# A SURVEY ON SOFTWARE MAINTENANCE PRACTICES IN KENYA

BY

MACHARIA, ROSEMARY WAITHERA

A MANAGEMENT RESEARCH PROJECT SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF BUSINESS ADMINISTRATION

UNIVERSITY OF NAIROBI
SCHOOL OF BUSINESS
NOVEMBER 2008

## DECLARATION

This research project is my original work and has not been presented for a degree in any other university.

Signed: _____ Date 3/12/08

Rosemary Macharia
D61/P/8658/04

This project paper has been submitted for examination with my approval as a university supervisor

Signed: _____ Date 3/12/2008

Joel K. Lelei
Lecturer
Department of Management Science
University of Nairobi

# ACKNOWLEDGEMENT

I thank God for the opportunity to undertake this degree and for the provision of all resources required.

I also thank my beloved husband, mother and father who sacrificed time and money to enable me to accomplish this degree. May God truly bless them.

I thank Mr. Lelei for his support, dedication, encouragement and patience throughout the research period. For his feed back and his attention to detail, I want to say, *asante.*

To my colleagues that we begun with the course and who in one way or other contributed to the success of this study, all I can say is thanks.

TABLE OF CONTENTS PAGE

# LIST OF TABLES

# LIST OF FIGURES

# ABSTRACT

This study examined the practice of software maintenance in Kenya. The need for the study arose from the fact that many firms in Kenya are heavily relying on ICT for business. ICT requires software to function. However software requires maintenance due to changes in technology and firms. The study covers background information, different types of software maintenance, the process followed in software maintenance, the tools used as well as the challenges encountered in software maintenance. The main objective of the study was to establish the practice of software maintenance in Kenya specifically to determine the level of importance placed on the different types of software maintenance, the processes and tools used to undertake software maintenance by firms and the challenges encountered in software maintenance.

A survey design was used for data collection from 53 ICT consultants and the data was collected using questionnaires. These were administered using a "drop and pick later" basis. Data was analyzed using frequencies; means; factor analysis techniques and standard deviation. The data was presented using tables accompanied by discussions.

Analysis revealed that majority of the firms undertook corrective maintenance followed by adaptive maintenance and perfective maintenance. Firms undertook software maintenance process activities to a moderate extent, although tools were used to a small extent. Several challenges were experienced although the major challenges were lack of management support and resources, lack of experienced staff as well as poor system testing. The least faced challenges were how the software was developed, in-house politics of contracting company and high software maintenance costs.

In conclusion it is seen that firms undertake one form or other of software maintenance although majority of the firms undertake corrective maintenance. Firms also undertake software maintenance process activities to a moderate extent. Software maintenance tools are used to a very small extent by firms. However as firms undertake software maintenance, they encounter several challenges and the major challenge being lack of management support and resources.

# CHAPTER ONE: INTRODUCTION

## 1.1 Background

The expansion of computing, fuelled by ongoing technology developments in computing, networked communications and the explosion of the Internet has altered the way firms work and compete in the global market. This has also led to firms integrating computers and communication facilities hence Information and Communication Technology (ICT) with the aim of transmitting, receiving and processing information. Using the hardware and software ICT components, shrinking barriers and time is possible.

Performance of hardware and software needs to be assured. Hardware performance assurance lie with the hardware manufacturers but software is with developers which could be manufacturers as well as consultants. Once software is developed changes are inevitable. This is because systems are tightly coupled with their environment and when a system is installed in an environment, it changes that environment and therefore changes the system requirements. On the other hand system requirements are likely to change while the system is being developed because the environment is changing, leading to a delivered system not meeting its requirements (Somerville, 2004). In addition firms have to enhance their information systems taking advantage of enhancement in technology. This is expected as developers are releasing more enhanced software which leads to the discontinuation of the previous version and results in firms being forced to upgrade the software that they have in order to continue being supported.

Changes are expected even when systems are working properly because successful software systems are still condemned to change over time (Lehman, 1984). A predominant proportion of changes are to meet the ever changing user needs (Somerville, 1995). Firms must accommodate changes given that, it is only those firms with the ability to permanently and to quickly adapt to the ever changing conditions that can survive (Mechanical Engineering Vol. 3, No 1, 2005). Further, only those firms which are capable of permanent and fast adaptation to quickly-changing conditions can survive. Therefore, firms need to maintain software programs so as to

extend their useful life or to keep them running efficiently (Reynolds, 1995). The adaptation implies changing and maintaining systems. Computer experts and specialists are now placing importance to the maintainability of the system.

Maintenance generally can be defined as the monitoring, evaluating and modifying of operational systems to make desirable improvements (O'Brien, 2001). This applies to computer hardware as well as computer software. Computer software maintenance is the general process of changing software after it has been delivered. Changes could be for correcting coding errors. Alternatively, there could be significant enhancements to correct specification errors or accommodate new requirements (Somerville, 1995). Software maintenance can also be viewed as including all changes made to a program or system after installation (Reynolds, 1995).

Software maintenance is part of the software development lifecycle (SDLC) which relates to models or methodologies that people use to develop computer systems. SDLC adheres to important phases that are essential for developers, such as planning, analysis, design and implementation, testing, software deployment, and maintenance. Maintenance is the last stage of the software life cycle (Post and Anderson, 2006). After the product has been released, the maintenance phase keeps the software up to date with environmental changes and the constantly changing user requirements.

Software maintenance according to Lientz and Swanson (1980) is of different types and these are corrective maintenance, adaptive maintenance, preventive maintenance and perfective maintenance. Corrective maintenance deals with the repair of the software and is reactive. Adaptive maintenance deals with customizing the software to the hardware and the operating system and is reactive. Perfective maintenance deals with making new changes to the software and is proactive. Preventive maintenance deals with increasing the systems maintainability, documentation and improving the modular structure and is proactive.

Software maintenance involves a process that begins with a change request from system users, management or customers (Pigoski, 1997). The process begins with developing plans that will be followed during the maintenance of the system. Cost and impact of changes are analyzed to

check how much of the system is affected and the cost of implementing them. If the changes were accepted, a new release of the system is planned and all the proposed changes (fault, adaptation, and new functionality) are considered. Finally the system is documented and migration is undertaken as well as the retirement of the previous system depending on the level of changes done. IEEE and ISO have both addressed software maintenance whereby the standard IEEE-1219 (1998) is specifically concerned with software maintenance and the standard ISO-12207 (1995) deals with the totality of the processes comprised in the software life cycle.

Tools are important in software maintenance as they are used to simplify tasks and increase the efficiency and productivity (Takang and Grubb, 1996). These tools include program understanding tools, reverse engineering tools, testing tools and configurations management tools. Program understanding tools convert code into text so as the maintainer can understand the program. Reverse engineering tools enable the maintainer to analyze the software in detail and make a change of the software without starting from scratch. After a successful change on software has been done, testing of the software has to follow thereafter. Testing tools on the other hand enable the maintainer to develop test cases that will be used in software maintenance. When a change is completed, the software is tested and configuration management and documentation tools are used to accomplish the task (Takang and Grubb, 1996).

Despite firms adhering to the maintenance process keenly and using available tools to undertake the process, there are several challenges that they may face in software maintenance. Software maintenance is viewed with a negative attitude (Wireman, 2003) and seen as the second-class activity by most managers. Such managers would have no incentive to spend money which would lead to reduced cost of software change in the future. Management also employ personnel who are either not competent and do not have an understanding of the software. Another challenge is attributed to the fact that software maintenance firms also do not have access to the required tools and equipment to undertake software maintenance. Furthermore, during the maintenance process, development engineers, managers, and others are diverted from their primary responsibilities to assist with high priority maintenance tasks, often in a crisis situation. Lack of complete understanding of the structure, behavior and functionality of the software being

modified is another challenge that software maintenance personnel experience because they were not involved in the development of the system.

Several studies have been undertaken internationally on software maintenance. One study on "Software Maintenance Maturity Model" addressed the improvement of the maintenance process (April et al, 2004). A study by Canfora and Cimitile (2000) on the other hand, concentrated on software maintenance its problems and solutions available. Another study concentrated on finding out more about the tools, procedures, and techniques project personnel use in their work (Dart et al, 1993). Bennett and Rajlich (2000), also undertook a study on "software maintenance and evolution" with the aim of improving the speed and accuracy of change.

Research carried out in Kenya on ICT is still in the infancy stage as researchers have concentrated more on the telecommunication aspect of ICT. Research has also been carried out by MBA students on application of ICT for competitive advantage of firms listed at Nairobi Stock Exchange by Taneja (2006), user involvement in the development of systems by Mwaniki (2004) and testing of software systems by Chepchieng (2006). There has however been no study undertaken on the practice of software maintenance in Kenya and hence this study will form a foundation to further research on software maintenance.

## 1.2 Summary

It is clear that software maintenance is of different types. Whichever type of maintenance a firm chooses to carry out they have to develop maintenance plans that will be followed as well as utilize tools that will make the handling of the maintenance tasks easier. During the whole software maintenance process, several challenges can be faced.

## 1.3 Statement of the problem.

The need for firms to enhance competitive advantage and productivity has led to reliance on ICT. ICT has enabled firms to merchandise high profile brands at the required volumes, at accessible locations, with acceptable margins and hence increased quality, quantity and availability of information shared between suppliers and customers. To ensure continued service in ICT,

software maintenance is inevitable. Managers are now awakening to the realization of its importance as business software degrades and changes over time. Software maintenance therefore has become imperative and managers need to ensure its effectiveness (Campbell and Picknell, 2006).

To ensure effective software maintenance, appropriate types of maintenance need to be chosen. This ensures that software maintenance being undertaken is of the correct nature. Moreover, software maintenance processes ought to be adhered to, followed and completed effectively when undertaking whichever type of maintenance. While undertaking the maintenance process, it is also important to use the right software maintenance tools. Use of the right tools will simplify the task and increase efficiency and productivity in maintenance. Lastly, as the person dealing with maintenance undertakes the maintenance process, challenges will be encountered. The challenges have to be managed in order to ensure effective maintenance.

Studies have been undertaken in Kenya relating to software. Included among them is "User involvement during development of software in Kenya" by Mwaniki (2004), "Application of ICT for competitive advantage of firms listed at Nairobi Stock Exchange" by Taneja (2006) and "Software testing processes used by developers in Kenya" by Chepchieng (2006). In the studies, software maintenance in Kenya has been left out.

Research on software maintenance processes and tools has been undertaken in other countries and findings have been documented. These findings may differ if applied in a Kenya situation due to the different environments. There is need therefore to undertake research in respect of the practice of maintenance of software in Kenya. The questions to be addressed in light of effective software maintenance are:

What types of software maintenance are undertaken in Kenya?
What are the processes and tools used in software maintenance by firms in Kenya?
What are the challenges encountered in software maintenance in Kenya?

## 1.4 Objectives of the study

The main objective of the study is to establish the practice of software maintenance in Kenya. Specific objectives are:

1. Establish the type of software maintenance undertaken
2. Determine the processes used in maintenance of software
3. Establish the tools used in maintaining software
4. Establish the challenges encountered in maintenance of software

## 1.5 Importance

The findings of this study are expected to be of interest to several persons. Management of firms will be informed in terms of status of software maintenance and can use the findings in decision making as to which type of maintenance to select as well as the tools that are readily available for use in undertaking the maintenance process. Management will also appreciate the various challenges that are associated with the maintenance of computer software and can prepare to address them.

System developers and vendors on the other hand will use the findings in their design solutions thus design the systems well and document them having taken the study into consideration. This then will lead to software that is maintainable.

The government will also benefit from this study. It could use the findings to give advisory services to the public on maintenance of computer software systems and therefore derive more benefits from ICT while reducing the challenges encountered during the process.

Societies in Kenya involved in ICT will benefit too from this study. They can put the findings in their newsletters, making the community at large more informed. The community would as such place more importance on maintenance of computer software.

The findings in this study will be used as a basis for further research on software maintenance.

# CHAPTER TWO: LITERATURE REVIEW

## 2.1 Introduction

Firms have goals to achieve and for that case they perform activities which need information and which in turn require information systems for its processing. Information systems should thus be considered in terms of their purpose in a firm. In information systems, computers systems have to communicate together, hence the need for data communications and the Internet. Indeed, everything that presently exists on a personal computer or a laptop, experts suggest, will move onto the Internet (Laudon and Laudon, 2006).

The Internet plays a big role in transforming the global business landscape, hence the resulting Internet culture where people expect services and dealings over the Internet. This results in the transformation of business firms into digital firms. The digital firms are digitally enabled, and all of the significant business relationships with customers, suppliers and employees is conducted digitally (Laudon and Laudon, 2006). Digital firms use the Internet and networking technology to make data flow seamlessly among different parts of the firm and create electronic links with other parties.

Components of the Internet and networking technology comprise of communication devices which basically are hardware and software. Continued service must be guaranteed from the components and hence the reason for maintenance. Hardware maintenance lies on the manufactures; firms need to concentrate on the maintenance of software. Software development goes through a lifecycle has many phases. They include requirements engineering, architecting, design, implementation, testing, software deployment, and maintenance (Erdil et al, 2006). The requirements, engineering and the architecture phase includes collecting the requirements from users and developing a project plan. The design phase plans the structure in a way that can be easily altered. Similarly, the implementation phase creates code that can be easily read, understood, and changed. Maintenance is the last stage of the software life cycle.

Maintenance phase keeps the software up to date with environmental changes and changing user needs (O'Brien, 2002). Software maintenance process is therefore important and necessary for continued Information system services. The process of software maintenance becomes of increasing importance for firms, as it directly affects other relevant processes leading to customers' satisfaction or dissatisfaction. Hung (2008), emphasizes the growing importance of software maintenance in firms and hence management in various firms needs to understand the type of change they seek to undertake on software.

## 2.2 Types of software maintenance

Lientz and Swanson (1980), define four different types of software maintenance which are categorized as corrective, adaptive, perfective, and preventive. Corrective maintenance deals with the repair of faults or defects found. A defect can result from design errors, logic errors and coding errors (Takang and Grubb, 1996). All these errors, sometimes called 'residual errors' or 'bugs', prevent software from conforming to its agreed specification. The need for corrective maintenance is usually initiated by bug reports drawn up by the end users (Coenen and Bench-Capon, 1993).

Adaptive maintenance consists of adapting software to changes in the environment, such as the hardware or the operating system. Takang and Grubb (1996) define the environment in context as totality of all conditions and influences which act from outside upon the system. Examples of such influences and conditions are business rule, government policies, work patterns, software and hardware operating platforms.

Perfective maintenance deals with accommodating to new or changed user requirements as well as functional enhancements to the system and activities needed to increase the system's performance or to enhance its user interface (Vliet, 2000). A successful piece of software tends to be subjected to a succession of changes, resulting in an increase in the number of requirements. This is based on the premise that as the software becomes useful; the users tend to experiment with new cases beyond the scope for which it was initially developed (Takang and Grubb, 1996).

Preventive maintenance concerns activities such as updating documentation, adding comments, and improving the modular structure of the system that are aimed at increasing the system's maintainability (Vliet, 2000). The long-term effect of corrective, adaptive and perfective changes increases the system's complexity (Takang and Grubb, 1996). As a large program is continuously changed, its complexity, which reflects deteriorating structure, increases unless work is done to maintain or reduce it.

Among these four types of maintenance, only corrective maintenance is 'traditional' maintenance. The other types can be considered software 'evolution'. It characterizes the growth dynamics of software (Chapin et al., 2001). A study was done by Lientz and Swanson in the late 1970s (Lehman, 1980) and they found out that around 75% of the maintenance effort was on adaptive and perfective maintenance and error correction consumed about 21% of the effort. Given the importance of the type maintenance, business success requires efficient, predictable and high quality of the software maintenance processes.

## 2.3 Software maintenance process

Successful software requires successful software maintenance standards. IEEE and ISO are renowned standards that address software maintenance. IEEE and ISO have both addressed software maintenance whereby the standard IEEE-1219 (1998) is specifically concerned with software maintenance and the standard ISO-12207 (1995) deals with the totality of the processes comprised in the software life cycle. The standard ISO-12207 (1995) software lifecycle identifies seventeen processes of which maintenance is one as seen on Appendix I.

Software maintenance provides for conducting major functions and initiating and exploiting support and organizational processes (Canfora and Climitile, 2000). There are several activities that are undertaken to achieve successful software maintenance.

The maintenance process begins with developing plans and procedures for software maintenance, creating procedures for receiving, recording, and tracking maintenance requests, and establishing an organizational interface with the configuration management process. Pigoski

(1997) affirms that maintenance plans (Appendix II) should be prepared in parallel with the development plans. The activity entails the definition of the scope of maintenance and the identification and analysis of alternatives, including offloading to a third party; it also comprises organizing and staffing the maintenance team and assigning responsibilities and resources.

Once the plan is completed the next activity is concerned with the analysis of the maintenance request, either a problem report or a modification request, to classify it, to determine its scope in term of size, costs, and time required, and to assess its criticality. This activity is said to consume more than half of the maintenance resources (Fjeldstad and Hamlen, 1982). It is also important to identify where in the system changes are to be effected and have an in-depth knowledge of how the parts to be corrected or modified work (Takang and Grubb, 1996) before developers begin the implementation of the changes.

Developers then begin the implementation of the changes while undertaking a change impact analysis. Change impact analysis is the activity by which the programmers assess the extent of the change (Bohner and Arnold, 1996). The documentation on the impact is done and the software is given to the developers to implement the change. Once the development process is complete several supporting processes may be invoked, including the quality assurance process, the verification and validation process as well as testing. The software has to undergo regression testing to confirm that no new faults have been added (Leung and White, 1990) .Testing procedures are also considered to ensure that the new or modified requirements are completely and correctly implemented. The original unmodified requirements should are not be affected. The software is then documented so that future changes will rely on the documentation of the previous changes/modifications (Kagan et al, 2003).

Once documentation of the software is complete, the maintained software is migrated from the development environment to the live environment. Migration plans are developed to help ensure that the operational transition to the new system goes smoothly (Bergey et al, 2001). In addition management requires that a retirement plan of the old software be developed and that all relevant stakeholders be notified. The use of tools for software maintenance simplifies the tasks and increases efficiency and productivity.

## 2.4 Software maintenance tools

The task of software maintenance has become so vital and complex that automated support is required to do it effectively. A software maintenance tool is an artifact that supports a software maintenance staff in performing a task (Takang and Grubb, 1996). Selecting a tool that promotes understanding is very important in the implementation of change since a large amount of time is used to study and understand programs (Hung, 2008). Selecting the right tool should be based on its various capabilities, features, cost/benefit, platform, programming language, ease of use, openness of architecture, stability of vendor, and organizational culture (Kagan et al, 2003).

Capability decides whether the tool is capable of fulfilling the task. Tools must be analyzed for the benefits it brings against its cost (Kagan et al, 2003). The benefit indicators of a tool are quality, productivity, responsiveness, and cost reduction. The tool should have a similar feel to the ones that the users are already familiar with and should have the ability to be integrated with different vendor tools. The vendor in return should be capable of supporting the tool in the future.

Tools are of various types and they must support program understanding, reverse engineering, testing, configuration management, and documentation (Takang and Grubb, 1996). Examples of program understanding and reverse engineering tools include the program slicer, static analyzer, dynamic analyzer, cross-referencer and dependency analyzer (Takang and Grubb, 1996).

The slicing static analyzer tool helps the programmers select and view only the parts of the program that are affected by the changes. The slicing static analyzer is used in analyzing the different parts of the software program such as modules, procedures, variables, data elements, objects and classes by marking all the sections of a program text that may influence the value of a variable at a given point in the software program (Weiser, 1984). The slicing static analyzer allows general viewing of the program text and generates summaries of contents and usage of selected elements in the program text, such as variables or objects (Takang and Grubb, 1996). It can be used in all the different types of software maintenance as the developers seek to understand how the program works.

The dynamic analyzer tool analyzes the software program while it is executing. A component of the dynamic analyzer tool is a data flow analyzer. It allows tracking data flow and control flow paths in the software (Vanek and Davis, 1990). It also allows for regression testing ensuring that no new faults have been introduced while displaying the relationship between components of the system. This tool is best applied in corrective software maintenance in the repair of defects and faults that may occur across the different components of the system, focusing on the parts that are affected by the change (Kagan et al, 2003).

The cross-referencer and dependency analyzer tools produce information on the usage of a software program as well as aid the software maintenance staff to analyze and understand the interrelationships between entities in a program (Takang and Grubb, 1996). The cross-referencer tool helps the user focus on the parts that are affected by the change while dependency analyzer provides capabilities to set up and query the database of the dependencies in a software program as well as provide graphical representations of the dependencies.

The test simulator tool aids in testing the effects of the change in a controlled environment before implementing the change on the actual system. Testing is the most time consuming and demanding task in software maintenance (Takang and Grubb, 1996) hence the use of test case generator to produce test data that is used to test the functionality of the modified system, while a test path generator helps find all the data flow and control flow paths affected by the changes.

Configuration management and version control tools help store the objects that form the software system whereby a source control system is used to keep a history of the files so that versions can be tracked thus controlling versions and that take place over time (Kagan et al, 2003).

Commercially available software maintenance tools include ProTeus III Expert CMMS by Eagle Technology, Inc. and are a maintenance software package that lets users schedule preventative maintenance, generate automatic work orders, document equipment maintenance history, track assets and inventory, track personnel, create purchase orders, and generate reports. Microsoft Visual Source Safe is a source control system tool that is used by configuration management. Another tool is the bug tracking tools, which play an important role in maintenance. Bugzilla by

the Mozilla Foundation is an example of such a tool. Other bug tracking products are Test director by Mercury Interactive, Silk Radar by Segue Software, SQA Manager by Rational software, and QA director by Compuware.

Products that are specific to programming languages are CCFinder and JAAT which is specifically designed for JAVA programs (Kamiya et al, 2001). CCFinder identifies code clones in JAVA program. JAAT executes alias analysis for JAVA programs. For C++ programs, there is a tool called OCL query-based debugger which is a tool to debug C++ programs using queries formulated in the object constraint language, OCL (Hobatr and Malloy, 2001).

## 2.5 Challenges encountered in the maintenance of computer software

Problems that are associated with software maintenance can be traced to deficiencies of the software development process. Sneidewind (1987) asserts that "the main problem in doing software maintenance is that we cannot do maintenance on software which was not designed for maintenance". However, there are intrinsic characteristics of software and its production process that contribute to making software maintenance challenging. Hung (2008), classifies software maintenance challenges into three categories: Cost estimation and measures challenges, technical challenges and managerial challenges.

Software maintenance costs consume a major share of software life cycle financial resources (Vliet, 2000) as well as available effort (McKee, 1984). According to Canning (1972), software maintenance was characterized as an "iceberg" to highlight the enormous mass of potential problems and costs that lie under the surface. Somerville (1995), states that the cost of software maintenance represents a large proportion of the budget of most firms that are largely due to enhancements rather than corrective in nature. Software maintenance costs are significantly affected by age, size Lientz and Swanson (1981) and complexity (Banker *et al.,* 1993). Software that has advanced in age will require more resources to maintain due to the structural changes that it has gone through over time. As the software changed over time, it becomes more complex and therefore the software engineers require more time to understand it resulting to higher costs. Understanding the complete structure of the software to be modified proves as a major challenge software maintenance staff.

Software maintenance staff needs to gain a complete understanding of the structure, behavior and functionality of the software being modified thus modification proposals to accomplish the maintenance objectives can be generated. Available estimates indicate that the percentage of software maintenance time consumed on program comprehension ranges from 50% up to 90% (Corbi, 1989; Livadas, 1994; Standish, 1984) and is frequently compounded because the software maintenance staff is rarely the author of the code or a significant period of time has elapsed between development and software maintenance and a complete, up-to-date documentation is even more rarely available (Canfora and Cimitile, 1999). Developers are not there to explain the code and therefore maintainers must learn the software on their own.

Measuring the effect of a proposed modification is another major challenge in software maintenance. Impact analysis is the activity of assessing the potential effects of a change with the aim of minimizing unexpected side effects (Queille et al, 1994; Yau and colleferro 1980). The task involves assessing the appropriateness of a proposed modification and evaluating the risks associated with its implementation, including estimates of the effects on resources, effort and scheduling. Software maintenance staff must possess an intimate knowledge of the software's structure and content and estimate of the resources needed to accomplish the change.

Testing is another major issue that is experienced during software maintenance. Leung and White (1990) have defined regression testing as the process of testing a system after it has been modified with an aim of establishing confidence that changes are correct and that unchanged portions of the system have not been affected. Indeed, changes made during a software maintenance process are usually small and, therefore, the simple approach of executing all test cases after each change may be excessively costly. When software maintenance staff comes across an error during maintenance, they have to go back to the drawing board resulting in the delay of deploying the software.

Alignment with the firms' objectives is a challenge that management has to contend with. Software maintenance often has the objective of extending the life of software for as long as possible as well meeting user demand for software updates and enhancements. This implies that software maintenance process demands time and resources that are likely to overrun during the

process. Another challenge is that senior management view on software maintenance is often of an activity consuming significant resources with no clear quantifiable benefit for the firm thus return on investment is not clear (Dorfman and Thayer, 1997). This later leads to requesting for resources difficult.

Inexperienced staff is another common managerial problem of software maintenance in firms. Beath and Swanson (1989) reported that 25% of the people doing software maintenance are students and up to 61% are new hires. Pigoski (1997) confirms that 60% to 80% of the software maintenance staff is newly hired personnel. Software maintenance is still perceived by firms as a non strategic issue, thus hiring students and new people. As a result they will take a much longer time understanding the code before they can make the change thus causing the costs to increase. Also due to the fact that most firms view maintenance as a non strategic issue, there is a tendency for the software maintenance personnel to be viewed as "second-class citizens" and morale therefore suffers (Deklava, 1992) resulting in delay of deploying a change. Firms require appreciating all the above challenges first and managing them to in order to effectively meet the software maintenance objective.

## 2.6 Studies undertaken on software maintenance

Bennett and Rajlich (2000) undertook a study on "software maintenance and evolution" The main objective was to describe a landscape for research in software maintenance and evolution in order to improve the speed and accuracy of change while reducing costs. They identified key problems, promising solution strategies and topics of importance. In their study they did not highlight activities that companies undertake when performing software maintenance.

"Software Maintenance" was undertaken by Canfora and Cimitile (2000). The main objective of the study was to present software maintenance not as a problem, but in terms of solutions. They described software maintenance, its relevance, problems, and available solutions.

April et al (2004) also undertook a study on "Software Maintenance Maturity Model: The software maintenance process model". The main objective was to address the assessment and improvement of the software maintenance function. They proposed improvements to the

software maintenance standards and introduced a proposed maturity model for daily software maintenance activities.

Locally there have been any studies done on "Information systems projects in Kenya: A study of user involvement" by Mwaniki (2004), and another done by Chepchieng (2006) on "Software testing processes used by software developers in Kenya". Software maintenance and its practice have not been explored locally hence the need for this research in Kenya.

## 2.7 Summary and Conclusion

The way in which software systems are designed and built is changing profoundly, and this will definitely have a major impact on future software maintenance. Object technology, commercial-off-the-shelf products, computer supported cooperative work, outsourcing and remote software maintenance, and Internet/Intranet enabled systems and infrastructures, are a few examples of areas that will impact on software maintenance.

There are several types of software maintenance that management may choose to undertake depending on the nature of the change and they include corrective, perfective adaptive and preventive maintenance. Software maintenance involves process which consists of developing software maintenance plans, impact analysis and system migration. When firms undertake software maintenance of the systems they use, there are several challenges that they are likely to face.

Though several studies have been undertaken both locally and internationally, there still remains a gap in the research of the practice of software maintenance in Kenya and hence the need for this research.

# CHAPTER THREE: RESEARCH METHODOLOGY

## 3.1 Research design

This is a survey of software maintenance practice specifically concerning types, tools and processes used by firms in Kenya, as well as challenges faced in software maintenance by ICT consultants in Kenya. A survey design was chosen given that no documentation had been done in relation to the processes, tools, types and challenges of software maintenance undertaken in Kenya.

## 3.2 Population

The population of the study consists of ICT firms that provide maintenance of computer software services to various firms in Kenya. The list of firms was chosen from the yellow pages of the Kenya telephone directory (2007) and the Nation Business Directory (2006) and specifically those in Nairobi. The survey targeted ICT consultancy firms in Nairobi due to their broad knowledge on developing and maintaining computer software. Most firms also have their headquarters in Nairobi.

## 3.3 Sampling

The study conducted a survey of consulting firms that dealt with system software. Purposive sampling was used in selecting the respondents who were ICT consultants. Eighty five questionnaires were administered and fifty three completed responses were received back.

## 3.3 Data collection

Primary data was collected using questionnaires which were divided into 5 sections namely A, B, C, D and E. They were administered to ICT consultants through "drop and pick later" method. The questionnaire was subdivided into 3 sections. Section A covered the demographic information. Section B covered the types of software maintenance. Section C covered the software maintenance processes. Section D covered the tools used in software maintenance. Section E covered the challenges experienced during software maintenance.

## 3.4 Data analysis

The data that was collected was coded, collated and edited for errors, mistakes, uniformity of consistency and completeness. Data collected in respect of Section A was analyzed using frequencies and percentages and presented using tables. This was so as to know who the respondents and the firms were and their nature.

Data collected in Section B, C and D was analyzed using frequencies and percentages to test the extent firms undertook all activities in the software process as well as the software maintenance types and tools used.

Data collected in Section E, was analyzed using factor analysis to test the expected challenges. Factor analysis refers to a collection of statistical methods for reducing correlation data into a smaller number of dimensions or factors. There are two types of factor analysis. They are Exploratory and Confirmatory factor analysis. Confirmatory factor analysis (CFA) is a special form of factor analysis used to assess the number of factors and the loadings of variables allowing for the explicit constraint of certain loadings to be zero. Exploratory factor analysis (EFA) is generally used to discover the factor structure of a measure and to examine its internal reliability. The researcher used the exploratory factor analysis method to reduce the number of interrelated variables, relating challenges in software maintenance, to a limited number of factors. The analysis was performed using statistical analysis software package (SPSS).

# CHAPTER FOUR: DATA ANALYSIS AND FINDINGS

## 4.1 Introduction

The data was collected using questionnaires of which 85 questionnaires were dispatched. Of the total 85 questionnaires distributed, 60 questionnaires were received back. Questionnaires that were rejected were 7 for being incomplete on material items. Consequently, the study was based on 53 questionnaires. Given the results of the response rate, the study was based on the 53 ICT consultants whose survey was accepted. This represents a 62.4% of the target population.

## 4.2 Demographic information

### 4.2.1 Respondents profile

The data on the age and gender was collected in order to get a demographic representation of the professionals being questioned. At the personal response level, the highest numbers of respondents were males, who were 34. The female respondents were only 19. The respondents who were between 26 years and 35 years old were the majority. There is a big difference of respondents' gender due to the male dominance of ICT managerial positions found in Kenya.

**Table 4.1: Gender of respondents**

| | Gender | | |
|---|---|---|---|
| **Age** | **Male** | **Female** | **Total** |
| Below 25 | 0 | 2 | **2** |
| 26-30 | 13 | 9 | **22** |
| 31-35 | 13 | 7 | **20** |
| 36-40 | 6 | 0 | **6** |
| 41-45 | 2 | 1 | **3** |
| **Total** | **34** | **19** | **53** |

One of the factors considered in work experience is the length of time one works. Table 4.2 shows that 16 of the respondents had worked for 0 to 5 years. The 16 respondents were of the age group 26 to 30 years followed by 7 respondents who had worked for 6 to 10 years. There were only 3 respondents who were between 41years and 45 years. The respondents have the experience sought for.

**Table 4.2: Number of years worked in relation to the age bracket of the employees**

| | Number of years worked | | | | |
|---|---|---|---|---|---|
| Age | 0-5 years | 6-10 Years | 11-15 years | 16-20 Years | Total |
| Below 25 | 0 | 2 | 0 | 0 | 2 |
| 26-30 | 16 | 6 | 0 | 0 | 22 |
| 31-35 | 6 | 7 | 5 | 2 | 20 |
| 36-40 | 1 | 3 | 2 | 0 | 6 |
| 41-45 | 0 | 0 | 2 | 1 | 3 |
| **Total** | **23** | **18** | **9** | **3** | **53** |

### 4.2.2 Firm profile

Firm ownership aimed at establishing the ownership of the respondent firms. Table 4.3 shows the distribution of the firms respondents by ownership. Most of the firms are locally owned with 86.8% which implies that the study results should be a fair indicator of the local professionals' practice of software maintenance.

**Table 4.3: Firm ownership**

| Ownership | Frequency | Percent |
|---|---|---|
| Local | 46 | 86.8 |
| Foreign | 2 | 3.8 |
| Both local and Foreign | 5 | 9.4 |
| **Total** | **53** | **100.0** |

Years in operation gives an indication of when the firm came into operation. As indicated in Table 4.4, most of the firms had been in operation for 6 to10 years and 0 to 5 years.

**Table 4.4: Years in operation**

| Years in operation | Frequency | Percent |
|---|---|---|
| 0-5 | 15 | 28.3 |
| 6-10 | 15 | 28.3 |
| 11-15 | 9 | 17.0 |
| 16-20 | 1 | 1.9 |
| Over 21 | 13 | 24.5 |

Table 4.5 respondents in 8 firms who have been in operation for 6 to 10 years have an annual turn over of above 6 million while only 6 respondent firms who have been in operation for over 21 years have an annual turn over of 6 million.

**Table 4.5: Annual turnover in relation to the number of years the firm has been in operation**

| Number of years the firms been in operation | Average annual turnover | | | | | Total |
|---|---|---|---|---|---|---|
| | 0-1.5M | 1.6M-3M | 3.1M-4.5M | 4.6M-6M | 6.1M and Above | |
| 0-5 years | 2 | 4 | 0 | 4 | 5 | 15 |
| 6-10 years | 3 | 3 | 1 | 1 | 8 | 16 |
| 11-15 years | 0 | 1 | 0 | 2 | 7 | 10 |
| 16-20 years | 0 | 0 | 0 | 0 | 2 | 2 |
| Above 21 years | 0 | 3 | 0 | 1 | 6 | 10 |
| Total | 5 | 11 | 1 | 8 | 28 | 53 |

The firms from which the respondents were drawn were analyzed in terms of the number of employees. Table 4.6 indicates that majority of the respondent firms had between 41 to 60 employees which were the highest at 20.8% while only 1.9% of the respondent firms had between 61 and 80 employees. From the data collected on Table 4.5 and Table 4.6 on the number of employees, the firms' size is well represented in data collection.

**Table 4.6: Number of employees in the firms**

| Employees | Frequency | Percent |
|---|---|---|
| 1-20 | 3 | 5.7 |
| 21-40 | 6 | 11.3 |
| 41-60 | 11 | 20.8 |
| 61-80 | 1 | 1.9 |
| 81-100 | 8 | 15.1 |
| Above 101 | 24 | 45.3 |
| **Total** | **53** | **100.0** |

Firms from which respondents were drawn were analyzed according to the services that they offer. As seen in Table 4.7, the highest number of respondents (67.9%) undertook information system audit. Respondents who undertook system development as well as software maintenance tied at 56.6%. There was a positive response on all services offered by the consultants.

**Table 4.7: ICT services offered by the respondent firms**

| Service offered | Response | Frequency | Percent |
|---|---|---|---|
| Information system development | Yes | 30 | 56.6 |
| | No | 23 | 43.4 |
| Information system audit | Yes | 36 | 67.9 |
| | No | 17 | 32.1 |
| Software maintenance | Yes | 30 | 56.6 |
| | No | 23 | 43.4 |
| Hardware maintenance | Yes | 21 | 39.6 |
| | No | 32 | 60.4 |
| System integration | Yes | 26 | 49.1 |
| | No | 27 | 50.9 |
| Enterprise security | Yes | 30 | 56.6 |
| | No | 23 | 43.4 |
| Information Technology architects services | Yes | 15 | 28.3 |
| | No | 38 | 71.7 |

## 4.3 Types of software maintenance and the extent firms undertook them.

The first objective of the study was to establish software maintenance types undertaken. The data collected was analyzed using frequencies, means, and standard deviations and the results were presented in tables.

**Table 4.8: Type of software maintenance firms undertook.**

| Maintenance type | Response | Frequency | Percent |
|---|---|---|---|
| Corrective maintenance | Yes | 50 | 94.3 |
| | No | 3 | 5.7 |
| Adaptive maintenance | Yes | 46 | 86.8 |
| | No | 7 | 13.2 |

| Maintenance type | Response | Frequency | Percent |
|---|---|---|---|
| Preventive maintenance | Yes | 19 | 35.8 |
| | No | 34 | 64.2 |
| Perfective maintenance | Yes | 49 | 92.5 |
| | No | 4 | 7.5 |

From Table 4.8 it is evident that 94% of the firms undertake corrective maintenance while the least was perfective maintenance. It is seen from Table 4.9 that the range is from 3.47 to 2.26. This implies that firms undertook the different software maintenance types to a moderate extent.

**Table 4.9: Extent to which the maintenance type is undertaken**

| Maintenance type | Mean | Std. Deviation |
|---|---|---|
| Extent corrective maintenance is undertaken | 3.36 | 1.002 |
| Extent adaptive maintenance is undertaken | 3.4 | 1.115 |
| Extent perfective maintenance is undertaken | 3.47 | 1.012 |
| Extent preventive maintenance is undertaken | 2.26 | 1.403 |

## 4.4 Extent to which firms pursue activities in software maintenance process.

The second objective was to establish the extent to which firms undertook all activities in the software maintenance process. It is seen in Table 4.10 that the mean ranges from 2.42 and 3.36. Majority of the activities lie in the 3 mark which according to the scale used implies that most firms undertook the activities to a moderate extent.

**Table 4.10: Activities in the software maintenance process**

| Software maintenance process activities | Mean | Std. Deviation |
|---|---|---|
| Development of plan | 3.02 | .971 |
| Implementation of plan | 3.170 | .8930 |
| Creating procedures for tracking requests | 2.60 | .884 |
| Implementing procedures | 2.72 | .928 |
| Analysis of request | 3.11 | .870 |
| Development and documentation of request | 3.19 | .810 |

| Software maintenance process activities | Mean | Std. Deviation |
|---|---|---|
| Identification of items to be modified | 3.34 | .831 |
| Development of test scenarios and implementation | 3.36 | .762 |
| Assessing integrity of modified system | 3.26 | .788 |
| Development of migration plans | 2.53 | .932 |
| Development of retirement plan | 2.42 | 1.262 |

## 4.5 Extent to which software maintenance tools are used by firms

The third objective was to establish to what extent firms used software maintenance tools as well as the importance they placed when choosing the maintenance tools. The results are shown in Table 4.11. In Table 4.11 the mean lies between 2.0 and 2.9 which according to the scale used imply that firms use maintenance tools to a very small extent. Testing tools are the only tools that lie near the mean 3 implying that it's the only tool that was used to a moderate extent.

**Table 4.11: Software maintenance tools**

| Software maintenance tools | Mean | Std. Deviation |
|---|---|---|
| Slicing tools | 2.04 | 1.018 |
| Dynamic analyzer tools | 2.30 | 1.170 |
| Cross-reference tools | 2.19 | 1.194 |
| Dependency analyzer tools | 2.11 | 1.138 |
| Test simulator tools | 2.92 | 1.016 |
| Configuration management tools | 2.08 | 1.207 |

The level of importance that firms placed when selecting the software maintenance is to a moderate extent. Table 4.12 shows an average mean of 3.5. This implies that the respondents placed some importance on the various factors when selecting software maintenance tools although they use the tools to a small extent.

**Table 4.12: Level of factors when selecting software maintenance tools**

| Factors | Mean | Std. Deviation |
|---|---|---|
| Ease of use | 3.38 | 1.096 |
| Capabilities of tools | 3.64 | 1.039 |
| Availability of tool | 3.58 | 1.046 |
| Stability of tool | 3.62 | 1.096 |
| Availability of trained staff | 3.25 | 1.09 |
| Tools used in Kenya | 2.98 | 1.083 |
| Sustainability of tool | 3.36 | 1.178 |
| Getting data in and out of the system in a standard format | 3.47 | 1.103 |
| Cost of acquisition | 3.68 | 1.088 |

## 4.6 Extent to which firms face software maintenance challenges

The fourth objective addresses the challenges that firms face during software maintenance. Data collected was subjected to factor analysis. Factor analysis is performed by examining the pattern of correlations between observed measures. Measures that are highly correlated (either positively or negatively) are likely to be influenced by the same factors while those that are uncorrelated are likely to be influenced by different factors. The process and result of factor analysis are discussed in Section 4.6.1 to 4.6.6.

### 4.6.1 Correlation Matrix

The respondents indicated the extent to which each of the 19 factors were a challenge during software maintenance. Since there might have been some group of factors that were similar to each other, factor analysis was used to identify and group such factors together in a correlation matrix. The correlation matrix gives correlations between all pairs of data sets. In correlation matrix of variables, the existence of clusters of large correlation coefficient between subsets of the variables suggests that the variables could be measuring aspects of the same underlying dimension or factors. Table 4.13 shows the correlation matrix of challenges faced during software maintenance.

**Table 4.13: Correlation Matrix**

| FACTOR | F1 | F2 | F3 | F4 | F5 | F6 | F7 | F8 | F9 | F10 | F11 | F12 | F13 | F14 | F15 | F16 | F17 | F18 | F19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| F1 | 1.000 | .828 | .761 | .302 | .438 | .478 | .490 | .380 | .783 | .412 | .379 | .371 | .358 | .521 | .449 | .740 | .693 | .677 | .686 |
| F2 | .828 | 1.000 | .801 | .292 | .420 | .423 | .472 | .542 | .743 | .373 | .300 | .259 | .328 | .467 | .269 | .581 | .657 | .562 | .738 |
| F3 | .761 | .801 | 1.000 | .350 | .362 | .452 | .243 | .395 | .627 | .290 | .239 | .275 | .260 | .406 | .237 | .562 | .591 | .594 | .487 |
| F4 | .302 | .292 | .350 | 1.000 | .366 | .437 | .235 | .255 | .269 | .188 | .219 | .228 | .314 | .232 | .231 | .267 | .195 | .181 | .098 |
| F5 | .438 | .420 | .362 | .366 | 1.000 | .695 | .600 | .312 | .429 | .271 | .376 | .442 | .415 | .313 | .235 | .367 | .327 | .352 | .409 |
| F6 | .478 | .423 | .452 | .437 | .695 | 1.000 | .743 | .576 | .487 | .226 | .442 | .386 | .486 | .393 | .286 | .479 | .407 | .416 | .423 |
| F7 | .490 | .472 | .243 | .235 | .600 | .743 | 1.000 | .666 | .570 | .204 | .439 | .307 | .528 | .461 | .230 | .533 | .560 | .430 | .577 |
| F8 | .380 | .542 | .395 | .255 | .312 | .576 | .666 | 1.000 | .581 | .152 | .326 | .153 | .524 | .470 | .182 | .366 | .423 | .410 | .557 |
| F9 | .783 | .743 | .627 | .269 | .429 | .487 | .570 | .581 | 1.000 | .248 | .252 | .256 | .322 | .460 | .258 | .578 | .621 | .576 | .616 |
| F10 | .412 | .373 | .290 | .188 | .271 | .226 | .204 | .152 | .248 | 1.000 | .681 | .537 | .441 | .201 | .464 | .073 | .117 | .233 | .204 |
| F11 | .379 | .300 | .239 | .219 | .376 | .442 | .439 | .326 | .252 | .681 | 1.000 | .717 | .666 | .450 | .553 | .195 | .121 | .199 | .283 |
| F12 | .371 | .259 | .275 | .228 | .442 | .386 | .307 | .153 | .256 | .537 | .717 | 1.000 | .588 | .478 | .611 | .315 | .256 | .259 | .271 |
| F13 | .358 | .328 | .260 | .314 | .415 | .486 | .528 | .524 | .322 | .441 | .666 | .588 | 1.000 | .710 | .602 | .326 | .295 | .353 | .470 |
| F14 | .521 | .467 | .406 | .232 | .313 | .393 | .461 | .470 | .460 | .201 | .450 | .478 | .710 | 1.000 | .676 | .616 | .544 | .493 | .527 |
| F15 | .449 | .269 | .237 | .231 | .235 | .286 | .230 | .182 | .258 | .464 | .553 | .611 | .602 | .676 | 1.000 | .442 | .271 | .232 | .382 |
| F16 | .740 | .581 | .562 | .267 | .367 | .479 | .533 | .366 | .578 | .073 | .195 | .315 | .326 | .616 | .442 | 1.000 | .855 | .743 | .599 |
| F17 | .693 | .657 | .591 | .195 | .327 | .407 | .560 | .423 | .621 | .117 | .121 | .256 | .295 | .544 | .271 | .855 | 1.000 | .758 | .570 |
| F18 | .677 | .562 | .594 | .181 | .352 | .416 | .430 | .410 | .576 | .233 | .199 | .259 | .353 | .493 | .232 | .743 | .758 | 1.000 | .473 |
| F19 | .686 | .738 | .487 | .098 | .409 | .423 | .577 | .557 | .616 | .204 | .283 | .271 | .470 | .527 | .382 | .599 | .570 | .473 | 1.000 |

### 4.6.2 Communalities

Communality is the proportion of variance that each item has in common with other items. The proportion of variance that is unique to each item is then the respective item's total variance minus the communality. Table 4.14 shows the communalities. The extraction method was the principle component analysis.

**Table 4.14: Communalities**

| Challenges | Initial | Extraction |
|---|---|---|
| Lack of management support | 1.000 | .890 |
| Lack of resources | 1.000 | .822 |
| High costs | 1.000 | .794 |
| Lack of provision of maintenance in the structure of software development | 1.000 | .391 |
| Poorly designed software | 1.000 | .637 |
| Poor documentation | 1.000 | .800 |
| Lack of support of tools | 1.000 | .833 |
| Non inclusion of software in the firms policy | 1.000 | .643 |
| Lack of experienced staff | 1.000 | .714 |
| Contracting company has in house politics | 1.000 | .739 |
| Maintenance on poorly versioned software | 1.000 | .803 |
| unrealistic expectations from the contracting company | 1.000 | .720 |
| Poor coordination during software maintenance | 1.000 | .797 |
| Poor selection of software maintenance tools. | 1.000 | .804 |
| Delay in commencing of software maintenance project | 1.000 | .771 |
| Lack of proper testing after change | 1.000 | .795 |
| Unrealistic time lines | 1.000 | .802 |
| Degraded an aged software | 1.000 | .657 |
| Lack of knowledge on how to use tools | 1.000 | .634 |

### 4.6.3 Factor Extraction

Table 4.15 represents the total original variance of all factors. Principle component analysis was used to extract factors which totaled to 19. Eigen values indicate the relative importance of each factor accounting for a particular set and hence those with a small Eigen values were left out. According to Table 4.15, only 4 factors were considered significant for analysis.

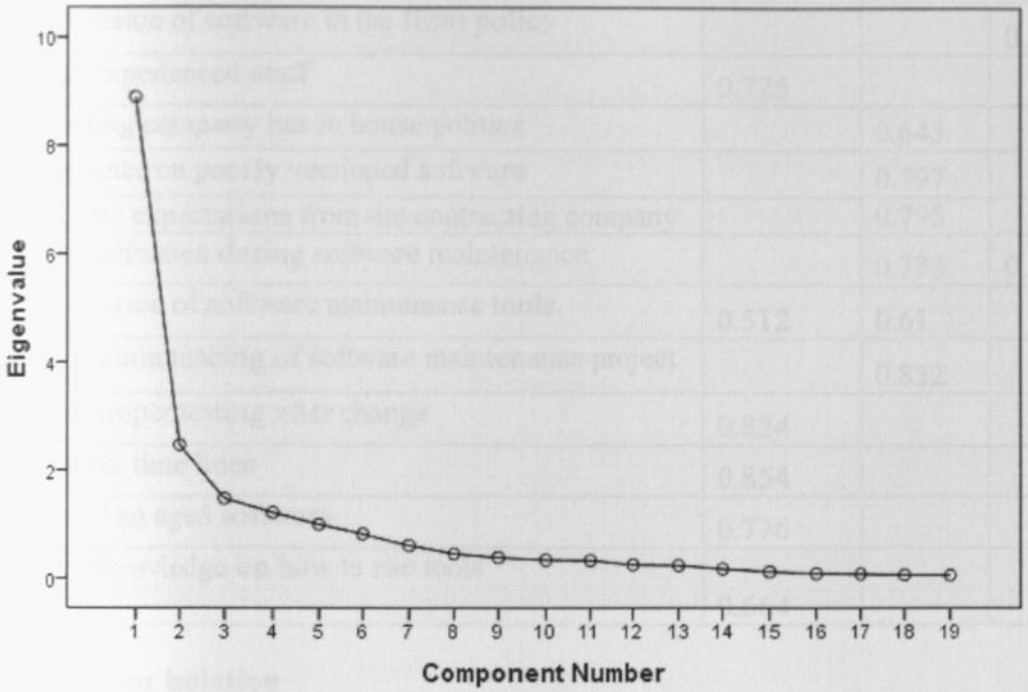
**Table 4.15: Total variance**

| Component | Initial Eigenvalues | | | Extraction Sums of Squared Loadings | | |
|---|---|---|---|---|---|---|
| | Total | % of Variance | Cumulative % | Total | % of Variance | Cumulative % |
| 1 | 8.893 | 46.806 | 46.806 | 8.893 | 46.806 | 46.806 |
| 2 | 2.467 | 12.982 | 59.789 | 2.467 | 12.982 | 59.789 |
| 3 | 1.473 | 7.752 | 67.541 | 1.473 | 7.752 | 67.541 |
| 4 | 1.213 | 6.386 | 73.927 | 1.213 | 6.386 | 73.927 |
| 5 | .991 | 5.217 | 79.144 | | | |
| 6 | .817 | 4.299 | 83.443 | | | |
| 7 | .606 | 3.191 | 86.633 | | | |
| 8 | .450 | 2.369 | 89.002 | | | |
| 9 | .380 | 2.000 | 91.002 | | | |
| 10 | .335 | 1.765 | 92.767 | | | |
| 11 | .327 | 1.721 | 94.487 | | | |
| 12 | .246 | 1.295 | 95.783 | | | |
| 13 | .234 | 1.230 | 97.013 | | | |
| 14 | .172 | .903 | 97.915 | | | |
| 15 | .119 | .627 | 98.543 | | | |
| 16 | .084 | .444 | 98.987 | | | |
| 17 | .077 | .408 | 99.394 | | | |
| 18 | .061 | .323 | 99.717 | | | |
| 19 | .054 | .283 | 100.000 | | | |

### 4.6.4 Scree plot

This is a plot of the factor Eigen values against the component numbers. According to scree plot in Figure 1, we only consider 4 factors because the curve tends to flatten from the fourth component onwards, due to relatively low factor Eigen values.

**Figure 1: Scree Plot**




### 4.6.5 Component matrix

Component matrix contains the relative Eigen values in respect of each factor. Each factor belongs to one of the 4 sets of factors extracted, and is determined by the Eigen values of the factors relative to each set. Table 4.16 shows which set of each factor falls into.

**Table 4.16: Component Matrix**

| Challenge | Component | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| Lack of management support | 0.84 | | | |
| Lack of resources | 0.798 | | | |
| High costs | 0.76 | | | 0.45 |
| Lack of provision of maintenance in the structure of software development | | | | 0.483 |
| Poorly designed software | | | 0.636 | |
| Poor documentation | | | 0.789 | |
| Lack of support of tools | | | 0.826 | |

| Challenge | Component | | | |
|---|---|---|---|---|
| | 1 | 2 | 3 | 4 |
| Non inclusion of software in the firms policy | | | 0.703 | |
| Lack of experienced staff | 0.725 | | | |
| Contracting company has in house politics | | 0.643 | | 0.558 |
| Maintenance on poorly versioned software | | 0.797 | | |
| Unrealistic expectations from the contracting company | | 0.795 | | |
| Poor coordination during software maintenance | | 0.733 | 0.48 | |
| Poor selection of software maintenance tools. | 0.512 | 0.61 | | |
| Delay in commencing of software maintenance project | | 0.832 | | |
| Lack of proper testing after change | 0.824 | | | |
| Unrealistic time lines | 0.854 | | | |
| Degraded an aged software | 0.776 | | | |
| Lack of knowledge on how to use tools | 0.664 | | | |

### 4.6.6 Factor isolation

Factor isolation involves isolating each of the variable factors and grouping them by these 4 extracted factors based on their factor loadings on each set. Table 4.17 shows the factors grouped with a minimum correlation of 0.4.

**Table 4.17: Isolation of factors**

| Factor group | Variables |
|---|---|
| Factor 1 | • Lack of management support<br>• Lack of resources<br>• High costs<br>• Lack of experienced staff<br>• Poor selection of software maintenance tools<br>• Lack of proper testing after change<br>• Unrealistic timelines<br>• Degraded an aged software<br>• Lack of knowledge on how to use tools |

| Factor group | Variables |
|---|---|
| Factor 2 | • Contracting company has in house politics<br>• Maintenance of poorly versioned software<br>• Unrealistic expectations from contracting company<br>• Poor coordination during software maintenance<br>• Poor selection of software maintenance tools<br>• Delay in commencing of software maintenance project |
| Factor 3 | • Poorly designed software<br>• Poor documentation<br>• Lack of support of software maintenance tools<br>• Non inclusion of software in the organization policy<br>• Poor coordination during software maintenance |
| Factor 4 | • High costs<br>• Software development<br>• Contracting company has in-house politics |

As shown in Table 4.17, there are four extracted group factors. Extracted group Factors 1, 2 and 3 contain the most number of variable components which are the challenges that firms face during software maintenance. Factor 1 is composed of lack of management support, lack of resources, high costs, lack of experienced staff, poor selection of software maintenance tools, lack of proper testing after change, unrealistic timelines, degraded an aged software and lack of knowledge on how to use tools.

Factor 2 comprises of the contracting company having in house politics, maintenance of poorly versioned software, unrealistic expectations from contracting company, poor coordination during software maintenance, poor selection of software maintenance tools and delay in commencing of software maintenance project. The third significant group Factor 3 comprises of poorly designed software, poor documentation, lack of support of software maintenance tools, non inclusion of software in the organization policy and poor coordination during software maintenance. Group

Factor 4 has only 3 challenges that firms face and they are high costs, software development and contracting company has in-house politics.

It is clear that most of the 19 factors listed in the questionnaire were grouped together by their correlation with each other, and brought down to a total of 4 main group Factors. The most number of Factors elements were in groups 1 to 3 whist the others fell in group 4.

# CHAPTER FIVE: CONCLUSIONS AND RECOMMENDATIONS

## 5.1 Introduction

This chapter presents the summary of the findings, conclusions and limitations as well as suggestions of further research.

## 5.2 Summary and findings

### 5.2.1 Demographic information

Demographic data was collected as a way of enriching the study and to get a profile of the respondents. This would indicate the type of population and thus validate the study. It was found that majority of the respondents were between 26 and 35 years of age. There were 22 respondents in the age group 26 and 30 years while there were 20 respondents who were in the age group 31 and 35 years. There were 2 respondents who were below 25 years and three were in the age group of 41 to 45 years.

Majority of the respondents had worked for not more than 5 years. 16 of the respondents had worked between 0 and 5 years while 7 respondents had worked between 6 years and 10 years. This implies that majority of the respondent consultancy firms had adequate experience in the ICT services that they offer.

### 5.2.2 Type of software maintenance

Findings show that 94.3% of the firms undertook corrective maintenance and this was undertaken to a moderate extent. This was closely followed by perfective maintenance. Firms undertook it at 92.5%. Perfective maintenance had a higher mean at 3.47 as compared to corrective maintenance with a mean of 3.36. Preventive maintenance was undertaken the least with only 64.2% responding that they undertook it but to a very small extent (with a mean of 2.26). This implies that firms place a high importance only on corrective and perfective maintenance even though they only used them to a moderate extent.

### 5.2.3 Extent to which firms undertook all activities in the software maintenance process

There were several activities that firms undertook while doing software maintenance. The findings revealed that firms undertook most of the activities to a moderate extent. The mean ranged from 2.42 and 3.36. This implies that majority of the firms undertook the activities in the software maintenance process to a moderate extent. The activities that were commonly undertaken included developing of maintenance plans and implementing them, developing, analyzing and documenting a request, identifying items to be modified, developing test scenarios and finally assessing the integrity of the systems after implementation. Other activities that were undertaken to a small extent included creating and implementing procedures for tracking requests, developing of migration plans and finally developing of system retirement plans.

### 5.2.4 Extent to which firms use software maintenance tools

The respondents used software maintenance tools while undertaking software maintenance. They also indicated what they considered to be important in the selection of the maintenance tools. The findings show that firms only use software maintenance tools to a very small extent as the mean only lies between 2.04 and 2.92. The testing tool is the tool that has a mean close to 3 implying that it is used to a moderate extent. Respondents also indicated the level of factors that they thought were important when selecting software maintenance tools. The findings show that the different factors that were listed were moderately important. The cost of acquisition of maintenance tools had the highest mean of 3.68 meaning that it was a very important factor to the firm. The least important factor was the sustainability of the maintenance tool which had a mean of 3.36 (implying that it was somewhat important to the firm).

### 5.2.5 The challenges faced by firms in software maintenance.

The last objective was to determine the challenges that firms have when they are undertaking software maintenance. Factor analysis was used to generate the cluster of challenges. The challenges that were considered to a very large extent were lack of management support, lack of resources, high costs during maintenance, lack of experienced staff, poor selection of software maintenance tools, lack of proper testing after change, unrealistic timelines, degraded and aged software and lack of knowledge on how to use tools.

Other challenges that firms identified with to a large extent were: In-house politics of contracting company; maintenance of poorly versioned software; unrealistic expectations from contracting company; poor coordination during software maintenance; poor selection of software maintenance tools and delay in commencing of software maintenance project.

Firms also faced software maintenance challenges to a moderate extent. The challenges comprised of poorly designed software; poor documentation; lack of support of software maintenance tools; non inclusion of software in the organization policy and poor coordination during software maintenance. Other challenges were only felt to a small extent and they included high costs in the maintenance of software; software development and contracting company having in-house politics.

## 5.3 Conclusion

The main findings of this study show that firms only undertook software maintenance to a moderate extent. This was concluded based on the responses from the respondents on the practice of software maintenance. Majority of the firms undertook software maintenance to a moderate extent. These firms on the other hand performed most of the activities in the software maintenance process though they did not fully utilize software maintenance tools. Majority of the firms however used the test scenario tools. The firms experienced major challenges that included lack of management support; lack of resources; high costs during maintenance; lack of experienced staff; poor selection of software maintenance tools; lack of proper testing after change; unrealistic timelines; degraded and aged software; and lack of knowledge on how to use tools.

## 5.4 Limitations of the study

This study was limited to a small sample size that included ICT firms in Nairobi. Although 86 firms were targeted, only 53 firms responded appropriately. The respondents found the questionnaire too long. The questionnaire could have been shortened but this would have reduced the findings being sought. Lack of enough literature material on software maintenance and computer maintenance in general was also another limitation encountered.

## 5.5 Areas for further research

This study has highlighted the extent to which firms undertook different types of software maintenance, activities in the maintenance process, maintenance tools used as well the challenges that they encounter. However better results can be achieved if a case study can be done with a small number of collaborating firms.

Research should be carried to out to establish whether each of the challenges in the study affects the practice of software maintenance.

More in-depth studies should be carried out to find out exactly why firms choose particular types of software maintenance over others as well as reasons why some of the activities are not undertaken in software maintenance process.

# REFERENCES


April, A., Haye J. H., Dumke & Abran A. (2004), **"Software Maintenance Maturity Model (SMmm): The software maintenance process model"** New York: John Wiley & Sons Ltd

Arnold, R. S. and Bohner, S. A. (1993), "**Impact Analysis – Toward a Framework for Comparison**", Proceedings of the Conference on Software Maintenance, Montreal, Canada, IEEE Computer Society Press, Los Alamitos, CA, pp. 292-301.

Beath, C. N., Swanson, E. B. (1989), **"Maintaining Information Systems in Organizations"**, John Wiley & Sons, New York, NY.

Bennett, K.H. (2000), "**Software Maintenance: A Tutorial. In Software Engineering"**, edited by Dorfman and Thayer. IEEE Computer Society Press: Los Alamitos, CA, 289-303 pp.

Bohner, S.A., Arnold R.S., (1996), **"Software Change Impact Analysis"**, IEEE Computer Soc. Press, Los Alamitos, CA.

Bowen, J. P., Breuer P.T and Lano K.C. (1993), **A compendium of formal techniques for software maintenance.** Oxford university computing laboratory programming research group 11 keble Road, Oxford OX 1 3QD, UK.

Brooks, F. P. Jr. (1987), "**No Silver Bullet**", IEEE Computer, 20(4):10-19

Burch, J. G. and Grudnitski G. (1986), **Information systems: Theory and practice (4th ed.), John Wiley and Sons, New York.**

Canfora G. and Cimitile A. (2000), "**Software Maintenance**". Palazzo Bosco Lucarelli, Piazza Roma, 82100, Benevento Italy

Canfora, G., Cimitile, A. and Munro, M. (1996), "An Improved Algorithm for Identifying Objects in Code", Software – Practice and Experience, 26(1):25-48. Ciborra C. (2002), Labyrinths of Information. Oxford, Oxford University Press.

Chepchieng, K.H. (2006), "**A survey of software testing processes used by software developers in Kenya"**, University of Nairobi

Corbi, T. A. (1989), "**Program Understanding: Challenge for the 1990s**", IBM System Journal, 28(2):294-306.

Dekleva, S.M., (1992) "**Delphi study of software maintenance problems. Proceedings of 1992 IEEE Conference on Software Maintenance**", ICSM, IEEE Computer Society: Orlando, Florida,10-17.

Dorfman, M., Thayer R.H (1997), "**software engineering**" IEEE computer society press

Erdil, K., Finn E., Keating K., Meattle J., Park S. and Yoon D.(2008), **software maintenance as part of the software life cycle.** www.hepguru.com/maintenance/final121603.pdf. Dated 14/03/2008

Fjeldstad, R.K., Hamlen W.T., (1982), "**Application Program Maintenance Study**": IEEE Computer Society Press, Los Alamitos, CA, pp. 13 – 30

Fyson, M. J. and Boldyreff, C. (1998), "**Using Application Understanding to Support Impact Analysis**", Journal of Software Maintenance – Research and Practice, 10(2):93-110.

Hartmann, J. and Robson, D. J. (1990), "**Techniques for Selective Revalidation**", IEEE Software, 16(1):31-38.

Hobatr, C. and Malloy, B., (2001). "**Using OCL-Queries for Debugging C++".** IEEE, 839-840.

Hirabad, J. S., Lethbridge T.C., and Matwin S. (2001), **Supporting software maintenance by mining software update records**. **Proceedings IEEE international conference on software maintenance ICSM** Florence, Italy pages 22-31,

IEEE. (1993), **IEEE Standard for Software Maintenance**. IEEE Std 1219- Electrical and Electronics Engineers, inc., New York, NY.

ISO/IEC 9126, (1991), "**Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for Their Use**", Geneva, Switzerland.

John, B., Liam, O., Dennis, S. (2001) **DoD Software Migration Planning**, Carnegie Mellon University.

Leung, H. K. N. and White, L. J. (1990), "**Insights into Regression Testing**", Proceedings of the Conference on Software Maintenance, Miami, Florida, IEEE Computer Society Press, pp. 60-69.

Livadas, P. E. and Small, D. T. (1994), "**Understanding Code Containing Preprocessor Constructs**", Proceedings of the 3rd Workshop on Program Comprehension,

Washington, DC, IEEE Computer Society Press, Los Alamitos, CA, pp. 89-97.

Kajko-Mattsson, M., Forssande S., and Allsonn U. (2001), **Corrective Maintenance Maturity Model Maintainers Education and Training. Proceedings of the 23rd international conference on software engineering.** Toronto Ontario

**Kenya telephone directory and yellow pages (2007)** Kenya postal directories limited, Nairobi

Laudon, K.C. and Laudon, J. P. (2006) **Management Information Systems: Managing the Digital Firm**, Prentice-Hall of India Private Limited, New Delhi.

Laudon, K. C. and Laudon, J. P. (2001), **Essentials of management information systems: (4th ed.),** prentice hall, New Jersey

Lehman, M. M., (1980) **"On understanding Laws, evolution and conversation in the large program lifecycle"**. Journal of Software & Systems, vol. 1, pp. 213 – 221

Mwaniki, W.K. (2004), **"Information systems projects in kenya: A study of user involvement",** University of Nairobi, Faculty of Commerce

Nation Business Directory (2006)

O'Brien, J. A, (2002), **Introduction to Information Systems. Essentials for Internet-worked E-business enterprise** (10th ed.), McGraw-Hill inc., New Delhi.

O'Neal, J. S. and Carver, L.C. (2001), **Analyzing the impact of changing requirements. Proceedings IEEE international conference on software maintenance ICSM (2001)** Florence, Italy pages 22-31.

Pfleeger, S.L. (1998), **"Software engineering-theory and practice"** prentice hall.

Post, G., and Anderson, D., (2006). ***"Management information systems: Solving business problems with information technology"***. (4th ed.). New York: McGraw-Hill Irwin.

Pigoski, T. M. (1997), **"Practical Software Maintenance – Best Practices for Managing Your Software Investment"**, John Wiley & Sons, New York, NY,

Queille, J. P., Voidrot, J. F., Wilde, N. and Munro M. (1994), **"The Impact Analysis Task in Software Maintenance**: A Model and a Case Study", Proceedings of the International

Conference on Software Maintenance, Victoria, Canada, IEEE Computer Society Press, Los Alamitos, CA, pp. 234-242.

Rajlich, V. (1994), "**Program Reading and Comprehension**", Proceedings of the Summer School on Engineering of Existing Software, Monopoli, Bari, Italy, Giuseppe Laterza Editore, Bari, Italy, pp. 161-178.

Robert L. G., (2006), **Software Conflict 2.0: The Art and Science of Software Engineering**. http://www.developerdotstar.com/mag/articles/PDF/DevDotStar_Glass_Maintenance.pdf

Reynolds, G. W. (1995), **Information systems for mangers (3rd ed.),** west publishing company

Rood, S. C. (1996) , **Computer hardware maintenance. An IS/IT managers guide.** , Domeniks S.D., Butter Worth Heinemann.

Rothermel, G. and Harrold, M. J. (1994), "**A Framework for Evaluating Regression Test Selection Techniques**", Proceedings of the 16th International Conference on Software Engineering, Sorrento, Italy, IEEE Computer Society Press, CA, pp. 201-210.

Schneberger, S. L. (1997), **Client/server software maintenance**. McGraw-Hill, New York, NY.

Schneidewind, N. F. (1987), "**The State of Software Maintenance**", IEEE Transactions on software Engineering, SE-13(3):303-310.

Schultheis, R. and Sumner, M. (1995), **Management Information systems: The manager's view. (3rd ed.), Richard D. Irwin, inc., New Delhi.**

Shirabad, J. S, Lethbridge, T. C and Matwin, S. (2001) **supporting software maintenance by mining software update records.** Proceedings IEEE international conference on software maintenance (ICSM 2001) Florence, Italy, pages 22-31 November 2001

Sommerville, I. (2001) **Software Engineering**, (6th ed.), Addison-Wesley publishers limited, Person Education limited.

Standish, T. A. (1984), "**An Essay on Software Reuse**", IEEE Transactions on Software Engineering, SE-10(5):494-497.

Stark, G. and Oman, P. W. (1997), **Software maintenance management strategies. Observations from the field.** Journal of software maintenance pages 356-378

Takang, A.A., and Grubb, P.A. (1996), **Software Maintenance Concepts and Practice**. Thompson Computer Press London, UK.

Taneja V (2006), **"A survey of application of information and communication technology for competitive advantage of firms listed at the Nairobi Stock Exchange"** University of Nairobi

Yau, S. S. and Colleferro, J. S. (1980), **"Some Stability Measures for Software Maintenance"**, IEEE Transactions on Software Engineering, SE-6(6):545-552.

Vliet, H. (2000), **Software Engineering: Principles and Practices**, 2nd Edition. John Wiley and Sons, West Sussex, England

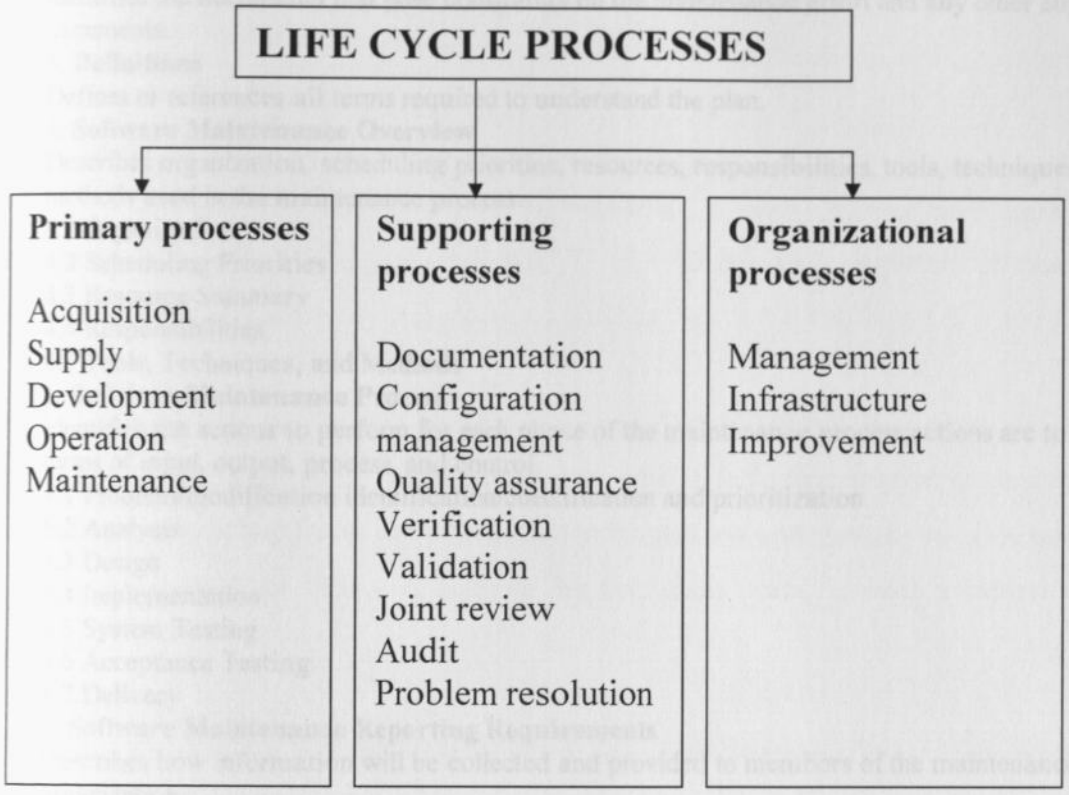
Hung, V. (2008), **"Software maintenance",** http://www.vocw.edu.vn/content/m10073/latest last updated Apr 26, 2008 5:11 pm.

Vigder, M. R. (1999)**, Building Maintainable Component-Based Systems**, Carnegie Mellon Software Engineering Institute. http://www.sei.cmu.edu/icse99/papers/38/38.htm

Zhou, S., Zedan, H. and Cau, A. (2001) **A framework of analyzing the effect of change in legacy code** Proceedings IEEE international conference on software maintenance (ICSM 99) , Oxford, England pages 411-420 , August 1999

# APPENDICES

## Appendix I: The ISO-12207 life cycle processes




**Source:** Canfora, G., Cimitile, A.: **Software Maintenance.** Palazzo Bosco Lucarelli, Piazza Roma, 82100, Benevento Italy, 2000

## Appendix II: - Sample Maintenance Plan

1. **Introduction**
Describes the purpose, goals, and scope of the software maintenance effort; determines deviations from the standard.
2. **References**
Identifies the documents that pose constraints on the maintenance effort and any other supporting documents.
3. **Definitions**
Defines or references all terms required to understand the plan.
4. **Software Maintenance Overview**
Describes organization, scheduling priorities, resources, responsibilities, tools, techniques, and methods used in the maintenance process.
4.1 Organization
4.2 Scheduling Priorities
4.3 Resource Summary
4.4 Responsibilities
4.5 Tools, Techniques, and Methods
5. **Software Maintenance Process**
Identifies the actions to perform for each phase of the maintenance process;actions are to be defined in terms of input, output, process, and control.
5.1 Problem/modification identification/classification and prioritization
5.2 Analysis
5.3 Design
5.4 Implementation
5.5 System Testing
5.6 Acceptance Testing
5.7 Delivery
6. **Software Maintenance Reporting Requirements**
Describes how information will be collected and provided to members of the maintenance organization.
7. **Software Maintenance Administrative Requirements**
Describes the standards, practices and rules for anomaly resolution and reporting.
7.1 Anomaly Resolution and Reporting
7.2 Deviation Policy
7.3 Control Procedures
7.4 Standards, Practices, and Conventions
7.5 Performance Tracking
7.6 Quality Control of Plan
8. **Software Maintenance Documentation Requirements**
Describes the procedures to be followed in recording and presenting the outputs of the maintenance process.

Source: IEEE Std. 1219-1998, "Standard for Software Maintenance", IEEE Computer Society Press, Los Alamitos, CA, 1998.

## Appendix III: Introduction Letter

**Dear respondent,**

TO WHOM IT MAY CONCERN

**A survey on software maintenance practice in Kenya**

I am a postgraduate student in the school of business, University of Nairobi, Pursuing masters in Business Administration degree program. I am undertaking a research on the practice of maintenance of software tools and processes used as well as the challenges that are encountered.

You have been selected as one of the respondent. I therefore kindly request you to fill in the matched questionnaire. The information is needed purely for academic purposes and will therefore be treated with utmost confidentiality. Your name and that of your firm will not appear any where in the final report. A copy of the final report can be made available to you upon request.

If you require any further information, do not hesitate to contact me by email rosy.macharia@gmail.com.

Thank you in advance

Yours faithfully,

Rosemary Macharia

## Appendix IV: Questionnaire

**Questionnaire No:** _____

**A SURVEY OF SOFTWARE MAINTENANCE PRACTICE IN KENYA**

(Kindly respond to this questionnaire with reference to the *firm you are working with*)

**SECTION A: DEMOGRAPHIC INFORMATION**

**Respondent's profile**

1. Which of the following best describes your role in the firm you work for?
   a. Chief Executive Officer..... [ ]
   b. Chief Information Officer..... [ ]
   c. Projects Manager ..... [ ]
   d. Quality Assurance Consultant..... [ ]
   e. Software developer / programmer..... [ ]
   f. Systems Analyst ..... [ ]
   g. Functional Consultant ..... [ ]
   h. Training Consultant..... [ ]
   i. Others, specify _____
2. How many years have you worked in ICT consultancy? _____
3. Kindly indicate by ticking appropriately your highest level of education
   a. O-level (Form IV)..... [ ]
   b. A -level (Form V, VI)..... [ ]
   c. Bachelors Degree..... [ ]
   d. Masters Degree..... [ ]
   e. PhD Degree..... [ ]
   f. Others, specify _____
4. Indicate by ticking appropriately your gender : Male [ ] Female [ ]
5. Indicate by way ticking the range of your age in the following range of ages :
   a. Below 25 years..... [ ]
   b. 26 --30 years..... [ ]
   c. 31 --35 years..... [ ]
   d. 36 --40 years..... [ ]

e. 41 -- 45 years..... [ ]

f. 46 -- 50 years..... [ ]

g. 51--55 years..... [ ]

h. Above 55 years..... [ ]

**Firm Profile**

6. For how many years has the firm been in operation? _____
7. How many branches does the firm have? _____
8. Indicate by ticking appropriately the ownership of the firm.
    a. Locally owned..... [ ]
    b. Foreign owned..... [ ]
    c. Both (Locally and Foreign owned)..... [ ]
9. What is the approximate number of employees in the firm? _____
10. What is the average annual turnover of the firm in KSH? _____
11. Indicate by ticking appropriately the IS/IT consultancy services the firm offers
    a. IS development..... [ ]
    b. IS Audit..... [ ]
    c. Software maintenance..... [ ]
    d. Hardware maintenance..... [ ]
    e. System integration.....[ ]
    f. Enterprise security.....[ ]
    g. IT architect services.....[ ]
    h. others, Specify _____

## SECTION B: TYPES OF SOFTWARE MAINTENANCE.

12. The following is a listing of some of computer software maintenance types. Kindly indicate what software maintenance your firm undertakes by ticking appropriately against each type of maintenance.

| | | YES | NO |
|---|---|---|---|
| 1 | Corrective maintenance (*deals with the repair of faults or defects found*) | | |
| 2 | Adaptive maintenance (*consists of adapting software to changes in the environment*) | | |
| 3 | Perfective maintenance (*deals with accommodating to new or changed user requirements*) | | |
| 4 | Preventive maintenance (*deals with increasing the system's maintainability, such as updating documentation, adding comments, and improving the modular structure of the system*) | | |

13. Below is a listing of the software maintenance types. Kindly indicate by ticking the extent to which your firm undertakes each type of maintenance.

| | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | | **No extent at all** | **Small extent** | **Moderate extent** | **Large extent** | **Very large extent** |
| 1 | Corrective maintenance | | | | | |
| 2 | Adaptive maintenance | | | | | |
| 3 | Perfective maintenance | | | | | |
| 4 | Preventive maintenance | | | | | |

## SECTION C: SOFTWARE MAINTENANCE PROCESS

14. Do firms to which you have provided consultancy on software maintenance set policy on software maintenance? YES [ ] NO [ ]
15. Do the firms have plans for software maintenance? YES [ ] NO [ ]
16. Which is the most common standard that firms to which you undertake software maintenance?
    a. Internally defined rules and procedures [ ]
    b. Locally set standards [ ]
    c. International standards (ISO/IEEE) [ ]

d. Others specify._____

17. The following is a listing of software maintenance process activities. Please rate (by ticking appropriately) the extent to which your firm undertakes the activities in the process.

| | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | | **No extent at all** | **Small extent** | **Mode rate extent** | **Large extent** | **Very large extent** |
| 1 | Development of computer software maintenance plan | | | | | |
| 2 | Implementation of computer software maintenance plan. | | | | | |
| 3 | Creating procedures for receiving, recording, and tracking maintenance requests | | | | | |
| 4 | Implementation of procedures for receiving, recording, and tracking maintenance requests | | | | | |
| 5 | Analysis of maintenance request as well as determine its scope in term of size, costs, and time required. | | | | | |
| 6 | Development and documentation of alternatives for change implementation as specified | | | | | |
| 7 | Identification of the items that need to be changed and the invocation of the development process to actually implement the changes. | | | | | |
| 8 | Test scenarios developed and implemented to ensure that the new/modified requirements are completely and correctly implemented. | | | | | |
| 9 | Assessing the integrity of the modified system | | | | | |
| 10 | Development of migration plans | | | | | |
| 11 | Development of a retirement plan | | | | | |
| 12 | notification to users on the change to the new system | | | | | |
| 13 | Others specify……<br>……<br>…… | | | | | |

**SECTION D: TOOLS USED IN SOFTWARE MAINTENANCE.**

18. The following is a listing of some of computer software maintenance tools. Kindly indicate by ticking appropriately the extent to which they are used for maintenance of computer software in your firm.

| | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | | **No extent at all** | **Small extent** | **Moderate extent** | **Large extent** | **Very large extent** |
| 1 | Slicing tool (*tool that helps the programmers select and view only the parts of the program that are affected by the changes*) | | | | | |
| 2 | Dynamic analyzer tool (*tool used to analyze the program while it is executing*) | | | | | |
| 3 | Cross-reference tool (*tools that help the user focus on the parts that are affected by the change.)* | | | | | |
| 4 | Dependency analyzer tool (*tools that assist the maintainer to analyze and understand the interrelationships between entities in a program*) | | | | | |
| 5 | Test simulator tool (*tools that help the maintainer to test the effects of the change in a controlled environment before implementing the change on the actual system*) | | | | | |
| 6 | Configuration management and version control tools (*Tools used to keep a history of the files so that versions can be tracked and the programmer can keep track of the file changes.)* | | | | | |
| 7 | Others , specify<br>.....<br>.....<br>..... | | | | | |

19. The following is a list of factors to consider necessary when selecting software maintenance tools. Please indicate by ticking appropriately the level of importance that your firm attaches to each of them.

| | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | | **Not important** | **Somewhat important** | **Important** | **Very important** | **Extremely important** |
| 1 | Ease of use of the maintenance tool | | | | | |
| 2 | Capabilities of the maintenance tool | | | | | |
| 3 | Availability of the tool | | | | | |
| 4 | Stability of the vendor of the maintenance tool in order to offer support. | | | | | |
| 5 | Availability of trained staff to use the tool for maintenance purposes | | | | | |
| 6 | Extent to which the maintenance tool is used in Kenya | | | | | |
| 7 | Sustainability of the tool | | | | | |
| 8 | Extent to which the tool can get data in and out in a standard format | | | | | |
| 9 | The cost of acquisition of the tool verses the benefit | | | | | |
| 10 | Reliability of the tool. | | | | | |
| 11 | Others, Specify<br>.....<br>.....<br>..... | | | | | |

## SECTION C: CHALLENGES OF SOFTWARE MAINTENANCE

20. The following is a listing of challenges faced in computer software maintenance. Kindly indicate by ticking appropriately the extent to which each is faced by your firm in software maintenance.

| | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | | **No extent at all** | **Small extent** | **Moderate extent** | **Large extent** | **Very large extent** |
| 1 | Lack of management support for software maintenance. | | | | | |
| 2 | Lack of software maintenance resources. | | | | | |
| 3 | High costs when undertaking maintenance. | | | | | |
| 4 | Lack of provision for maintenance in the structure of software development. | | | | | |
| 5 | Purchase or development of poorly designed computer software | | | | | |
| 6 | Lack of understanding and poor documentation of computer systems | | | | | |
| 7 | Lack of support of software maintenance tools | | | | | |
| 8 | Non- inclusion of software maintenance in the organizations policy | | | | | |
| 9 | Lack of experienced staff to undertake maintenance of software | | | | | |
| 10 | Contracting company has in house politics | | | | | |
| 11 | Software maintenance on poorly versioned systems | | | | | |
| 12 | Unrealistic expectations | | | | | |
| 13 | Poor coordination during software maintenance. | | | | | |
| 14 | Poor selection of software maintenance tools | | | | | |
| 15 | Delay in commencing of software maintenance project. | | | | | |

| | | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| | | **No extent at all** | **Small extent** | **Moderate extent** | **Large extent** | **Very large extent** |
| 16 | Lack of proper testing after a change is made to the computer software | | | | | |
| 17 | Unrealistic timeline to undertake software maintenance. | | | | | |
| 18 | Degraded and aged software | | | | | |
| 19 | Lack of understanding on how to use software maintenance tools | | | | | |
| 20 | Others specify<br>.....<br>.....<br>..... | | | | | |

THE END

***Thank you for taking your time to fill this questionnaire***

## Appendix V: List of Target Respondents

1. Aequitas technologies
2. Afrosoft technologies
3. Alien technologies
4. Alphatech microsystems
5. Alphax infosys ltd
6. Amarco (kenya) ltd
7. Ankem computer services
8. Archways technologies ltd
9. Aren software systems
10. Asa computerised information
11. Ascent technologies
12. Assured agencies
13. Automated software systems
14. Business connections and technologies
15. Capital compter systems
16. Capital technologies kenya
17. Computer capacities and innovations
18. Computer planet
19. Computron systems ltd
20. Dac-net communications
21. Dataflex computer consultation
22. Dee Dee computers plus
23. Desktop micro services ltd
24. Digital systems solutions
25. East africa software
26. Emerging technologies consultations
27. Enterprise software solutions
28. Excel integrated solutions
29. Executive support consultants
30. Executive information systems consultations
31. Executive support consultations
32. Extreme computer engineering
33. Fabit automated systems
34. Fasttech solutions ltd
35. Fintech limited
36. Fishnet technologis
37. Footman walker ltd
38. Houston technologies
39. HP East Africa
40. IBIS systems ltd
41. Infoline consultants
42. Interpay limited
43. Inventech solutions ltd
44. IQplus (kenya) ltd

45. Izon future systems
46. Jawchan software services
47. Ken data systems ltd
48. Kingsway business systems
49. Matrix group
50. Micro expert limited
51. Milestone software ltd
52. Nanosoft technologies ltd
53. Neptune software
54. Next technologies
55. Nextech software ltd
56. Ojanga Asego systems ltd
57. Online computer systems
58. Open view business systems
59. Orange works
60. Westend business solutions
61. Pentium technologies
62. Pinnacle relational database system
63. Precision software consultants
64. Premier software ltd
65. Prosoft consultants
66. SAP East Africa
67. Sera software (EA) ltd
68. Simple computers
69. Skyweb technologies ltd
70. Softcom business solutions
71. Software applications
72. Software applications ltd
73. Software associates ltd
74. Softwise (kenya) ltd
75. Solution for information systems Ltd
76. Soluziana systems
77. Stack systems ltd
78. Symphony
79. Today computers
80. Todays online ltd
81. Unitek computer services
82. Vega software ltd
83. Vision technologies
84. Web engineering limited
85. Zodiac systems ltd