them more assurance that their privacy is protected. Therefore, the research made ethical consideration to enhance participants' confidence that the information collected was to be utilized "appropriately". In doing so, the research indicated through the consent letter that was issued to participants, highlighting the purpose of the study, how the data was to be utilized, and the ethical consideration to be applied accordingly (Oliver, 2010). Therefore, it was important indicate clearly and honestly, via the consent form to participant, the purpose, and that the information collected for the research was to be utilized strictly for the study, not for any other purpose. In this sense, participants made informed decisions to freely (at will) engaged in the research process.

Additionally, to enhance privacy, the consent form highlighted the research intention to maintain anonymity while utilizing the data collected. Finally, since the research was conducted across multiple organizations, whose policies seem to differ, it depended on the chosen participant's consent if a research permission letter was explicitly required from the organization and would be acquired on which basis.

CHAPTER 4: FINDINGS AND DISCUSSIONS

4.1 Introduction

The chapter represents the findings of the research, which are based on the responses collected from the questionnaires. The findings serve as the basis upon which the research objective will be met, while also answering the research questions. Additionally, the chapter also involves the analysis of the research findings from which the research will discuss and draw conclusions that will form a suitable basis for forming recommendations on software methodology development practices in Kenya, and other similar contexts. The chapter also highlights the limitations of the tests conducted.

4.2 Response Rate

Data was collected through anonymous online survey. Out of 350 sent request via emails, 203 responded. This represented a response rate of 58%, which is considered to be an adequate sample size as established earlier.

4.3 Descriptive statistics

4.3.1 Size of the organization

The size of the organisation was measured by number of teams in the organisation. The findings (as summerized in table 4.1) indicates that 35.0 % of the organizations are micro (with employees ranging from 1 to 10). Additionally, the results indicates that firms with employees ranging from 11 to 50 had the largest percentage (at 42.9%), while those firms that can be classified as medium and large, represented 11.3% and 10.8% respectively of the firms. Therefore, this suggests that most of the studied firms were largely characterised as small firms. This findings agrees with literature on software development firms in the world, which suggests that a majority of them are small firms , with the percentage of such firms being higher in developing countries (Hassan, 2000; Ayalew & Motlhala, 2014), similar to Kenya in this context.

		Frequency	Percent	Valid Percent (%)	Cumulative Percent
Valid	1-10 employees (Micro)	71	35.0	35.0	35.0
	11-50 employees (Small)	87	42.9	42.9	77.8
	51-250 eemployees (Medium)	23	11.3	11.3	89.2
	251-2000 employees (Large	22	10.8	10.8	100.0
	Total	203	100.0	100.0	

Table 4. 1 Size of the organisation

4.3.2 Role of the respondents

According to the findings, when asked their roles in the software developing firms, most of the participants indicated that they were developers and project managers (accounting for 27.1% and 22.2% respectively), followed by analysts (at 15.8%), quality assurance managers (at 14.3%), and software architects (at 14.3%), and others (as illustrated in table 4.2). This agrees with the findings from other research in SMD, which indicates that most project managers seemed to use SMD (Griffin & Brandyberry, 2010).

Role of the respondent		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Project Manager	45	22.2	22.2	22.2
	Project Sponsor	3	1.5	1.5	23.6
	Analyst	32	15.8	15.8	39.4
	Developer	55	27.1	27.1	66.5
	Quality Assurance Manager	29	14.3	14.3	80.8
	Software Tester	6	3.0	3.0	83.7
	Software Architect	29	14.3	14.3	98.0
	Other roles(Technical lead, functional consultant among others)	4	2.0	2.0	100.0
	Total	203	100.0	100.0	

Table 4. 2 Primary role in the organisation

4.3.3 Target customers of the software product

		Responses		Percent of Cases
		N	Percent	
Customer Sector[a]	Banking and Financial Services Institutions	130	43.3%	64.0%
	Manufacturing	14	4.7%	6.9%
	Logistics	22	7.3%	10.8%
	Government	31	10.3%	15.3%
	Hospitality	21	7.0%	10.3%
	Internal use	29	9.7%	14.3%
	Media	10	3.3%	4.9%
	Others	43	14.3%	21.2%
Total		300	100.0%	147.8%

Table 4. 3 Customer target

In order to determine the users of the software "products", paarticipants were asked to indicate which sector their target market falled under. The findings from the survey revealed that 43.3% of the target customers for the software products were from the Banking and Financial services industry, manufacturing (at 4.7 %), logistics (at 7.3 %), Government (at 10.3%), hospitality (at 7.0%) , internal use (at 9.7%), while media and other sectors (such as NGOs) recorded 3.3% and 14.3 % respectively (as illustrated in table 4.3). The findings agrees with similar studies in other developing countries (such as Botswana), where main target customers for the SDM firms operate in the financial & insurance, public enterprises, agriculture, retail, and mininng (Ayalew & Motlhala, 2014). According to Hassan (2000), the target markets, along with their dynamics, serve as one of the critical external factors that influence how SDM firms evolve. Therefore, this suggests that the dynamics of the above list of target sectors tend to offer opportunities for SMD firm; with sectors such as NGOs leading and the government. Similarly, Ayalew & Motlhala (2014) indicates that almost all of firms pursue business with the government, a part from those that focus on very small firms. According to Hassan (2000), this may include the need for new software products, and shifts in specific customers' needs that may require SMD firms to solve in these sectors.

4.2.4 The characteristics of the project (s) undertaken by the organizations

Company Size * Type of project most common in your company Crosstabulation

			Type of project most common in your company					
			Commercial Off-the-Shelf software customization	New software development	System integration	Quality assurance	Software upgrade	Total
Company Size	1-10 employees (Micro)	Count	0	53	10	8	0	71
		% of Total	0.0%	26.1%	4.9%	3.9%	0.0%	35.0%
	11-50 employees (Small)	Count	43	13	31	0	0	87
		% of Total	21.2%	6.4%	15.3%	0.0%	0.0%	42.9%
	51-250 Employees (Medium)	Count	0	14	0	9	0	23
		% of Total	0.0%	6.9%	0.0%	4.4%	0.0%	11.3%
	251-2000 employees (Large	Count	0	0	15	0	7	22
		% of Total	0.0%	0.0%	7.4%	0.0%	3.4%	10.8%
Total		Count	43	80	56	17	7	203
		% of Total	21.2%	39.4%	27.6%	8.4%	3.4%	100.0%

Table 4. 4 Type of Projects

Participants were asked to identify the most common type of project that their organizations engaged in. The findings indicated that 39.4% of the firms engaged in new software development, 27.6% engaged in system integration, 21.2% engaged in commercial-off-the-shelf software customization, 3.4% engaged in system upgrade, and 8.4% engaged in projects related to quality assurance projects (as illustrated in table 4.4). These aspects tend to relate to rapid shifts in technologies and dynamics in market, which according to Hassan's (2000) study on

software evolution in the context of developing countries, differentiates the software industry with the rest. According to Ayalew & Motlhala (2014), small software firms normally participate in development and maintenance of software products, which is used by larger systems. This may include the type of projects that Kenyan software firms also engage in (as expressed in table 4.4).

4.2.5 Average Project Duration

Company Size * What is the average Project Duration Crosstabulation

			What is the average Project Duration					
			Less than 3 month	3 to 6 months	6 to 12 months	12 to 18 months	18 to 24 months	Total
Company Size	1-10 employees (Micro)	Count	10	47	10	0	4	71
		% of Total	4.9%	23.2%	4.9%	0.0%	2.0%	35.0%
	11-50 employees (Small)	Count	0	70	17	0	0	87
		% of Total	0.0%	34.5%	8.4%	0.0%	0.0%	42.9%
	51-250Employees (Medium)	Count	0	11	0	1	11	23
		% of Total	0.0%	5.4%	0.0%	0.5%	5.4%	11.3%
	251-2000 employees (Large	Count	17	0	0	0	5	22
		% of Total	8.4%	0.0%	0.0%	0.0%	2.5%	10.8%
Total		Count	27	128	27	1	20	203
		% of Total	13.3%	63.1%	13.3%	0.5%	9.9%	100.0%

Table 4. 5 Average duration of the projects

A crosstabulation of the firms' size and projects duration was done to determine the average duration of the projects partaken by the firms (As indicated in table 4.4). The findings indicates that, generally, projects took short time to be completed; whereby 63.1% of the respondents indicated that their firms took between 3 to 6 months to complete the project, while the lowest percentage (at 0.5%) of the projects took 12-18 months to complete. This expresses the value attached to timely deliverance of software products by software firms, even in the context of developing countries. For instance, as expressed by Ayalew., & Motlhala (2014), software firms plays a great role in the economy, but for them to be effective, they need to develop quality products on a timely basis, and within budgets, since failure to do so poses the risk of losing customers.

4.2.6 Programming Language used

Programming Language used

		Responses		Percent of Cases
		N	Percent	
Programming language used[a]	JAVA	148	33.6%	72.9%
	C++	22	5.0%	10.8%
	Python	37	8.4%	18.2%
	C#	53	12.0%	26.1%
	PHP	95	21.5%	46.8%
	Other Programming language(SCALA, Android)	86	19.5%	42.4%
Total		348	441	100.0%

Table 4. 6 programming language used

Respondents were asked to identify the programming languages used in their organization. As indicated in table 4.5., 33.6 % of the respondents revealed that their firm used Java programming language, followed by PHP (accounting for 21.5%), while other programming languages (including Android and Scala) recorded19.5%, and C++, python, C# reported 4.0%, 8.5% and

12.0% respectively. Therefore, this suggests that Java programming language was the most commonly used language among software developing firms in Kenya

4.2.7 Standard process used

Company Size * Standard Process Crosstabulation

			Standard Process			
			Yes, all projects are executed as per standard process	Each project individually select process to follow	No particular process exists	Total
Company Size	1-10 employees (Micro)	Count	28	33	10	71
		% of Total	13.8%	16.3%	4.9%	35.0%
	11-50 employees (Small)	Count	24	63	0	87
		% of Total	11.8%	31.0%	0.0%	42.9%
	51-250 Employees (Medium)	Count	23	0	0	23
		% of Total	11.3%	0.0%	0.0%	11.3%
	251-2000 employees (Large	Count	22	0	0	22
		% of Total	10.8%	0.0%	0.0%	10.8%
Total		Count	97	96	10	203
		% of Total	47.8%	47.3%	4.9%	100.0%

Table 4. 7 standard process

Participants were asked to specify if their organizations follow software development standards during software engineering process. The findings indicated interesting information (as illustrated in table 4.6). From the findings, 5.7% of organizations, with employees ranging from 1-10, do not follow any process. However, over 94% of the respondents reported that they use a standard process in their software development activities. This matches with literature that

suggests that firms normally employ software process standards and undergo certifications according to the capability of their process in order to enhance the ability to produce quality software (Ayalew & Motlhala, 2014). However, according to Ayalew & Motlhala (2014), this is characterised by challenges of developing countries' software firms in relating international standards to their business needs and justify the use of such standards in their organisations.

4.2.8 Analysis of approaches employed on various Software development Project-related activities

In order to determine the most common approach adopted by firms in their software development project-related activities, the research surveyed SWEBOK related practices by asking participants to specify the approach adopted by their firms among the traditional, balanced between traditional and agile, and Agile approach. The results showed that 49.7% of firms prefer to run software development activities in an agile manner, 31.8% prefer a hybrid model (comprising of a mix balance between traditional and agile approach), while 18.5% preferred to use the traditional approaches (as illustrated in table 4.6).

The survey went further to assess the approach employed per each SWEBOK practices. The findings indicated that the traditional approach was largely used for Risk management-related activities (at 38.9%) and Maintenance and Evolution-related activities (at 31%), while least used in integration and testing (at 0%). Secondly, the survey also indicated that the balanced approach between the traditional and agile approach was largely used for integration and testing activities (at 49.8%), followed slightly close by transition and operation activities (at 49.3%), while least used in architecture and design activities and maintenance and evolution-related activities (both at 12.8%)/ Lastly, the survey indicated that the agile approach was largely used in architecture and design-related activities (at 70.4%) , followed by implementation/coding-related activities (63.1%), while lowest used in transition and operation-related activities (at 34.5%)(as illustrated in table 4.7). In general, the survey findings also indicated that the agile approach was the most used in Kenya, recording a significantly high percentage across the activities (as illustrated in table 4.7). The dominant use of the agile approach agrees with existing literature, which suggests that increased use of the agile approach due to the attached advantages (Griffin &

Brandyberry, 2010). According to Reifer (2002), such advantages include lowered costs, shorter durations-to-market, enhanced productivity, and enhanced quality.

Software Development Approaches Frequencies

		Responses		Percent of Cases
		N	Percent	
Approaches Used	Traditional Manner	413	18.5%	203.4%
	Balanced between traditional and Agile	711	31.8%	350.2%
	Agile Manner	1109	49.7%	546.3%
Total		2233	100.0%	1100.0%

Table 4. 8 Summary of Software development approaches adopted by firms

The survey also seeks to establish the framework used by firms in their software development project activities. From the findings, the highest percentage of firms that use the classic waterfall method seem to use it sometimes (at 40.4%); with 22.7% indicating that they never used it. Secondly, the highest percentage of firms that use the extreme programming framework seemed to suggest that they rarely use it (at 40.9%), with 13.3% of the respondents indicating they do not know the framework, 19.7% do not know if they use it, 3.4% and never use it. Thirdly, the highest percentage of firms that use the SCRUM framework often use it (at 34.5%), followed closely by 33.5% that indicated that they use it sometimes. Fourthly, the highest percentage of firms that use the Rational Unified Process seem to do so sometimes (at 30.5%); with 13.3% of the respondent indicating that they do not know about it and 9.4% never use it. Lastly, the highest percentage of firms that used the Spiral framework tends to do so rarely (at 54.7%); with 5.9% of the respondents suggesting that they do not know about it (as illustrated in table 4.8). Therefore, in general, the SCRUM framework tends to be largely often used in Kenya, relative to the rest of the framework that were surveyed. This agrees with Griffin & Brandyberry's (2010) study, which indicates an increasing trend in the usage of SCRUM as one of the agile methodologies.

	Traditional Manner		Balanced between traditional and Agile		Agile Manner	
Practice	**Count**	**%**	**Count**	**%**	**Count**	**%**
Quality Management	10	4.90%	81	39.90%	112	55.20 %
Risk Management	79	38.90%	29	14.30%	95	46.80 %
Configuration Management	41	20.20%	75	36.90%	87	42.90 %
Change Management	48	23.60%	72	35.50%	83	40.90 %
Requirements Analysis/Engineering	26	12.80%	75	36.90%	102	50.20 %
Architecture and Design	34	16.70%	26	12.80%	143	70.40 %
Implementation/Coding	19	9.40%	56	27.60%	128	63.10 %
Integration and Testing	0	0.00%	101	49.80%	102	50.20 %
Transition and Operation	33	16.30%	100	49.30%	70	34.50 %
Maintenance and Evolution	63	31.00%	26	12.80%	114	56.20 %

Table 4. 9 Approaches employed per each SWEBOK practice

Methodology	Do not know the framework		Do not know if we use it		We never use it		We rarely use it		We sometimes use it		We often use it		We always use the framework	
	Count	%	Count	%	Count	%	Count	%	Count	%	Count	%	Count	%
Classic waterfall Method	0	0.00%	0	0.00%	46	22.70%	28	13.80%	82	40.40%	38	18.70%	9	4.40%
Extreme Programming	27	13.30%	40	19.70%	7	3.40%	83	40.90%	36	17.70%	10	4.90%	0	0.00%
SCRUM	0	0.00%	0	0.00%	0	0.00%	10	4.90%	68	33.50%	70	34.50%	55	27.10%
Rational Unified Process	27	13.30%	19	9.40%	31	15.30%	46	22.70%	62	30.50%	18	8.90%	0	0.00%
Spiral Model	12	5.90%	0	0.00%	31	15.30%	111	54.70%	40	19.70%	9	4.40%	0	0.00%

Table 4. 10: Methodologies used

4.2. 9 Perceived Benefits frequencies

Respondents were asked to specify to what extent they agree with statements related to their perceived usefulness and perceived ease of use of software development methodologies. In doing so, the highest percentage strongly agreed (at 40.9%) that SMD give them greater control, followed by 36.9% who strongly agreed that SMD provides helpful guidance. Secondly, a majority of the respondents agreed that the SMD used was flexible (at 66%), followed closely by 64% who agreed that SMD improves job performance, and that SMD used enhances project effectiveness (at 61%). On the other hand, a significant number of respondents disagreed with the statement that SMD used was flexible, SMD used was easy to learn, and that the used SMD improves job performance (all at 13.3%)(as illustrated in Table 4.9).This suggests that Kenyan software firms highly prefer to use SMD based on the perceived benefit that it offers greater control and ability to offer helpful guidance. This can be linked to the perceived benefit as expressed by Geambaşu *et al.* (2011), in the study of the factors that influence the choice of a

SDM. First, greater control can be linked to software criticality, described by Geambaşu *et al.* (2011) it include the ability to offer tracking capability for proper running of the development process, through managing risks, starting from the early stages of the project and sufficient ability to plan and monitor the iteration, so as to attain the objectives of projects. On the other hand, the ability to offer helpful guidance can be linked to what Geambaşu *et al.* 's (2011) study regards to as clarity of the original requirements, which entails having accurate and complete description of the requirements from the start of the projects, and integrating all the required client's functionality.

	Strongly Agree		Agree		Neutral		Disagree		Strongly Disagree	
	Count	%	Count	%	Count	%	Count	%	Count	%
SDM provides Helpful guidance	75	36.90%	128	63.10%	0	0.00%	0	0.00%	0	0.00%
SDM gives greater Control	83	40.90%	103	50.70%	17	8.40%	0	0.00%	0	0.00%
SDM used is understandable	69	34.00%	116	57.10%	18	8.90%	0	0.00%	0	0.00%
SDM used is Flexible	10	4.90%	134	66.00%	1	0.50%	31	15.30%	27	13.30%
SDM used is Easy to learn	0	0.00%	110	54.20%	38	18.70%	37	18.20%	18	8.90%
SDM used enables us to work more Quickly	47	23.20%	94	46.30%	14	6.90%	21	10.30%	27	13.30%

used SDM Improves Job Performance	20	9.90 %	130	64.0 0%	18	8.90%	8	3.90%	27	13.30 %
SDM used increase Productivity	68	33.5 0%	112	55.2 0%	4	2.00%	19	9.40%	0	0.00%
SDM used enhances project Effectiveness	74	36.5 0%	116	57.1 0%	0	0.00%	1	0.50%	12	5.90%
SDM used makes Job easier	41	20.2 0%	125	61.6 0%	18	8.90%	19	9.40%	0	0.00%
SDM used simplify software Maintainability	40	19.7 0%	80	39.4 0%	24	11.80%	47	23.20 %	12	5.90%

Table 4. 11 Perceived benefits on software development methodologies frequencies

4.3 Hypothesis testing

4.3.1 SmartPLS Bootstrapping Direct Effect Results

We utilize a variance based Structured Equation Modelling method known as Partial Least Squares Structured Equation Modelling (PLS-SEM) and SmartPLS 3.0 statistical software (Ringle, Wende, & Becker, 2015) to test the hypotheses. PLS-SEM is considered in management research as one of the useful tool for developing and extending existing theories (Hair et al., 2014)

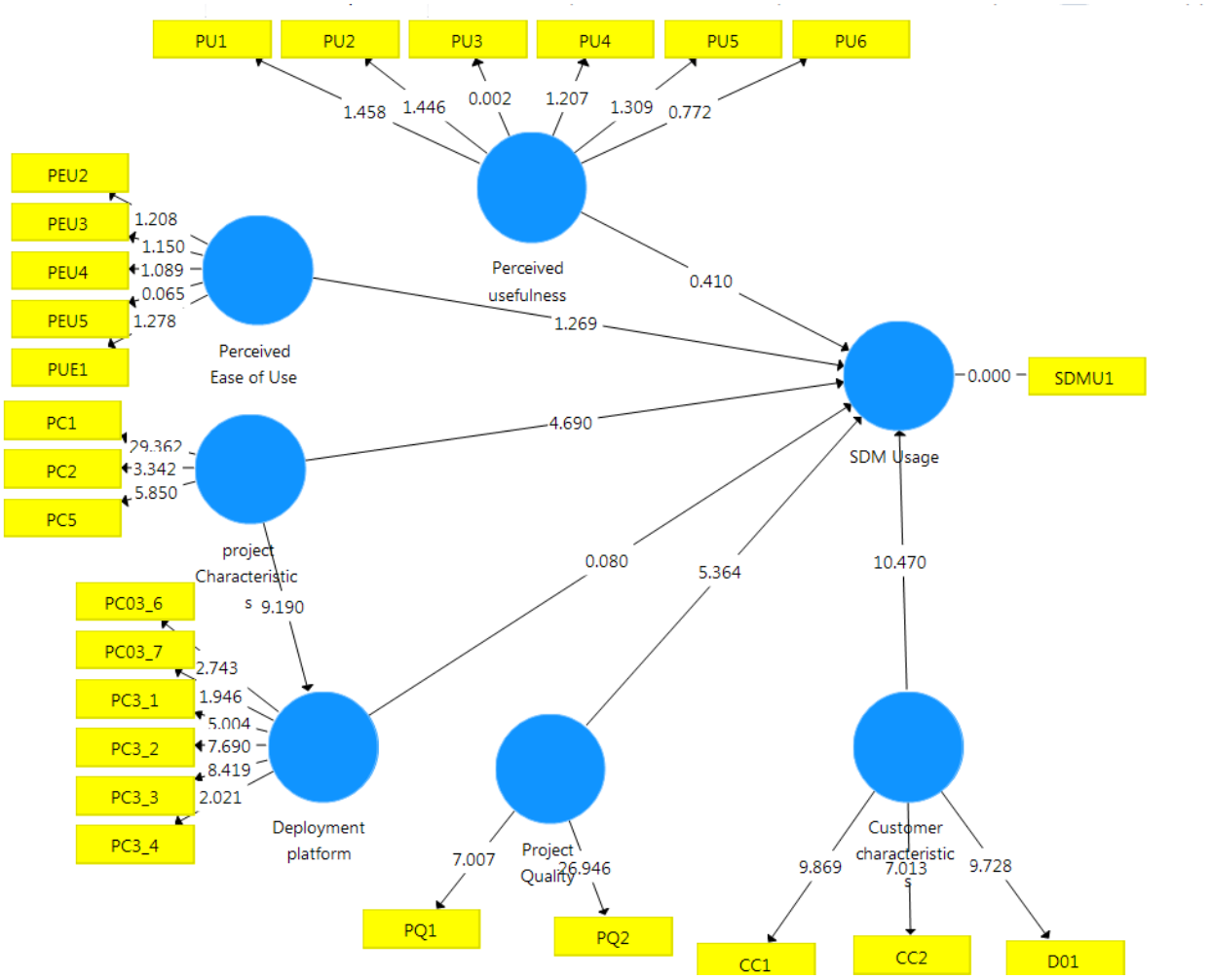


Figure 4. 1: SDMU Research Model and PLS-SEM results

Path coefficients estimates and R values for the model are shown in Figure 4.10 above. The path coefficients illustrates the strength of the relationships between the dependent variable software development methodology usage and the independent latent variables Perceived ease of use, perceived usefulness, project characteristics, and customer characteristics and project quality.

4.3.2 Hypothesis testing summary

Hypothesis	Relationship	Std Beta	Std Error	|t-value|^	Decision	P Values	95% CI LL	95% CI UP
H1	Perceived Ease of Use -> SDM Usage	0.142	0.201	1.269	Non Supported	0.205	-0.268	0.342
H2	Perceived usefulness -> SDM Usage	-0.036	0.089	0.410	Non Supported	0.682	-0.170	0.119
H3	project Characteristics -> Deployment platform	0.539	0.057	9.19**	Supported	0.000	0.482	0.598
H4	Deployment platform -> SDM Usage	0.005	0.084	0.080	Non Supported	0.937	-0.121	0.157
H5	Project Quality -> SDM Usage	0.575	0.111	5.36**	Supported	0.000	0.406	0.764
H6	Customer characteristics -> SDM Usage	-0.750	0.072	10.47**	Supported	0.000	-0.871	-0.648
H7	project Characteristics -> SDM Usage	0.530	0.118	4.69**	Supported	0.000	0.350	0.727

Table 4. 12 Hypothesis testing summary

The hypothesis were tested by determining the relationship between variables; whereby a strong relationship was determined in a case where the P-Value was less than 0.01

H_0: Software producing companies in Nairobi use some form of software development methodology,

H1: Individuals Perceived ease of use has a positive correlation on methodology usage,

As derived from the results, H1; P-Value= 0.205, with t-value= 1.269 suggested that the relationship is insignificant, thus H1 is rejected.

H2: Individuals Perceived usefulness has a positive correlation with methodology usage,

As derived from the results; H2 P-Value=-0.682, with t-value= 0.410 these results shows that the effect is insignificant, and thus we reject the hypothesis that perceived usefulness have significant relationship with methodology usage.

H3: There is a positive relationship between the characteristics of the project and development platform being used,

As derived from the results, H3; P-Value= 0.000, with t-value =9.19, which is significant hence the hypothesis holds.

This agrees with Geambaşu *et al.* (2011), who argues that each project owns its specific characteristics, which needs to be considered when making the choice of the software methodology to be applied in software development. This would offer a good explanation on the variations in the feedback given by the respondents during the survey, which would suggests that they there project had different characteristics, which informed the variation in their choice of software methodology used, even though a majority of the respondents indicated that they used the agile methodology, which most of them indicating to often use the SCRUM framework, as a subset of the agile methodology.

H4: There is no relationship between the development platform being used and the decision to use software development methodology,

As derived from the results, H4; P-Value= 0.937, with t-value =0.080, which is insignificant, and hence we reject H4.

H5: There is a positive relationship between the characteristics of the project being undertaken and the decision to use software development methodology,

As derived from the results, H5; P-Value= 0.000, with t-value =4.69, which is significant, and hence H5 is supported.

This finding tends to agree with Griffin & Brandyberr (2010), who explain that many SDM have been developed over the years, especially in the attempt to enhance processes and developers acceptance. For instance, Griffin & Brandyberr (2010) argue that agile methodologies (such as SCRUM and Extreme) were developed to resolve elements of development process that might be inadequate or lack in other methodologies, or which specific forms of development environment or application may require. Therefore, the vary characteristics of the project undertaken by Kenyan SD firms acts as the predictor of the chosen SDM to be used in the project. Therefore, the common choice of Java application, and the agile (and often SCRAM framework) seem to be the favourable choice of SDM that fulfils the requirements and needs of SD in the Kenyan market.

H6: There is a positive correlation between process quality of software producing organization and the decision to use software development methodology

As derived from the results, H 6: P-Value= 0.000, with t-value =5.36, which is significant, and hence the hypothesis holds.

For instance, as expressed by Griffin & Brandyberry (2010), the CMM model focussed on process of higher maturity, thus leading to higher productivity, lowered cycle time, and lesser defects. According to Griffin & Brandyberry (2010), such model could be adopted with organisations that stress on inspections and peer review as a focus of ensuring quality in their processes. On the other hand, Ayalew & Motlhala (2014) explains how ISO/IEC 15504 has been customised to fit the changing needs of small SD firms. Therefore, in this research context, the agile, specifically SCRUM methodology seemed to offer the best option to satisfy the process quality of most Kenyan SD firms.

H7: There is a positive correlation of the customer characteristics the software is being developed for and the decision to use software development methodology

As derived from the results, H7; P-Value= 0.000, with t-value =10.47, which is significant, and hence H7 holds.

As expressed by Griffin & Brandyberry (2010), the need to incorporate the end user into the SDM process was informed by the increased effort by software firms to try and enhance the

chances of meeting the user's needs. Where methodologies (such as prototyping) make efforts to incorporate user's input in their design process, while other methodologies strive to strategically incorporate user's input in specific stages of the design process (Griffin & Brandyberry, 2010). Additionally, Hassan (2000) stresses on the value of product quality and customer trust, since the degree of trust between the customer and the firm, has a significant influence on the success of the SDM project. Therefore, since NGOs and the government (rather public enterprise) seem to be the most common target customer for Kenyan SD firms, their characteristics are the main predictor on which SDM the SD firms adopts.

Reduced Model

Research results shows out of the six constructs considered to affect usage of the software methodology only three (Customer Characteristics, project characterictics , Project quality)are found to be significant.

Revised model based on the survey is depicted in figure 4.2 below.

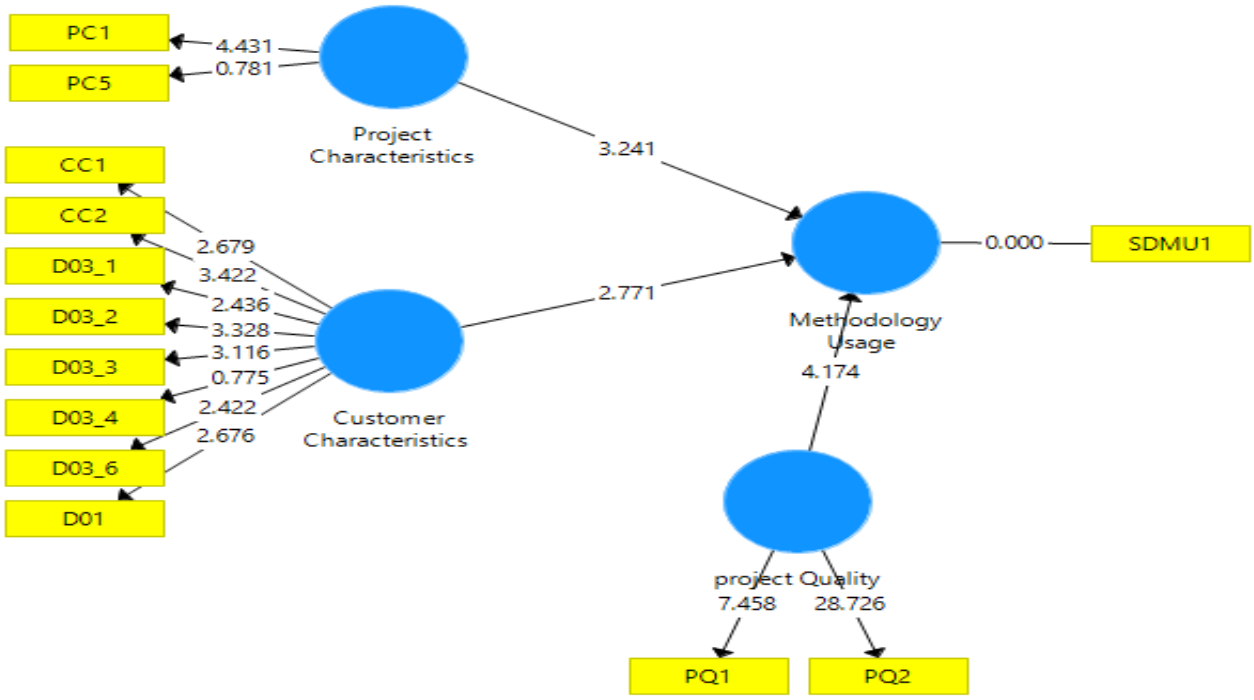


Figure 4.2 Reduced model

CHAPTER 5: CONCLUSION AND RECOMMENDATIONS

5.0 Overview

Here we summarize key results from this research and relevancy to software development methodology usage. Following section details the research objectives and how this research addresses the questions therein.

Linking Research objectives to the findings

The study was aimed at investigating software development methodologies in practice and factors influencing usage in software development firms in Nairobi. We summarize how research objectives were realized and the research questions addressed.

Objective 1: Identify software development methodologies used in Kenya

The study found out that variously software development methodologies are practiced and in different manner this includes; Traditional manner, hybrid of traditional and Agile, and pure agile manner. The findings indicate Agile manner is the most adopted manner.

Objective 2: Get information on what software development organization consider as benefits of using software development methodologies

Perceived usefulness and perceived ease of use we designed to predict what individual consider as key benefits of firms using software development methodology. Findings appears to indicate the constructs may not be predictors of whether individual will use a system development methodology.

Objective 3. Determine factors influencing the acceptance of software development methodology.

Various factors we considered as part of the study to determine extent to which they influence acceptance of software development methodology. These factors included; customer characteristics, project characteristics, deployment platform and project quality aspects . Apart from deployment platform which was included in the model as an intervening variable the rest of the factors were found as significant in deciding the usage of software methodology.

5.1 Conclusions

The study was aimed at investigating software development methodologies in practice and factors influencing usage in software development firms in Nairobi.In doing so, the research focused on achieving several research objectives; that is, to identify software development methodologies used in Kenya, to determine what software development organizations consider as the benefits of using software development methodologies, and to determine the factors influencing the usage of SDM in Nairobi ,Kenya. First, the first objective is fulfilled where the research indicates that majority of SD firm use standard process in their SD activities. The research also suggests that most SD firms use the Java application and agile methodology in SDM projects; whereby the research identified further that the SCRUM framework, which is a form of agile methodology framework, is the most often used by SD firms.

Secondly, to determine the factors influencing the acceptance of software development methodologies in Kenya, the research assessed the target customer of the software products, with findings suggesting that a majority of the customers are in the NGOs sector and the government, which serves as customer to SD firms. Secondly, the research established that a majority of SD firms engage in projects relating to new software development , and closely followed by system integration. This agrees with studies in developing countries that suggest that most small firms engage in new software development product production and maintenance that serves large firms.

Thirdly, in determining what the software firms perceive as the benefits of SMD, research established that Kenyan firms highly use software development methodologies due to the perceived benefits of offering greater control, and offer helpful guidance.

In addressing the objectives, the study also confirmed that the perceived ease of use will have a significant relationship with methodology usage, there is a positive relationship between the characteristics of the project and development platform uses, that there is a significant relationship between process quality of software producing organization and the decision to use software development methodology, that there is a significant relationship between the customer's characteristics of the software being developed and the decision to use software development methodology, and that there is significant relationship between the characteristics of the project being undertaken and the decision to use software methodology. In this sense, this

suggests that although most of the firm used the Java programming language in software development ; with a majority adopting the agile method, and more specifically the SCRUM framework, the slight variations existed due to the differences in the characteristics of the project being developed. Secondly, the choice of SDM is also influenced by the process quality of software producing organization, which determines what SDM would be more appropriate to satisfy the expected process quality of each of the organization. Thirdly, the characteristic of the customer plays a critical role in the decision made by Kenyan SD firms on which SDM to apply in product development; whereby in this case that would entail NGOs and the government as the common customer for Kenyan SD firms. Lastly, the choice of the dominant choice of the Java programming language, and the agile methodology describes their suitability to satisfy the nature of product requirements and customer needs for software development.

5.2 Recommendations

The findings and the conclusion from this research offer a suitable basis for SD firms to establish the most suitable software development methodology to use in developing products for the local market, and how they can enhance their competitiveness and sustainability therein. This includes using appropriate programming language in this research preferred language being Java and adopting agile methodology in their product development efforts. Additionally, the report also indicates that banking/financial institutions and the government (rather public entities) are the main target customers in the Kenyan Software development market, which offers useful insight for software development firms in the Kenyan market, and those intending to enter into the market on the need develop products that suits this customer base in order to strategically target them. However, as argued by Hassan (2000), the target market and its dynamics seem to change, and thus Software development firms need to be flexible enough to adapt to such dynamics in order to enhance their competitiveness and long-term sustainability. Lastly, the research also offers key indications on the role of the government and public entities in developing the Software development market, since it is not only the main determinant of the playing field, but also a customer for the Software development firms. This is especially useful since the findings indicate that a majority of software development firms in Nairobi, Kenya are small in size, which tends to be common aspects across the world, especially in developing countries. Therefore, as expressed in other similar contexts (Hassan, 2000; Ayalew & Motlhala, 2014) the government

can offer a favourable environment for Software development related activities to thrive, which can help the overall Software development industry in Kenya and the economy to be stimulated.

5.3 Limitations and Future study

Firstly, despite the expressed advantages related to using questionnaires, the research also had some limitations. According to Phella, Bloch & Seale (2011), questionnaires are normally linked with dishonesty, in cases where respondents tend to get tempted to offer responses that are not truthful. Such dishonesty tends to originate from the need and temptation by participants to protect their privacy. Therefore, in order to reduce the chances of such occurrences, the research made effort to reduce the use of sensitive information. Moreover, questionnaires bear the disadvantages, since participants tend to offer answers to questionnaire questions based on their understanding, which influences variations in interpretations, and also leading to irrelevant results. This informed the need to use well-structured and more closed-ended questions, coupled with clearer and simpler sentences. Additionally, the pilot testing was necessary in order to determine any potential weakness, and thus enhance the suitability of the questionnaires. All these were aimed at ensuring that the findings from the research were reliable in in answering the research questionnaires as much as possible. Secondly, number and nature of firms that participated in this study may not fully offer the true reflection of the Kenyan SDM market, since the research was undertaken in Nairobi. But, by comparing the sample size used in this study with other similar studies (Hassan, 2000; Ayalew & Motlhala, 2014), this study still stands to be reliable. More studies are necessary to understand interrelation between Scrum methodologies and Java. Future research should also consider the opening to other cities or entire country to understand adopted software development practices

6. REFERENCES

Alanezi, M. A. (2018). The Adoption of Software Process Improvement in Saudi Arabian Small and Medium Size Software Organizations: An Exploratory Study. IJACSA) International Journal of Advanced Computer Science and Applications, 9(3). Retrieved from www.ijacsa.thesai.org

Anees, A. (2017). Software process improvement models and their comparison. International Journal of Advanced Research in Computer Science, 8(5), 928-932

Apoorva, M., & Deepty, D. (2013). A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios. International Journal of Advance Research in Computer Science and Management Studies Research, 1(5), 65–79. Retrieved from http://ijarcsms.com/docs/paper/volume1/issue5/V1I5-0008.pdf%5Cnhttp://www.iosrjen.org/Papers/vol2_issue7 (part-2)/D0272124.pdf%5Cnhttp://www.ijarcsms.com/docs/paper/volume1/issue5/V1I5-0008.pdf

Ayalew, Y., & Motlhala, K. (2014). Software process practices in small software companies in botswana. Proceedings - 14th International Conference on Computational Science and Its Applications, ICCSA 2014, 49–57. https://doi.org/10.1109/ICCSA.2014.20

Bhattacharya, P., & Neamtiu, I. (2011). Assessing Programming Language Impact on Development and Maintenance : A Study on C and C ++ Categories and Subject Descriptors. (2).

Bryman, A., & Bell, E. (2011). Ethics in business research. Business Research Methods, 7(5), 23-56.

Chen, S. C., Shing-Han, L., & Chien-Yi, L. (2011). Recent related research in technology acceptance model: A literature review. Australian Journal of Business and Management Research, 1(9), 124.

Chrissis, M., Konrad, M., & Shrum, S. (2011). CMMI for Development. In Engineering.

Clarke, P., & O'connor, R. V. (2012). Towards a comprehensive reference framework. Journal of Information Software and Technology, 54(5), 433–447. Retrieved from http://doras.dcu.ie/16823/1/ClarkeAndOConnor-Vol54No5-pp433-447.pdf

Colecchia, A. (n.d.). DEFINING AND MEASURING ELECTRONIC COMMERCE. Towards the development of an OECD methodology. Towards the development of an OECD methodology. Retrieved from http://css.escwa.org.lb/SD/0977/b8.pdf

Cooper & Schindler. (2011). Business Research Methods. India: McGraw-hill.

Dasoriya, R. (2017). Significance of Software Development Models. International Journal of Advanced Research in Computer Science, 8(8), 732–736. https://doi.org/10.26483/ijarcs.v8i8.4839

Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, 13(3), 319–340.

Diéguez, M., Sepúlveda, S., & Cachero, C. . (2012). UMAM-Q: An instrument to assess the intention to use software development methodologies. Iberian Conference on Information Systems and Technologies, CISTI, 1–6. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-84869057750&partnerID=40&md5=6e007ae521c4aab682a380abf0b214d5

Dutta, S., Lee, M., & Van Wassenhove, L. (1999). Software engineering in Europe: a study of best practices. IEEE Software, 16(3), 82–90. https://doi.org/10.1109/52.765792

Garousi, V., Coşkunçay, A., Betin-Can, A., & Demirörs, O. (2015). A survey of software engineering practices in Turkey. Journal of Systems and Software, 108, 148–177. https://doi.org/10.1016/j.jss.2015.06.036

Geambaşu, C. V., Jianu, I., Jianu, I., & Gavrilă, A. (2011). Influence factors for the choice of a software development methodology. Accounting and Management Information Systems, 10(4), 479-494.

Geambaşu, C., Jianu, I., Jianu, I., & Gavrila, A. (2011). Influence Factors for the Choice of a Software Development Methodology. Accounting and Management Information Systems, 10(4), 479–494. https://doi.org/http://dx.doi.org/10.1016/S0044-328X(83)80043-9

Griffin, A. S. (2008b). EXAMINING THE DECISION PROCESS AND OUTCOMES OF SYSTEM DEVELOPMENT METHODOLOGY ADOPTION. Kent State University.

Griffin, A. S., & Brandyberry, A. A. (2010). System development methodology usage in industry: a review and analysis. Journal of Information Systems Applied Research, 3(19), 1-18.

Halit Soydan, G., & M. Kokar, M. (2012). A Partial Formalization of the CMMI-DEV—A Capability Maturity Model for Development. Journal of Software Engineering and Applications, 05(10), 777–788. https://doi.org/10.4236/jsea.2012.510090

Hassan, S. Z. (2000). Software industry evolution in a developing country: An in depth study. In Proceedings of the 33rd annual Hawaii international conference on system sciences (pp. 10-pp). IEEE.

Henseler, J., Ringle, C. M., & Sinkovics, R. R. (2009). The use of partial least squares path modeling in international marketing. In New challenges to international marketing (pp. 277–319). Emerald Group Publishing Limited.

Humphrey, W.S. (2015). Introduction to Software Process Improvement.

Jalote, P. (2008). Software Processes. In A Concise Introduction to Software Engineering (pp. 1–28). https://doi.org/10.1007/978-1-84800-302-6_2

Jiang, L., & Eberlein, A. (2009). An analysis of the history of classical software development and agile development. Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, (October), 3733–3738. https://doi.org/10.1109/ICSMC.2009.5346888

Kaur, S. (2015). International Journal of Advanced Research in Computer Science and Software Engineering A Review of Software Development Life Cycle Models. 5(11), 354–360.

Khalifa, M., & Verner, J. M. (2000). Drivers for software development method usage. IEEE Transactions on Engineering Management, 47(3), 360–369. https://doi.org/10.1109/17.865904

Kirk, D., & Tempero, E. (2012). Software development practices in New Zealand. Proceedings - Asia-Pacific Software Engineering Conference, APSEC, 1, 386–395. https://doi.org/10.1109/APSEC.2012.51

Kozina, M., & Kirinic, V. (2013). Analyzing the Pam's Structure Using the Iso/Iec 15504-5 Standard (Spice). 475–490. https://doi.org/10.2507/daaam.scibook.2013.26

Liviu DESPA, M. (2014). Comparative study on software development methodologies. Database Systems Journal, V(3), 37–56. https://doi.org/10.1109/MAHC.1983.10102

Macías-Escrivá, F. D., Haber, R., Del Toro, R., & Hernandez, V. (2013). Self-adaptive systems: A survey of current approaches, research challenges and applications. Expert Systems with Applications, 40(18), 7267-7279.

Mall, R. (2014). FUNDAMENTALS OF SOFTWARE ENGINEERING, Fourth Edition.

Marco Kuhrmann, Philipp Diebold, J. M. (2017). Hybrid Software and System Development in Practice: Waterfall, Scrum, and Beyond. CEUR Workshop Proceedings, 2226, 26–32. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Margarido, I. L., Vidal, R. M., & Vieira, M. (2012). Lessons Learnt in the Implementation of CMMI (R) Maturity Level 5. 2012 EIGHTH INTERNATIONAL CONFERENCE ON THE QUALITY OF INFORMATION AND COMMUNICATIONS TECHNOLOGY (QUATIC 2012). https://doi.org/10.1109/QUATIC.2012.37

Oliveira, T., & Martins, M. F. (2011). Literature review of information technology adoption models at firm level. Electronic Journal of Information Systems Evaluation, 14(1), 110.

Orodho, A. J. and Kombo, D. K. (2002). Research Methods. Nairobi: Kenyatta University, Institute of Open Learning.

Oso, W. Y. & Onen, D. (2011). A general guide to writing research proposal and report; Handbook for beginning researchers. Nairobi. Jomo Kenyatta Foundation Wesley Longman.

OTIENO, B. (2017, September 13). ICT hitch stalls customer services at National Bank - Business Daily. Business Daily. Retrieved from www.businessdailyafrica.com/news/ICT-hitch-stalls-customer-services-at-National-Bank/539546-4094996-oexgvg/index.html

Ozturk, V. (2016). Selection of appropriate software development life cycle using fuzzy logic. (January). https://doi.org/10.3233/IFS-120686

Peng Xu, R., & Balasubramaniam. (2007). Software Process Tailoring: An Empirical Investigation. Journal of Management Information Systems, 23(2), 149–171. https://doi.org/10.2753/MIS0742-

Phellas, C. N., Bloch, A., & Seale, C. (2011). Structured methods: interviews, questionnaires and observation. Researching society and culture, 3

Pierre Bourque, & Fairley, R. E. (Dick). (2014). Swebok V3.0. In Guide to the Software Engineering Body of Knowledge. https://doi.org/10.1234/12345678

Popli, R., Anita, & Chauhan, and N. (2012). MAPPING OF TRADITIONAL SOFTWARE DEVELOPMENT METHODS TO AGILE METHODOLOGY. 3(5), 1–15.

Putta, A. (2016). Comparative Analysis of Software Development Practices across Software Organisations. https://doi.org/10.1002/0470011815.b2a10020

Reifer, D. J. (2002). How good are agile methods?. IEEE software, 19(4), 16-18.

Riemenschneider, C. K., Hardgrave, B. C., Society, I. C., & Davis, F. D. (2002). 2002 - Riemenschneider, Hardgrave, Davis - Explaining software developer acceptance of methodologies a comparison of five theoretical models.pdf. Computer, 28(12), 1135–1145.

Seale, C. (2004). Researching society and culture. Sage.

Shadish W R, Cook T D, Levitzin L C, 1991 Foundations of Program Evaluation: Theories and Practice (Sage, Newbury Park, CA)

Staples, M., & Niazi, M. (2010). Two case studies on small enterprise motivation and readiness for CMMI. Proceedings of the 11th International Conference on Product Focused Software - PROFES '10, (January 2010), 63–66. https://doi.org/10.1145/1961258.1961274

The Standish Group. (2015). Chaos Report 2015. Retrieved from https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf%0Ahttp://www.laboratorioti.com/2016/05/16/informe-del-caos-2015-chaos-report-2015-bien-mal-fueron-los-proyectos-ano-2015/

Tighy, G. (2012). Evaluation of software engineering management best practices in the Western Cape. Proceedings of the 2012 4th Software Engineering Colloquium, SE 2012, 21–23. https://doi.org/10.1109/SE.2012.6242352

Vijayasarathy, L. R., & Butler, C. W. (2016). Choice of Software Development Methodologies: Do Organizational, Project, and Team Characteristics Matter? IEEE Software, 33(5), 86–94. https://doi.org/10.1109/MS.2015.26

Wallin, C., & Rikard, L. (2010). Software development lifecycle models. The basic types. ACM SIGSOFT Software Engineering Notes, 35(3), 8. https://doi.org/10.1145/1764810.1764814

Wynekoop, J. L., & Russo, N. L. (1995). Systems development methodologies: Unanswered questions. Journal of Information Technology, 10(2), 65–73. https://doi.org/10.1057/jit.1995.9

Yu, J. (2018). Research Process on Software Development Model. IOP Conference Series: Materials Science and Engineering, 394(3). https://doi.org/10.1088/1757-899X/394/3/032045

Zainol, A., & Mansoor, S. (2008). Investigation into requirements management practices in the malaysian software industry. Proceedings - International Conference on Computer

Science and Software Engineering, CSSE 2008, 2, 292–295. https://doi.org/10.1109/CSSE.2008.962