CONTENT DEVELOPMENT FOR E-BUSINESS SYSTEMS: A CASE STUDY OF FOSS TOOLS IN CONTENT DEVELOPMENT FOR KAMPALA INTERNATIONAL UNIVERSITY - ICT

BY

MARGARET KAREYO

BIS (KIU), CCNA (MUK)

SUPERVISED BY

MRS. OUMA PASCHALIA NDUNGWA

A research report submitted to the

School of Postgraduate Studies

in partial fulfillment for the award of the degree of

Master of Business Administration -Information Technology

Of

Kampala International University

July 2008

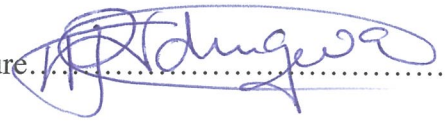
DECLARATION

I **Kareyo Margaret**, hereby declare that this research presented here is my original work. Where any part of this work has been obtained from the work of any author, it is duly acknowledged. I also wish to declare that it has never been submitted to any academic institution for the award of a degree or the equivalent. Any similarity is coincidence of ideas.

Signature..... Date...17th Oct, 2008...

APPROVAL

This is to certify that this thesis has been done under my supervision and is now ready for submission to the board of examiners with my approval.

Signature.....  ..... Date.....17/10/08.....

Ouma Pascalia Ndungwa (MRs)
Supervisor

DEDICATION

The Prince of peace is worth more than this. This work is a result of the Grace granted unto me by Yahweh. Truly Ebenezer this far have You ushered me. May the fruits of this work be a constant reminder to all who shall read it that You are the present help unto all who call upon You in times of trouble. For ever be blessed.

ACKNOWLEDGEMENTS

My deepest thanks to:

Colleagues: Wabule, Ssempebwa, Sarah Rachael, Murungi, Kimani, Lance,Amongi and Ambrose. Your ideas were profound

They don't come any better

The church family (The Gaba cell and Rosette crew):

Your prayers, encouragement and support were invaluable. The fun we have keeps me going.

Mr.Muyira Kiwanuka:

You are one of the finest guardians on earth; you gave me a life-time stepping stone.

Dad and Mum:

Could I ever enjoy the challenges of life without your involvement and prayers?

Luswata Jacent and Otimong Vincent:

What would I do without your eleventh hour aid and twenty four hour friendship?

Mrs. Ouma Maureen and Mr. Bada Joseph

Am grateful for the technical advice and the time spared off your tight schedules to read through my script.

The MBA Class

Amigos queridos

Mr. Wafula. V. Innocent

If a star fell each time I thought of you, the sky would be empty. Thanks for acting the wife through my long hours of work.

LIST OF ACRONYMS

ANSI	American National Standards Institute
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Naming System
EDI	Electronic Data Interchange
FLOSS	Free Libre Open Source Software
FOSS	Free/Open Source Software
FOSSFA	Free Software and Open Source Foundation for Africa
FTP	File Transfer Protocol
GPL	General Public License
GPL GNU	General Public License (the most common OSS/FS license)
HTTP	Hyper Text Transfer Protocol
ICT	Information and Communications Technology
IDC	International Data Corp
IETF	Internet Engineering Task Force
IIS	Internet Information Server
NGO	Non- Governmental Organizations
NOAA	National Oceanographic and Atmospheric Administration
NUMA	Non-uniform memory architecture
OS	Operating System
OSes	Operating Systems
OSI	Open Source Initiative
OSS	Open Source Software

OSS/FS	Open Source Software/Free Software
PC	Personal Computer
PHP	Hypertext Preprocessor
RAND	Reasonable and Non Discriminatory licensing
SMP	Symmetric Multiprocessing
SSL	Secure Sockets Layer

LIST OF TABLES

LIST OF FIGURES

GLOSSARY

Digital divide is the fact that different peoples, cultures, and areas of the world or within a nation do not have the same access to information and telecommunications technologies.

Electronic commerce is commerce accelerated and enhanced by IT, in particular, the Internet. It enables customers, consumers, and companies to form powerful new relationships that would not be possible without the enabling technologies.

Information-literate knowledge worker is a knowledge worker who can define what information is needed, knows how and where to obtain that information, understands the meaning of the information once received, and can act appropriately based on the information to help the organization achieve its greatest advantage.

Internet is a vast network of computers that connects millions of people all over the world.

Operating system software is system software that controls your application software and manages how your hardware devices work together

Pirated software is copyrighted software that is copied and distributed without permission of the owner.

Software is the set of instructions that your hardware executes to carry out a particular task.

Telecommuting is the use of communications technologies (such as the Internet) to work in a place other than a central location.

Business model is an architecture for the product, service and information flows, including a description of the various business actors and their roles; and a description of the potential benefits for the various business actors; and a description of the sources of revenue". A business model should aim to generate revenue and be profitable.

ABSTRACT

Proprietary software are programs whose licenses give the user permission to run them but are not allowed to share, alter or even redistribute them while Open Source Software/Free Software (OSS/FS), also abbreviated as FOSS (Free Open Source Software) licenses give users the freedom to run the program for any purpose, study, modify, and redistribute copies of either the original or modified program without having to pay royalties to previous developers. This research employed both an explanatory and experimental approach to assess the maturity of FOSS by analyzing its reliability, security, usability and availability, as compared to Proprietary Software. The users can use the discussed results to compare software and thereby purchase appropriate programs for their organizations.This research reveals that OSS/FOSS and proprietary performances depend on the context and situation. When it comes to this parameter, it is hard for one to conclusively say that FOSS or proprietary software is better. The research, however, reveals that FOSS is more reliable and more secure than the proprietary software.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.0 General Introduction

One of the hottest topics today in the business world is electronic commerce. Its all about performing commerce electronically, and includes giving customers the ability to order products and services over the Internet and connecting with other organizations electronically to exchange such information as sales invoices, and purchase orders. It even includes moving money electronically, that is; electronic funds transfer for organizational use and electronic or digital cash for personal use.

The idea of electronic commerce sounds exciting, however, its availability and affordability to the business circles more so in the developing countries is pretty questionable. Accessibility and of the required software becomes a major determining factor in the success of a business that opts for e-commerce. Widely offered on the IT market are proprietary and free open source software, about which people in the business world know little or have no knowledge at all.

Proprietary software also known as 'closed' are programs whose licenses give users permission to run them but one is not allowed to share, alter or even redistribute them without permission. It's illegal to make copies and distribute proprietary software without paying additional licensing fees. Examples of "closed" include MS-Windows Vista, XP, I info server, Sun Solaris, Mac OS, UNIX and many others. On the other hand Free/Open Source Software (FOSS) is software that is availed to the user along with the source code as a distinctive feature. It is often available at no cost and users are allowed to redistribute it. And if they so wish, they can study the source code and modify it to suit their needs. Examples are Mozilla Firefox and Apache Web Server.

Therefore, this research aimed at analyzing the accessibility, usability, reliability and security of the software on market so as to enlighten the business community. Hence, giving

them an opportunity to make wise choices as what software could serve the purpose of excellent profit yielding and yet still keep their customers smiling.

1.1 Background of the study

Free Open Source Software is a recent phenomenon that has the potential to revolutionize the software industry. It has already gained a strong foothold in the server software segment, with a leading market share worldwide in some software categories. It is also gaining ground in desktop applications.

Interest in FOSS is growing globally, particularly in developing countries. Governments are considering policies to promote its use, businesses are recognizing its potential and various other sectors are giving increasing attention to the opportunity for the localization it presents.

1.1.1 A Brief History of Free/Open Source Software (FOSS)

Free or non-proprietary software has existed since the invention of the first computers in the mid-1940s, and for many years it was the norm.

The FOSS movement dates back to almost the very beginning of the computer industry, although it was not then formally defined or conceptualized. It was only in the late 1970s and early 1980s that the sharing of software began to really come in conflict with proprietary software.

One of the earlier references to proprietary software was made by William H. Gates III in his now-famous "An Open Letter to Hobbyists. In this letter, dated 3rd February 1976, he rails against the prevailing culture of software sharing:

Why is this? As the majority of hobbyists must be aware, most of you steal your software. Hardware must be paid for, but software is something to share. Who cares if the people who worked on it get paid?

Proprietary software would gain momentum over the years. At the pioneering MIT Artificial Intelligence Lab in the early 1980s, a company called Symbolics was formed and took what was freely available code (the *LISP programming language*) and made it proprietary.

1.1.2 FOSS in developing countries

For many people in developing countries, commercial software packages are not an option because they are expensive, do not come in local languages and cannot be shared. Yet the fact remains that software and other information and communication technologies (ICTs) is becoming increasingly crucial to social and economic development.

In today's world, the development of nations depends not only on the free flow of information, but also on people's ability to control and adapt the information as they see fit. This latter need is particularly important when it comes to software packages, and is globally acknowledged by governments, local authorities, educational institutions and civil society.

However, the reality is that commercial software companies continue to have a proprietary hold over software packages sold and used in developing countries, and restrict the way the software can be used.

This concern has led to an international movement for software packages that can be bought cheaply, shared freely and changed around and developed as the users want.

Advocates of this movement, known as Free and Open Source Software, or FOSS, say the tools of information and communication should be in the public domain so that they can generate greater knowledge sharing in society.

Commercial software companies counter this by saying that they spend huge sums of money on the research and development of software, and that, just like any other commercial venture, they need to restrict usage in order to recoup the investments. Anyway, the question remains of how various opposing views can be reconciled in the interests of millions of potential software users in the developing world.

1.1.3 Foss in Africa

Despite its high priorities, Africa can investigate how to better use the opportunities presented by the emergence of Free and Open Source Software in the context of limited financial resources and capacities. Open source provides a way for Africans to help themselves, not to wait for the first world, but to get up and do it themselves.

This will help African countries leapfrog into the information age through reduced costs, less dependency on imported technology and its ability to be customized to local languages. Moreover, by giving users access to its inner workings, open source could stimulate the local software industry. The open source philosophy lends itself to making technology available to the masses.

1.1.4 Africa's Technology Development and the Digital Bridge

Rapid expansion of the Internet holds substantial promise for Africa, because it can benefit greatly from the Internet's communication and information delivery capabilities to help meet several needs. The accelerating transition of information to electronic media is making information resources of the world available to an increasingly global audience through the Internet. Africa has much to gain from that revolution in communication and information access.

In contrast to the situation in the developed world, where transport and communications infrastructures for delivery of both physical goods and information services are well established, the alternatives available within Africa are generally slow, expensive, or nonexistent. The communications and information delivery capability of the Internet serves all sectors of society. The areas of education, health, social policy, commerce and trade, government, agriculture, communications, and science and technology all benefit from Internet access to information and to individuals through electronic mail. In developing countries, information poverty is one of the more significant and insidious obstacles to effective exploitation of information processing and other types of technology. Lack of adequate information regarding developments in other countries and other environments is often not noticed, and in the absence of new information, old techniques and procedures are continued without conscious knowledge of alternatives.

In addition, even though developing countries may not be hurt in an absolute sense by lack of information, they are certainly negatively affected by any relative measure.

1.1.5 FOSS Community in Africa

Open Source developers and practitioners in Africa are fragmented into minor efforts that fail to compete with profit driven proprietary software industry. Free Software and Open Source Foundation for Africa (FOSSFA) has taken up the task of organizing and giving direction and leadership that bridge such fragmenting in the African FOSS community. FOSSFA's action plan supports a south-to-south interchange of experiences and collaboration on Open Source projects that will help in developing Africa.

Recently, the FOSSFA and a number of partner organizations convened a conference that was content rich and brought together a diverse range of people working in the areas of FOSS and Open Content and saw to the release of a Kiswahili spellchecker. Translate.org.za is making great strides forward under the banner of the Zuza Software Foundation, translating computer software into the eleven official languages of South Africa.Current languages include Xhosa, Zulu, Venda, Northern Sotho, Siswati and Tswana.

In Uganda, a mailing list has been put in place for Linux Users (those working in the IT field in Uganda and are interested in and use Open Source Software). Plans are also underway to conduct regular meetings of Linux Users in Uganda and the monthly I-Network seminar series for a day dedicated to Open Source Software (OSS) where the general public, university students and people from government and industry are invited to participate in information sessions and demonstrations of OSS.

1.1.6 Background to the Case Study

Kampala International University (KIU), located in Kansanga, Kampala commenced operation in October 2001. The mission of KIU is to respond to societal and educational needs by developing and delivering pragmatic academic programmes that are responsive to the market place.

The University offers both undergraduate and graduate programmes on full-time as well as part time basis during the day, evening and weekend. All these programmes include ICT training, and information management in the different staff offices, as well as network management for the university as a whole.

In KIU, the bulk of the computers are using proprietary software especially the student's computer labs. This setting is set to change so as to balance the use of FOSS more for study purposes.

1.2 Problem statement

Open Source and proprietary software use in the ICT industry is growing steadily side-by-side but there is a need to find out the challenges and benefits of each software such that one can objectively make a choice of the appropriate software to use for a particular purpose so as to effectively meet the objectives of the organization.

This research seeks expose the available software by analyzing the accessibility, usability, reliability and security of both OSS/FS and proprietary software such that the findings can act as a guideline to the beneficiaries, when considering OSS/FS or proprietary software usage to meet their business needs.

1.3 Objectives of the study

1.3.1 General objective

The overall objective of this research paper is to access the maturity of FOSS tools in developing content for on-line systems. This shall be based on the analysis of its accessibility, usability, reliability and security.

1.3.2 Specific objectives

i. To investigate the accessibility of FOSS especially to the business sector.
ii. To measure the quality of electronic services when using FOSS as compared to "closed" software to enable the business operators or users to measure the pros and cons of either software.

iii. To analyze the reliability of FOSS tools when used in day to day operations of businesses.
iv. To determine whether FOSS tools offer comfortable levels of security.

1.4 Research questions

The research attempted to answer several queries. The following were some of the questions that guided and directed this research.

i. Has FOSS developed or matured to the level of developing dependable applications?
ii. Does FOSS offer the necessary tools for developing content for on-line commerce?
iii. Why have so few businesses embraced FOSS in developing countries?
iv. How have you evaluated the pros and cons of using FOSS versus "closed" software?

1.5 Scope of the research

Content scope

This research attempted to analyze the accessibility, usability (user friendliness), reliability and security of commonly used software in business. Emphasis is focused on server software and operating systems.

Considering that many business operators may have little or no knowledge of IT's advantage to the fast and success expansion of their businesses, this paper will introduce some must-know fundamentals about e-commerce.

Location scope

The study so much utilized the information literate knowledge workers in Tertiary Institutions (e.g. Kampala International University) and Companies dealing in IT products (Kazinga channel Ltd). Computer Students were also approached for their views. Selection of this scope was determined by the results from the preliminary study about IT knowledge dissemination in our society.

Time scope

Initialization of this study, analysis and compilation of the several findings run from February 2008 to July 2008.

1.6 Significance of the study

Business is in the business of securing its customers. Whether its selling cars, mowing lawns or providing telecommunications around the world, a business will only survive if it provides perfect service to the customers.

Perfect service occurs at the customer's moment of value. That is, Perfect service occurs when the customer wants it (time), where the customer wants it (location), how the customer wants it(form or quality)and in a form that is guaranteed to the customer.

The findings of this research will hopefully act as a guide to the business owners or operators to objectively choose the right choice of software in an effort to integrate IT's advanced strategies in their businesses. The people will gain knowledge of the qualitative standards of their software of interest. So by all means the business sector shall be the primary beneficiaries of the findings in this paper.

In addition, this study is the springboard or the basis of the future researcher to improve the limitations of this study.

CHAPTER II

LITERATURE REVIEW

2.0 Introduction

This chapter basically looks at the various views and findings of other researchers, as well as attempting to introduce some terminologies that are to be continually referred to in the following chapters. This chapter in a way wishes to express the meaning of words in a layman's language that will be easily understood by the targeted community or audience.

2.1 Software

Software is anything that can be stored electronically. It can either be systems software like the operating system and all other utilities or application software that do the real work for users like word processor, spreadsheets etc.

2.1.1 FOSS and proprietary software

Software carries the instructions that tell a computer how to operate. The human authored and human readable form of those instructions is called *source code*. Before the computer can actually execute the instructions, the source code must be translated into a machine readable (binary) format, called the *object code*. All distributed software includes the object code.

Proprietary software owners license their copyrighted object code to a user, which allows the user to run the program. Most software manufacturers retain the ownership rights over their goods and do not release the source codes. This proprietary software is governed by intellectual property law with the owners holding a copyright that gives them exclusive rights to publish, copy, modify and distribute the software and they usually keep the source code hidden. Most proprietary software companies sell an end-user license that lays down how people can use the software on their computers for example, only allowing non-commercial use, or restricting the user's ability to share the software.

Free and Open Source Software is designed to maximize the freedoms of its users. The license conditions for "free" and "open source" software are such that users can exercise four

freedoms; the freedom to use the programme in whatever way they want, the freedom to study and adapt the software code, the freedom to modify and improve the programme, and the freedom to redistribute the original or modified version without having to pay royalties to the original developers.

With conventional software—mostly known as 'proprietary', such as that made by Microsoft or Adobe—it's not possible to see or change the source code, the nuts and bolts that make up the programme. Free and Open Source Software (FOSS) however, is made with exactly the opposite goal in mind. Not only is it possible to see and change the nuts and bolts that make it work, but the community who developed it actually encourage this and rely on this philosophy to see the software spread and grow beyond its original creators.

FOSS has a natural philosophical fit with NGOs and social movements, promoting a 'gift culture', where contributions are given in the faith that others will do the same, in turn creating greater mutual value.

FOSS tools are developed through shared collaboration, often involving teams of developers that are spread globally. This practice of cooperation and collaboration makes the FOSS movement a "natural" political ally of peace, human rights and social change movements. The global collaboration builds a sense of community where the product – FOSS tools – belongs to its community of developers and users, and not to individual programmers or corporations. This enhanced ability to use the technology is part of a shift from being passive consumers in the knowledge economy to helping shape the online environment and collective knowledge base.

FLOSS programs, however, license both the object and the source code, permitting the user to run, modify and possibly redistribute the programs. With access to the source code people have the freedom to run the program for any purpose, redistribute, probe, adapt, learn from, customize the software to suit their needs, and release improvements to the public for the good of the community.

Open source software (OSS) refers to technical advantages of such software (for instance better reliability and security), while Free Software (FS) refers to freedom from control by another and/or ethical issues. The opposite of OSS/FS is "closed" or "proprietary" software.

Other alternative terms for OSS/FS include "libre or "livre" software (where libre or livre means freedom), free-libre / open-source software (FLOSS), free / open source software (FOSS or F/OSS). FLOSS will be used in this research because its easier to pronounce.

All FOSS developers claim copyright, but then use licenses innovatively to give users a variety of freedoms. Dominant examples of these licenses are copylefted and noncopylefted (Lessig, 2003). Under the copylefted license, subsequent FLOSS developers must adopt the same license which ensures that their modifications to the code will remain open. With the non-copylefted license, developers can choose any license to cover their subsequent modifications, even a proprietary license.

2.1.2 Why Open Source Software?

Open source learning environments are becoming widely adopted by educational institutions as the main virtual learning environment in the recent past. A news report released by Becta in Netherlands suggested that schools could save significant sums by switching to open source software. Below are some of the benefits of open source software.

i. The most important is the tendency for open source to be developed using open standards. Open standards are key to interoperability. Of course the growing popularity of open source VLEs encourages proprietary VLEs to also move towards open standards. That benefits everyone.
ii. The ability to tailor the system completely to local needs. Both open and proprietary software are both customizable in a shadow sense- you can tailor the interface within the bounds given by the programmers. With open source, if an institution needs the software customized in ways not thought of by the programmers, they can have the program changed, either in-house or by a third party. Once made, the changes can often be contributed back to the community, making it easier for the next tailor to interface.
iii. Flexibility of use: Proprietary software typically charges in steps. For example, the first 500users might cost x thousand pounds per year and every hundred users above that add a further increment. It does not matter whether the software has been used 8hours a day, 46weeks a year or once in two weeks. There are other costs (servers, bandwidth, support

etc), but theses are approximately proportional to the use, how much data is remote and how much is hand holding.

iv. Lack of surprises: Since the licenses are free and perpetual, a license fee increase cannot happen. There is no vendor to go bankrupt, get taken over or discontinue the system and leave the institutions in the lurch. It is however, true that an institution that has never really used open source in the past will find their first use of it a learning experience.

v. Cooperative development: Open source removes the commercial imperative to compete, enabling genuine cooperation between developers and institutions, among developers and between projects.

vi. Alternative to illegal copying: Institutions that cannot afford to pay for licensing fees may resort to using illegal copies of proprietary software. With FOSS availed, there is no restriction against making as many copies as may be needed.

2.1.3 Draw backs of FOSS

FOSS however is certainly not without limitations. It can often be badly documented and hard to use for the non-technical. In places where pirated versions of proprietary software are easily accessible the benefits of FOSS can be a harder sell. Most people looking for an instant practical solution to their needs aren't going to care too much about whether a programme is open source or not. In these scenarios FOSS has more work to do to prove itself.

2.1.4 Views from developers of open source software

James Dalziel (2004), Architect of LAMS on why they went open source says, "There are several types of benefits for education from LAMS going open source. First, the software itself will freely be available, so although there are still costs involved in set up, maintenance's, there is no license component…."

Selwyn Lloyd of Phosphorix, a company that created the ioNode web services: We have long recognized an opportunity for phosphorix to sustain an ethical business, creating code and products which can be re-used and repurposed by communities which otherwise could not afford to get started, perhaps even creating opportunities for new web communities to begin. This ethical model inspires the ionode team at phosporix and is great for morale. High morale is in

turn good for productivity. Innovation and solutions are not withheld. Most developers want their work to be of value and widely used which improves job satisfaction. Certain OSS products/services are aimed at clients who wish to save time and money; others are for those who advocate the ethical vision. As an OSS project matures it enjoys the benefit of more "committers"".

Brad Wheeler, 2006 in his article Educause Review, states that, "The rising costs of software licensing are an incentive for institutions to consider open source solutions. Open source software projects that are developed of, by, and for higher education are providing favorable economics and are harnessing the industry's vast innovation capability".

2.1.6 The Open Source Initiative (OSI) view

The OSI philosophy is somewhat different from the FSF.

When programmers can read, redistribute and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, and people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.

The OSI is focused on the technical values of making powerful, reliable software, and is more business friendly than the FSF. It is less focused on the moral issues of Free Software and more on the practical advantages of the FOSS distributed development method. While the fundamental philosophy of the two movements are different, both FSF and OSI share the same space and cooperate on practical grounds like software development, efforts against proprietary software, software patents, etc. As Richard Stallman, founder of the FSF, says, the Free Software Movement and the Open Source Movement are two political parties in the same community.

2.1.7 Is FOSS really free?

There are two senses in which free software is free: it has zero direct cost to the user, and it provides the freedom to modify the software.

Stallman, in common with most FOSS advocates, emphasizes the latter usage. Free software, he explains, is "*free as in 'free speech,' not as in 'free beer'*". This distinction is important for two reasons: First, free software is not at all the same as 'freeware', which is zero-price software with closed source code that is often provided as a trial product. Second, it is highly misleading to view the main economic attribute of free software as its price. As is well known, the total cost of installing a software program includes many other costs; even with proprietary software, the price of the software is usually only a modest portion of the total user cost. Large economic benefits arise from the freedom to modify the source code.

2.2 Open standards

2.2.1 Definition of Open Standards

Well-known open source exponent Bruce Perens argues that an open standard is more than just a specification, and that the principles underlying the standard and the practice of offering and operating the standard are what make the standard open. He proposes that open standards should follow the principles of availability, and maximize end-user choice. In addition, there should be no royalty, no discrimination, extension of subset and predatory practices, and certain practices should be followed to ensure that these principles are adhered to. The Perens definition has found wide acceptance among the FOSS communities worldwide.

American National Standards Institute (ANSI), has described open standards8 as those standards that are developed by a process where there is consensus by a group or "consensus body" that is open to representatives from all materially affected and interested parties and there is consideration of and response to comments submitted by voting members of the relevant consensus body as well as by the public. There should also be broad-based public review and comment on draft standards. An avenue for appeal is available for participants who feel that the ANSI open standards principles were not respected during the standards-development process.

In addition, ANSI tries to balance the interests of the implementers and users of the standard with the parties who own IPRs that are essential to implement the standard by allowing the payment of reasonable license fees and/or other reasonable and non-discriminatory license terms that may be required by the IPR holders.

2.2.2 Principles of Open Standards

Bruce Perens has proposed the following principles for open standards. According to his definition, an open standard is more than just a specification. The principles behind the standard, and the practice of offering and operating the standard, are what make the standard open. The principles listed by Bruce Perens are:-

Availability

Open standards are available for all to read and implement. Maximize end-user choice: Open standards create a fair, competitive market for implementations of the standard. They do not lock the customer into a particular vendor or group.

Maximize end-user choice

Open standards create a fair, competitive market for implementation of the standards. They do not lock the customer into a particular vendor or group.

No royalty

Open standards are free for all to implement, with no royalty or fee. Certification of compliance by the standards organization may involve a fee.

No discrimination

Open standards and the organizations that administer them do not favor one implementer over another for any reason other than the technical standards compliance of a vendor's implementation. Certification organizations must provide a path for low- and zero-cost implementations to be validated, but may also provide enhanced certification services.

Extension or subset

Implementations of open standards may be extended, or offered in subset form. However, certification organizations may decline to certify subset implementations, and may place requirements upon extensions.

Predatory practices

Open standards may employ license terms that protect against subversion of the standard by embrace-and-extend tactics. The licenses attached to the standard may require the publication of reference information for extensions, and a license for all others to create, distribute and sell software that is compatible with the extensions. An open standard may not otherwise prohibit extensions.

Practice

Recommended practices for offering and operating each of the principles above have also been discussed by Bruce Perens.

The Open Standards Policy of the State of Massachusetts, USA defines it as specifications for systems that are publicly available and are developed by an open community and affirmed by a standards body. The European Commission's European Interoperability Framework (EIF) [adds on the requirements that open standards should be available either for free or at a nominal charge for usage, copying and distribution and any patents present are to be made irrevocably available on a royalty-free basis, and there should be no constraints on the re-use of the standard. Thus, the Perens definition is consistent generally with those of the EIF and the State of Massachusetts, and this approach has gained currency with many policy makers as the basis of their open standards policies.

Open standards in Information and Communications Technologies (ICTs) allow freedom to access as well as to contribute to the development of the standard by any interested party, which is not possible under a proprietary standard. Standards are *de facto* and *de jure,* and both arise out of complex dynamics influenced by economic, political and social forces.

Open standards in ICTs are critical to allow new entrants to participate, innovate on standards implementation, and compete. With a proprietary standard, the owner can prevent competitors and entrants from capturing market share through their legally enforceable IPR. In a developing economy like Uganda, proprietary ICT standards are typically held by foreign enterprises which effectively relegate domestic engagement to the level of franchisee.

It cannot be over-emphasized that the Internet and World Wide Web, which are having so much impact on the world today, would not exist without FOSS and the inter-related development and adoption of the essential open standards. Today, nearly two-thirds of all web servers use FOSS. A FOSS program must be released under some license giving its users a certain set of rights; the most popular FOSS license is the GNU General Public License (GPL). All software released under the GPL is FOSS, but not all FOSS software uses the GPL (Wheeler, 2007).

In Uganda, Inveneo, a San Francisco non-profit organization, together with Action Aid, an international agency whose aim is to fight poverty worldwide, installed their first rugged Linux desktop systems in western Uganda. The systems run Linux and KDE desktops, and also include the OpenOffice.org productivity suite (Kendrick, 2005).

"The system is up and running since this June, 2005 where 5 units have been installed, 4 of which are in villages with no access to power. The system provides Internet access and phone capabilities to the users. Phone calls among the connected villages are free of charge, with the ability to place and receive calls to the Ugandan phone network. The systems are linked using 802.11 WiFi links." (Summer, 2005).

2.3 Terminology

2.3.1 E-commerce and E-business

Electronic commerce, commonly known as e-commerce or eCommerce, consists of the buying and selling of products or services over electronic systems such as the Internet and other computer networks. The amount of trade conducted electronically has grown dramatically since the spread of the Internet. A wide variety of commerce is conducted in this way, spurring and drawing on innovations in electronic funds transfer, supply chain management, Internet

marketing, online transaction processing, electronic data interchange (EDI), inventory management systems, and automated data collection systems. Modern electronic commerce typically uses the World Wide Web at least at some point in the transaction's lifecycle, although it can encompass a wider range of technologies such as e-mail as well. E-commerce is buying things from the internet but many people are unsure about its reliability as there are many unreputable vendors.

A small percentage of electronic commerce is conducted entirely electronically for "virtual" items such as access to premium content on a website, but most electronic commerce involves the transportation of physical items in some way. Online retailers are sometimes known as e-tailers and online retail is known as e-tail. E-commerce or electronic commerce is generally considered to be the sales aspect of e-business.

In many cases, an e-commerce company survives not only based on its product, but through a competent management team, post-sales services, well-organized business structure, network infrastructure and a secured, well-designed website. The factors can be divided between technical or organization aspects and direct service to the consumer.

Technical and organizational aspects

i. Sufficient work done in market research and analysis. Like traditional models, e-commerce implicates good business planning and the fundamental laws of supply and demand.

ii. A good management team armed with information technology strategy. A company's IT strategy can involve the business re-design process.

iii. Providing an easy and secured way for customers to effect transactions. Credit cards are the most popular means of sending payments on the internet, accounting for 90% of online purchases. In the past, card numbers were transferred securely between the customer and merchant through independent payment gateways. Such independent payment gateways are still used by most small and home businesses. Most merchants process credit card transactions on site through arrangements made with commercial banks or credit cards companies.

iv. Providing reliability and security. Parallel servers, hardware redundancy, fail-safe technology, information encryption, and firewalls can enhance this requirement.

v. Providing a 360-degree view of the customer relationship, defined as ensuring that all employees, suppliers, and partners have a complete view, and the same view, of the customer. However, customers can react against a big brother experience.

vi. Constructing a commercially sound business model.

vii. Engineering an electronic value chain focused on a "limited" number of core competencies. Electronic stores have succeeded as either specialist or generalist in aim.

viii. Operating on or near the cutting edge of technology and staying there as technology changes.

ix. Setting up an organization of sufficient alertness and agility to respond quickly to any changes in the economic, social and physical environment.

x. Providing an attractive website. The tasteful use of color, graphics, animation, photographs, fonts, and white-space percentage may aid success in this respect.

xi. Streamlining business processes, possibly through re-engineering and information technologies.

xii. Providing complete understanding of the products or services offered, which not only includes complete product information, but also sound advisers and selectors.

Other standard necessities include honesty about its product and its availability, shipping reliably, and handling complaints promptly and effectively. A unique property of the Internet environment is that individual customers have access to far more information about the seller than they would find in a brick-and-mortar situation. (Of course, customers can, and occasionally do, research a brick-and-mortar store online before visiting it, so this distinction does not hold water in every case.)

Customer experience

A successful e-commerce organization must also provide an enjoyable and rewarding experience to its customers. Many factors go into making this possible. Such factors include:

1. Providing value to customers. Vendors can achieve this by offering a product or product-line that attracts potential customers at a competitive price, as in non-electronic commerce.

2. Providing service and performance. Offering a responsive, user-friendly purchasing experience, just like a flesh-and-blood retailer, may go some way to achieving these goals.

3. Providing an incentive for customers to buy and to return. Sales promotions to this end can involve coupons, special offers, and discounts. Cross-linked websites and advertising affiliate programs can also help.

4. Providing personal attention. Personalized web sites, purchase suggestions, and personalized special offers may go some of the way to substituting for the face-to-face human interaction found at a traditional point of sale.

5. Providing a sense of community. Chat rooms, discussion boards, soliciting customer input and loyalty programs (sometimes called affinity programs) can help in this respect.

6. Owning the customer's total experience. E-tailers foster this by treating any contacts with a customer as part of a total experience, an experience that becomes synonymous with the brand.

7. Letting customers help themselves. Provision of a self-serve site, easy to use without assistance, can help in this respect. This implies that all product information is available, cross-sell information, advice for product alternatives, and supplies & accessmry selectors.□Helping customers do their job of consuming. E-tailers and online shopping directories can provide such help through ample comparative information and good search facilities. Provision of component information and safety-and-health comments may assist e-tailers to define the customers' job.

8. Helping customers do their job of consuming. E-tailers and online shopping directories can provide such help through ample comparative information and good search facilities. Provision of component information and safety-and-health comments may assist e-tailers to define the customers' job.

E-Business is much more than e-Commerce – buying and selling on-line. Companies are increasingly using Information and Communication Technologies (ICTs) to link together their business processes and systems:

- **internally:** hooking departments together to provide better products and more responsive services more efficiently;
- with those of their **suppliers, distributors and other partners**, increasing efficiencies even further;
- With **public authorities**, reducing red tape in both public and private sectors.
- With their **customers**, allowing them to respond more directly to market trends and sell worldwide.

E-Business therefore allows new forms of partnership, and improves both the way companies work and the products and services they offer.

2.3.2 Interoperability

Interoperability is a property referring to the ability of diverse systems and organizations to work together (inter-operate). The term is often used in a technical systems engineering sense, or alternatively in a broad sense, taking into account social, political, and organizational factors that impact system to system performance.

The IEEE defines interoperability as; the ability of two or more systems or components to exchange information and to use the information that has been exchanged.

In telecommunication, the term can be defined as:

1. The ability of systems, units, or forces to provide services to and accept services from other systems, units or forces and to use the services exchanged to enable them to operate effectively together.
2. The condition achieved among communications-electronics systems or items of communications-electronics equipment when information or services can be exchanged directly and satisfactorily between them and/or their users. The degree of interoperability should be defined when referring to specific cases.

In two-way radio, interoperability is composed of three dimensions:

- compatible communications paths (compatible frequencies, equipment and signaling),
- radio system coverage or adequate signal strength, and;
- scalable capacity.

Software Interoperability

With respect to software, the term interoperability is used to describe the capability of different programs to exchange data via a common set of exchange formats, to read and write the same file formats, and to use the same protocols. (The ability to execute the same binary code on different processor platforms is 'not' contemplated by the definition of interoperability.) The lack of interoperability can be a consequence of a lack of attention to standardization during the design of a program. Indeed, interoperability is not taken for granted in the non-standards-based portion of the computing world.

According to ISO/IEC 2382-01, Information Technology Vocabulary, Fundamental Terms, interoperability is defined as follows: "The capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units".

The definition is somewhat ambiguous because the user of a program can be another program and, if the latter is a portion of the set of program that is required to be interoperable, it might well be that it does need to have knowledge of the characteristics of other units. This

definition focuses on the technical side of interoperability, however, interoperability is often more of an organizational issue: often interoperability has a significant impact on the organizations concerned, raising issues of ownership (do people want to share their data?), labor relations (are people prepared to undergo training?) and usability. In this context, a more apt definition is captured in the term "business process interoperability".

Interoperability can have important economic consequences, such as network externalities. If competitors' products are not interoperable (due to causes such as patents, trade secrets or coordination failures), the result may well be monopoly or market failure. For this reason, it may be prudent for user communities or governments to take steps to encourage interoperability in various situations. In the United Kingdom, for example, there is an e-Government-based interoperability initiative called e-GIF. As far as user communities, Neutral Third Party is creating standards for business process interoperability. Another example of a neutral party is the RFC documents from the Internet Engineering Task Force (IETF).

Achieving Interoperability

Interoperability can be achieved in four ways: through product engineering, industry/community partnership, access to technology and IP, and implementation of standards.

Interoperability as a question of power and market dominance

Interoperability tends to be regarded as an issue for experts and its implications for daily living are sometimes underrated. The case of Microsoft vs. the European Commission shows how interoperability concerns important questions of power relationships. In 2004, the European Commission found that Microsoft had abused its market power by deliberately restricting interoperability between Windows work group servers and non-Microsoft work group servers. By doing so, Microsoft was able to protect its dominant market position for work group server operating systems, the heart of corporate IT networks. Microsoft was ordered to disclose complete and accurate interface documentation, which will enable rival vendors to compete on an equal footing ("the interoperability remedy"). As of June 2005 the Commission is market testing a new proposal by Microsoft to do this, having rejected previous proposals as insufficient.

Recent Microsoft efforts around interoperability may indicate a shift in their approach and level of commitment to interoperability. These efforts including the migration of Microsoft Office file formats to ECMA Office Open XML, and several partner interoperability agreements, most notably their recent collaboration agreement with Novell.

Interoperability has also surfaced in the Software patent debate in the European Parliament (June/July 2005). Critics claim that because patents on techniques required for interoperability are kept under RAND (reasonable and non discriminatory licensing) conditions, customers will have to pay license fees twice: once for the product and, in the appropriate case, once for the patent protected programme the product uses.

Bill Gates Speaks on Software interoperability

On February 3rd, 2005, Bill Gates sent out an executive email to the public regarding the importance of software interoperability. He stated:

"Interoperability is more pragmatic than other approaches, such as attempting to make all systems compatible at the code level, focusing solely on adding new layers of middleware that try to make all systems look and act the same, or seeking to make different systems interchangeable. With a common understanding of basic protocols, different software can interact smoothly with little or no specific knowledge of each other. The Internet is perhaps the most obvious example of this kind of interoperability, where any piece of software can connect and exchange data as long as it adheres to the key protocols".

He then goes on to explain how Microsoft approaches interoperability by working to ensure that its software and operating systems work well with the other software and systems already in existence. He further explains that the next step is to define the next generation of software with a greater degree of "interoperability by design", specifically using XML as a common framework from which different applications can communicate.

Using XML as a common interoperability mechanism is a great idea. It allows one to publicize your communication / interoperability schemas without requiring interfacing

programs/programmers to understand all of the details about the inner workings of your software. Furthermore, the self-describing nature of XML eases interoperability by reducing the need for extensive documentation surrounding your interoperability mechanisms. Finally, since it's a common standard, it can be used and accepted broadly.

Below is an expression of software interoperability playing the two role network game, when one of the player clients (top left) runs under Sun Microsystems and another under GNU Classpath with JamVM. The applications execute the same bytecode and interoperate using the standard RMI-IIOP messages for communication

2.3.3 Reliability

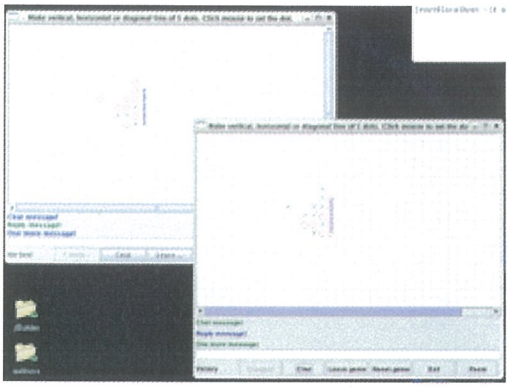


Figure 2.1

Reliability is popularly measured by how long software can resist crashing or freeze-ups from the time it is installed. For instance paper "Fuzz Revisited" measured reliability by feeding programs random characters and determining which ones resisted crashing and freeze-ups.

FOSS had higher reliability by this measure. A later paper published in 2000, "An Empirical Study of the Robustness of Windows NT Applications Using Random Testing", found that with Windows NT GUI applications, they could crash 21% of the applications they tested, hang an additional 24% of the applications, and could crash or hang *all* the tested applications when subjecting them to random Win32 messages.

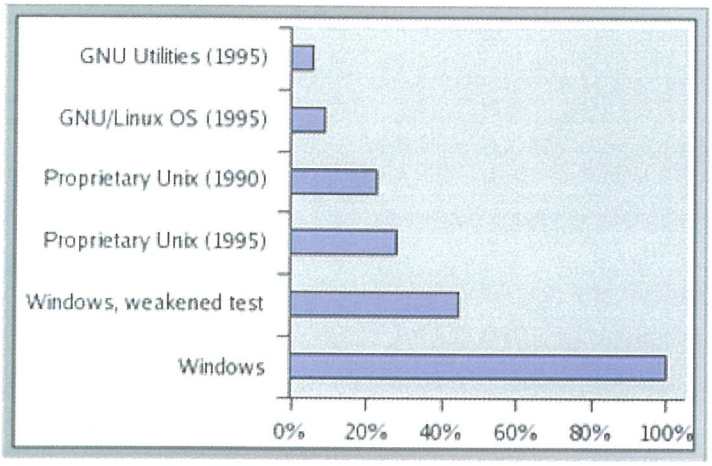


Figure 2.2. Chart comparing failure rates of OSS/FS and proprietary software (Wheeler, 2007).

The fuzz paper's authors also found that proprietary software vendors generally didn't fix the problems identified in an earlier version of their paper (from 1990). There was a slight decrease in failure rates between their 1990 and 1995 paper, but many of the flaws they found (and reported) in the proprietary Unix programs were still not fixed 5 years later. In contrast, Scott Maxwell led an effort to remove every flaw identified in the OSS/FS software in the 1995 fuzz paper, and eventually fixed every flaw. Thus, the OSS/FS community's response shows why, at least in part, OSS/FS programs have such an edge in reliability; if problems are found, they're often fixed.

IBM put Linux to a 10-month test and ran Caldera Systems OpenLinux, Red Hat Linux, and Windows NT Server 4.0 with Service Pack 3 on duplicate 100MHz Pentium systems with 64MB of memory. The results: the NT server crashed an average of once every six weeks. Each failure took roughly 30 minutes to fix. That is really bad considering that neither Linux server ever went down. (Vaughan-Nichols, et al, 1999).

Table 2.1. Uptime for OSes (www.apple.com)

Plotted Value	No. samples	Max	Latest
MacOSX	156	134.54	31.06
BSD/OS	3	16.18	16.18
Solaris	413	180.18	25.23
90-day Moving average	669	97.158	13.62

The table above shows the Maximum and Latest uptimes in days for each of the Operating Systems detected. The number of samples is the number of samples for each type recorded, usually one per day. (Netcraft, 2007)

A 3-month Swiss evaluation, on the difference between Netscape (proprietary) and Apache (FOSS) shows the results of Syscontrol AG's analysis of website uptime (announced February 7, 2000) They measured over 100 popular Swiss web sites over a three-month period, checking from 4 different locations every 5 minutes. Below is their set of published data on "average down-time (in hours in that month) for each type of server",.

Table 2.2. AG's Analysis of website uptime (Wheeler, 2006).

Downtime	Apache	Microsoft	Netscape	Other
September	5.21	10.41	3.85	8.72
October	2.66	8.39	2.80	12.05
November	1.83	14.28	3.39	6.85
Average	3.23	11.03	3.35	9.21

German import company Heinz Tröber compared Linux-based desktops to Windows desktops and found that Windows had a 15% daily failure rate, while Linux has 0%. Günter Stoverock, the data processing manager at German import company Heinz Tröber, reported that they had decided to run its ERP software on Linux-based systems, instead of Windows, because Windows was much less reliable. Stoverock stated that on Windows, "Out of 65 desktops, around 10 desktops crashed daily. Employees wasted around 30 minutes, that's five times 30 minutes per week." Note that this is a 15% daily failure rate, and the actual impacts were almost certainly more severe than simply a loss of 2 minutes of lost time per reboot.

Damien Challet and Yann Le Du of the University of Oxford have written a paper titled Closed source versus open source in a model of software bug dynamics, where they develop a model of software bug dynamics in which users, programmers and maintainers interact through a given program. Then they analyzed the model, and found that all other things being equal (such as number of users, programmers, and quality of programmers), "debugging in open source projects is always faster than in closed source projects." (Damien, et al, 2005).

However, one problem with reliability measures is that it takes a long time to gather data on reliability in real-life circumstances. Thus, there's more data comparing older Windows

editions to older GNU/Linux editions. The key is that these comparisons are fair, because they compare contemporaneous products. (Wheeler, 2007).

2.3.4 Usability

Free and Open Source Software (F/OSS) have special characteristics due to the nature of the development process. First, the entity behind the product is not an organization, rather a community of volunteers and freelance professionals. Typical user support is provided through online communities and paid consultants. Therefore immediate customer service is not a viable option.

Second, number of available choices for a particular tool is considerably high. For example there are over 300 Linux distributions available. The number of available choices for a particular software tool sometimes exceeds 10. This results in quick turnover rates for open source software. Furthermore, cost characteristics of F/OSS software allow users for easy turnover.

Third, a tough interface standardization entity is not present in development community. The best approach for exposing new users is Out Of Box Experience. However, lack of standardization and consistency are a cause of usability problems during OOBE period. This may adversely affect "quality and functionality perception" of the software being used.

According to Pirhonen (2002), learnability is the central factor on OOBE and suggests good use of metaphors to provide better learnability therefore better OOBE .
However, it is possible to consider "computer software" as a piece of consumer product and expect an OOBE associated with it. A user will go through a learning phase for both computer hardware and software. Pirhonen suggests that a model like one shown below could be of help.

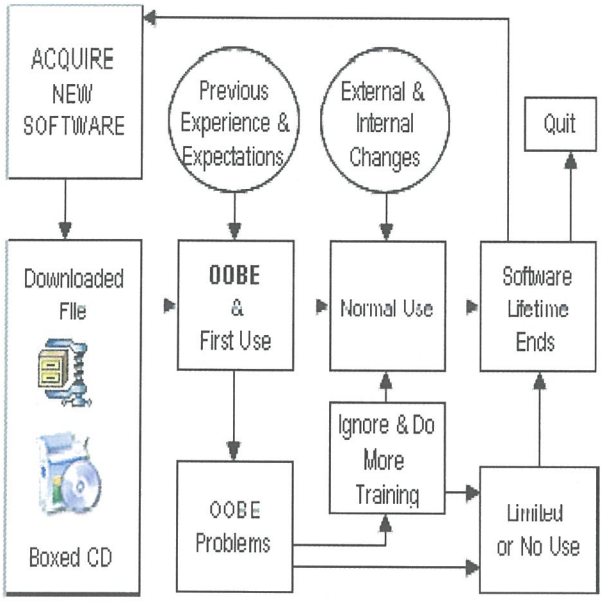


Fig. 1. Software OOBE lifecycle model

2.3.5 Security

Quantitatively measuring security is very difficult. Security is it is often measured by a survey of opinions, from people who are informed and have use used the software. because it is hard to measure. Though opinions can be wrong it is very hard to ignore the opinions collected from a large sample space of users who are in the know. This research will concentrate on comparing FOSS to Windows systems, since other proprietary systems are increasingly including FOSS components (making comparisons more difficult).

At one time the security of FOSS systems was widely debated. Clearly FOSS systems are not magically invincible from security flaws. But for most of those who study the question, the issue of whether or not FOSS improves or reduces security appears to be an increasingly settled issue. Below are some quantitative studies that compare this software.

According to a combination of studies from the Honeynet Project (2004), AOL, and others that compared the unpatched Linux systems to unpatched Windows systems, the average Linux system lasts three months before being compromised, (a significant increase from the 72 hours life span of a Linux system in 2001). Unpatched Windows systems continue to be

compromised far more quickly, sometimes within minutes. This data on Windows compromise is consistent with other studies. Avantgarde found that Windows did not last long, and one unpatched Windows XP system (pre-SP2) only lasted 4 minutes on the Internet before it was compromised and in general did not last long. Note, however, that users who install Windows Service Pack 2 have much less risk than previous versions of Windows. Symantec's Internet Security Threat Report (January 1-June 30, 2004), The Internet Storm Center's Survival Time History claims that by December 2004 a Windows survival time of 18 minutes.

The Bugtraq vulnerability database that examines security was used to compare the vulnerability of OSes running FOSS and proprietary software in 1999-2000. One approach to examining security is to use a vulnerability database. Below is an analysis of one database from the Bugtraq Vulnerability Database Statistics page as of September 17, 2000, listing the total number of vulnerabilities for some leading OSes:

Table 2.3. Comparing vulnerability of some leading OSes (Wheeler, 2007)

OS	1997	1998	1999	2000
Debian GNU/Linux	2	2	30	20
OpenBSD	1	2	4	7
Red Hat Linux	5	10	41	40
Solaris	24	31	34	9
Windows NT/2000	4	7	99	85

Some vulnerabilities are more important than others (some may provide little if exploited or only be vulnerable in unlikely circumstances), and some vulnerabilities are being actively exploited (while others have already been fixed before exploitation).

Comparing the FOSS and proprietary response to security problems; Red Hat (an FOSS vendor) responded more rapidly than Microsoft or Sun to advisories. Sun had fewer advisories to respond to yet took the longest to respond. SecurityPortal compiled a list of "the time it takes for vendors to respond to vulnerabilities", and concluded that Red Hat had the best score, with 348 recess days on 31 advisories, for an average of 11.23 days from bug to patch. Microsoft had 982 recess days on 61 advisories, averaging 16.10 days from bug to patch. Sun proved itself to be very slow, although having only 8 advisories it accumulated 716 recess days (three months) to fix each bug on average. Their table of data for 1999 is as shown:

Table 2.4. 1999 Advisory Analysis (Wheeler, 2007)

Vendor	Total Days, Hacker Recess	Total Advisories	Recess Days/Advisory
Red Hat	348	31	11.23
Microsoft	982	61	16.10
Sun	716	8	89.50

It is worth noting that the OpenBSD OS (FOSS) had fewer reported vulnerabilities than all of these. Clearly, having a proprietary OS doesn't mean you're more secure - Microsoft had the largest number of security advisories, by far, using either counting method.

Eweek Labs' article "Open Source Quicker at Fixing Flaws" (September 30, 2002) listed specific examples of more rapid responses. This article can be paraphrased as follows: In June 2002, a serious flaw was found in the Apache Web server; the Apache Software Foundation

made a patch available two days after the Web server hole was announced. In September 2002, a flaw was announced in OpenSSL and a patch was available the same day. In contrast, a serious flaw was found in Windows XP that made it possible to delete files on a system using a URL; Microsoft fixed this problem in Windows XP Service Pack 1 without notifying users of the problem. A more direct comparison can be seen in how Microsoft and the KDE Project responded to an SSL (Secure Sockets Layer) vulnerability that made the Internet Explorer and Konqueror browsers, respectively, potential tools for stealing data such as credit card information. The day the SSL vulnerability was announced, KDE provided a patch.

In an August 18, 2004 interview, Symantec's chief technology officer Robert Clyde argued that proprietary vendors were more reliable for fixing problems within a fixed timescale, and that he didn't know of a single vendor who would sit on a vulnerability. Yet the day before (August 17), an eWeek article revealed that Oracle waited 8 months to fix a vulnerability. And Microsoft waited 9 months to fix a critical IE vulnerability (and only fixed it after it was being actively exploited in 2004).

A study on software immunity from outside attacks by Evans Data Corp.'s (2002), over 400 GNU/Linux developers found that even though computer attacks have almost doubled annually since 1988 (according to CERT), 78% of the respondents to the GNU/Linux developers survey have never experienced an unwanted intrusion and 94% have operated virus-free. Clearly, the survey shows that GNU/Linux "doesn't get broken into very often and is even less frequently targeted by viruses," according to Jeff Child Evans Data Corp.'s Linux Analyst who tested Linux systems immunity from attacks from outsiders notes that it's much harder to hack a knowledgeable owner's system (and most Linux developers have hands-on, technical knowledge) and that because there are fewer desktop GNU/Linux systems there are fewer viruses being created to attack GNU/Linux. The developers being surveyed attributed the low incidence of attacks to the Open Source Software (OSS) environment; "more than 84% of Linux developers believe that Linux is inherently more secure than software not created in an OSS environment," and they ranked "Linux's security roughly comparable in security to Solaris and AIX ... and above any of the Windows platforms by a significant margin."

Eweek's report that compared the security of Apache to Microsoft's IIS, examined that Apache's last serious security problem (one where remote attackers could run arbitrary code on the server) was announced in January 1997. A group of less serious problems (including a buffer overflow in the server's logresolve utility) was announced and fixed in January 1998 with Apache 1.2.5. In the three and a half years since then, Apache's only remote security problems have been of denial-of-service and information leakage problems (where attackers can see files or directory listings they shouldn't). eWeek's April 10, 2002 article noted that ten more IIS flaws were found in IIS Server 4.0, 5.0, and 5.1, some of which would allow attackers to crash the IIS service or allow the attacker to run whatever code he chooses.

Apache wisely follows the good security practice of "least privilege." While IIS is designed so that anyone who takes over IIS can take over the whole system, performing actions such as reading, modifying, or erasing any file on the system. In contrast, Apache is installed with very few privileges by default, so even taking over Apache gives attackers relatively few privileges. For example, cracking Apache does not give attackers the right to modify or erase most files.

The article claims there are four reasons for Apache's strong security, and three of these reasons are simply good security practices. Apache installs very few server extensions by default (a "minimalist" approach), all server components run as a non-privileged user (supporting "least privilege" as noted above), and all configuration settings are centralized (making it easy for administrators to know what's going on). However, the article also claims that one of the main reasons Apache is more secure than IIS is that its "source code for core server files is well-scrutinized," a task that is made much easier by being FOSS, and it could be argued that OSS/FS encourages the other good security practices.

IIS was attacked 1,400 times more frequently than Apache in 2001, and Windows was attacked more than all versions of Unix. SecurityFocus co-founder and CEO Arthur Wong reported an analysis of the various vulnerabilities and attacks (based on SecurityFocus's data) in the February 2002 article. IIS was attacked 17 million times, but Apache was attacked only 12,000 times. In 2001, Windows systems were attacked 31 million times, while Unix systems were attacked 22 million times.

Some security vulnerabilities are more important than others, for a variety of reasons. Thus, some analysis centers try to determine what's "most important," and their results suggest that OSS/FS just doesn't have as many vulnerabilities.

The CERT Coordination Center (CERT/CC) is federally funded to study security vulnerabilities and perform related activities such as publishing security alerts. Four of the six most important security vulnerabilities were specific to Microsoft: W32/Nimda, W32/Sircam, cache corruption on Microsoft DNS servers, and "Code Red" related activities. Only one of the six items primarily affected non-Microsoft products (a buffer overflow in telnetd); while this vulnerability is important, it's worth noting that many open source systems (such as Red Hat 7.1) normally do not enable this service (telnet) in the first place and thus are less likely to be vulnerable.

Computer viruses are a serious security problem in software. Virus infection has been a major cost to users of Microsoft Windows. The LoveLetter virus alone is estimated to have cost $960 million in direct costs and $7.7 billion in lost productivity, and the anti-virus software industry sales total nearly $1 billion annually. Dr Nic Peeling and Dr Julian Satchell's *Analysis of the Impact of Open Source Software* includes an analysis of the various data sources for virus counts, noting the disproportionate vulnerability of Windows systems.

They found out that numbers differ in detail, but all sources agree that computer viruses are overwhelmingly more prevalent on Windows than any other system. There are about 60,000 viruses known for Windows, 40 or so for the Macintosh, about 5 for commercial Unix versions, and perhaps 40 for Linux. Most of the Windows viruses are not important, but many hundreds have caused widespread damage. Two or three of the Macintosh viruses were widespread enough to be of importance. None of the Unix or Linux viruses became widespread - most were confined to the laboratory.

In contrast, while it's possible to write a virus for FOSS OSes, their design makes it more difficult for viruses to spread... showing that Microsoft's design decisions were not inevitable. It appears that FOSS developers tend to select design choices that limit the damage of viruses, probably in part because their code is subject to public inspection and comment (and redicule, if

deserving of it). For example, FOSS programs generally do not support attacker-controlled start-up macros, nor do they usually support easy execution of mail attachments from attackers.

Also, leading OSS/FS OSes (such as GNU/Linux and the *BSDs) have always had write protection on system directories, making it more difficult for certain attacks to spread. OSS/FS systems are *not* immune to malicious code, but they are certainly more resistant.

According to a June 2004 study by network management firm Sandvine, 80% of all spam comes from computers contaminated with Trojan horse infections. Trojans and worms with backdoor components turn infected PCs into drones in vast networks of compromised zombie PCs. Most PCs affected were running Windows programs.

Sandvine identified subscribers bypassing their home mail servers and contacting many mail servers within a short period of time over sustained periods - i.e., spammers. It also looked at SMTP error messages returned to clarify the total volume of spam. They then compared this with the messages passing through the service provider's mail system.

National Cyber Security Alliance's study of May 2003 reported that 91% of Broadband users have spyware on their home computers running proprietary operating systems. In contrast, there's no evidence of that this is an issue for OSS/FS systems. America Online, Inc. conducted a study for the National Cyber Security Alliance. Its results, "Fast and Present Danger: In-Home Study on Broadband Security among American Consumers" (May 2003). They found that "91% of Broadband Users Have Spyware Lurking on Home Computers". Their study method did not appear to permit collection of data from OSS/FS systems, and spyware systems are essentially nonexistent on OSS/FS systems anyway.

The SecurityTracker Statistics paper (March, 2002) analyzed vulnerabilities from April 2001 through March 2002 and identified 1595 vulnerability reports, covering 1175 products from 700 vendors. Their analysis found that Microsoft had 187, or 11.7% of all vulnerabilities), and more than four times the next vendor. The next largest were Sun (42, 2.6% of the total), HP (40, 2.5%), and IBM (40, 2.5%). Solely OSS/FS vendors did much better: the Apache Software Foundation had 13 (0.8% of the total), and Red Hat had 10 (0.6% of the total).

In late June 2004, Microsoft had failed to patch a critical vulnerability for 9 months, and IE was being actively exploited in horrendous ways. Customers then rushed to download Mozilla and Mozilla Firebird, popular OSS/FS alternatives, to replace IE. The U.S. CERT warned that the Microsoft browser (IE) cannot protect against vulnerabilities, and there were dangerous active attacks exploiting them. A team of crackers (supposedly Russia-based) exploited Microsoft IE vulnerabilities by also exploiting other vulnerabilities in Microsoft's IIS. The crackers broke into IIS sites and inserted malicious code that IE users would download if they viewed an IIS site they'd broken into. The IE users who visited those sites (who legitimately trusted these sites) would have their IE program exploited, which then compromised their system.

FOSS programs sometimes have vulnerabilities too, but it's rare that they last so long. More importantly, users of FOSS programs can always fund to have a repair created and implemented quickly if it is important to them, and can have that fix reviewed and shared with others worldwide. Proprietary users have no such options; proprietary users are completely dependent on the proprietary vendor for making any emergency repairs, and for more reacting more responsibly than this. Mozilla argues that IE is in general less secure, in part because Microsoft's ActiveX technologies, IE's tight integration into the Microsoft operating system, and IE's weak default security settings make IE easier to exploit than its competition. For example, Semantic recommends that users consider disabling ActiveX altogether, because of ActiveX's problems.

In contrast, every change made to Mozilla applications is first peer reviewed by at least two engineers who are familiar with the code and overall architecture of the system before the new code is allowed into the product. The product then goes through automated tests and evaluations, and then Mozilla users and the development community are invited to review the impact of each change by downloading the test builds that are produced two or three times a day. All source code is available for review by anyone.

Symantec Internet Security Threat Report, Volume VII (released March 2005), found that Internet Explorer had 9 highly severe vulnerabilities affecting it in the time period, while Firefox

had 7. In addition, the Internet Explorer flaws also took longer to fix -- an average of 43 days, compared to 26 days for Mozilla browsers (which presumably includes Firefox).

In all previous reports, the total number of Mozilla vulnerabilies was lower than IE. The bad news is that this March 2005 report reports that in this period there were more total vulnerabilities (though fewer high severity ones) in Mozilla-based browsers than in IE. There are 13 vulnerabilities affecting Internet Explorer, compared to 21 vulnerabilities affecting the Mozilla and Mozilla Firefox browsers during the survey period. Symantec was encouraged that the security vulnerabilities found in Firefox, were at least less likely to be of high severity.

CNet's September 2005 reported that Mozilla browsers were more vulnerable than IE. A latest study found that 25 vendor-confirmed vulnerabilities were disclosed for the Mozilla browsers during the first half of 2005 (18 were high severity); during the same period, 13 vendor-confirmed vulnerabilities were disclosed for IE (eight were high severity).

CNet noted that Microsoft generally releases patches only on a monthly basis, which is more than 6 days. Even more importantly, since IE has many more unaddressed vulnerabilities compared to in Mozilla, IE's average response times would increase more rapidly too (making "equality" only make sense when you ignore the unpatched vulnerabilities).

Also, David Hammond's "Internet Explorer is dangerous" of (2005, August) examined the Secunia reports on Internet Explorer, Firefox, and Opera, as shown below.

Table 2.5. Comparing Internet Explorer, Mozilla Firefox and Opera vulnerabilities.

Feature	Internet Explorer	Firefox	Opera
Historical quantity	43	21	23
Present quantity	25	4	0
Historical relative danger	121	56	59
Present relative danger	50	9	0

The "quantity" shows the number of vulnerabilities, but doesn't account for their criticality. Thus, he also computes a "relative danger" by simply "adding up the criticality levels for each vulnerability (not critical=1, extremely critical=5)". As of that date:

- "Internet Explorer has had 43 reported vulnerabilities. 7 were marked as moderately critical, 11 were marked as highly critical, and 6 were marked as extremely critical. There are still 25 unfixed vulnerabilities, including 6 that were marked as moderately critical, 1 that was marked as highly critical, and 1 that was marked as extremely critical."

- "Mozilla Firefox has had 21 reported vulnerabilities. 8 were marked as moderately critical, 4 were marked as highly critical, and 0 were marked as extremely critical. There are still 4 unfixed vulnerabilities, including 1 that was marked as moderately critical."

- "Opera has had 23 reported vulnerabilities. 14 were marked as moderately critical, 0 were marked as highly critical, and 0 were marked as extremely critical. All reported vulnerabilities have since been fixed."

Brian Krebs "Security Fix" column compiled statistics on vulnerability response times, including those for Microsoft Internet Explorer (IE) and Mozilla Firefox. He found that for "a total 284 days in 2006 (or more than nine months out of the year), exploit code for known, unpatched critical flaws in pre-IE7 versions of the browser was publicly available on the Internet. Likewise, there were at least 98 days last year in which no software fixes from Microsoft were available to fix IE flaws that criminals were actively using to steal personal and financial data from users.

In TechWeb.com (February 9, 2005), Gregg Keizer's article "Spyware Barely Touches Firefox" describes some research work from the University of Washington. Henry Levy stated that his research showed that users "will have a safer experience [surfing] with Firefox." Researchers Henry Levy and Steven Gribble crawled 45,000 websites, cataloguing their executable files, and then exposed unpatched Internet Explorer (IE) and Firefox browsers to

them. "If you browse, you're eventually going to get hit with a spyware attack." (Levy) Perhaps choosing the program with the better record would help.

Microsoft took 134 days on average to release patches for security problems in 2004-2005; Mozilla averaged 37 days. Brian Krebs' "A Time to Patch II: Mozilla" compared patch times of Mozilla with Microsoft. Even with an outlier included, Mozilla did much better on average than Microsoft. Mozilla took an average of about 37 days to issue patches for critical security problems in its products over a 3-year period. In general it did much better; one-third of its critical security updates were within less than 10 days of being notified. (The longest time was for a bug that perhaps should not have been marked as "critical"; Microsoft had exactly the same bug but marked it only as "moderate.)

In a similar study of Microsoft's vulnerability report response times, he notes that "In 2003, Microsoft took an average of three months to issue patches for problems reported to them. In 2004, that time frame shot up to 134.5 days, a number that remained virtually unchanged in 2005."

Christian Payne's Information Systems Journal, Vol.12, Issue 1, February 2002, includes the peer-reviewed paper "On the security of open source software" by Christian Payne of Murdoch University (Perth, Australia). In it, Payne first summarizes the various arguments made for and against open source software. He discusses some of the arguments that FOSS is more secure, in particular, claims that the process of peer review improves security, FOSS flexibility and freedom is a significant aid (e.g., organizations are free to audit FOSS, modify it to meet their security needs, and rapidly patch OSS/FS without having to wait for a vendor), and that OSS/FS projects tend to respond more quickly with security fixes. He also discusses some of the arguments made against OSS/FS, such as claims that that vulnerabilities are harder for attackers to find in proprietary programs (since the source code is not available), and that there are flaws in the peer review argument (e.g., it may be available but not necessarily reviewed). In short, there are different effects, and it's easy to have opinions about the strengths of those different effects. Without measurement, it's hard to know what effects are stronger.

But Payne goes beyond a mere summary of arguments, and actually works to try to gather quantitative data to measure the effect of these alternative approaches. Payne devised a scoring system for measuring security features, measuring reported security vulnerabilities, and then rolling those two factors into a final score. He then applied this to two OSS/FS systems (Debian and OpenBSD) and one proprietary system (Solaris, which at the time was proprietary); all are Unix-based operating systems.

Table 2.6. Security of OSS/FS (Wheeler, 2007)

	Debian	Solaris	OpenBSD
Number of Features:	15	11	18
Features score:	6.42	5.92	7.03
Number of Vulnerabilities:	12	21	5
Vulnerabilities score:	7.72	7.74	4.19
Final Score:	-1.0	-3.5	10.2

OpenBSD had the most security features (features that support confidentiality, integrity, availability, or audit), with Debian second and Solaris third. OpenBSD also had the highest score for those features. In terms of vulnerabilities, OpenBSD had the fewest reported vulnerabilities, and those vulnerabilities "were also relatively minor[,] only rating an average of 4.19 out of 10". Solaris, the proprietary system, had the largest number of vulnerabilities. The final rolled-up score is quite intriguing: of the three systems, the proprietary system had the worst security by this rolled-up measure.

The author correctly notes that these are only a few systems, using information taken at only one point in time, so these results are "far from being final". And the author certainly does not take the view that any OSS/FS program is automatically more secure than any proprietary

alternative. Hiding the source code certainly did not reduce the number of reported vulnerabilities, contrary to some proprietary vendors' claims; the proprietary system had the most vulnerabilities reported about it.

A BZ Research survey of 6,344 software development managers in April 2005 asked about the security of different popular enterprise operating environments; OSS/FS did very well. Below are some of the results; the margin of error for the survey is 2.5 percentage points.

Among server operating systems, there was uniform agreement that both Sun Solaris and Linux were much more secure than Microsoft's Windows Server against operating system related attacks. When comparing Sun Solaris against Linux by this measure, There was no consensus as to whether Sun Solaris or Linux were better against operating system level attacks; more people ranked Linux as "secure or very secure" compared to Sun Solaris, yet more people also ranked Linux as "very insecure or insecure" than Sun Solaris.

Table 2.7. Comparing security of Windows, Linux and Solaris (Wheeler, 2007)

	MS Windows Server	Linux	Sun Solaris
Very insecure or Insecure:	58%	6%	13%
Secure or very secure:	38%	74%	66%

Windows Server also did poorly against application-related "hacks and exploits":

Table 2.8. Comparing Windows to Linux (Wheeler, 2007)

	MS Windows Server	Linux
Very insecure or Insecure:	58%	18%
Secure or very secure:	30%	66%

OSS/FS was also far ahead of proprietary programs in 4 of the 8 categories they considered: desktop/client operating systems (44% to 17%), Web servers (43% to 14%), server operating systems (38% to 22%), and components and libraries (34% to 18%). Results were essentially equal in three categories: desktop/client applications, server applications and application servers. Only in one area was proprietary software considered more secure than OSS/FS, database servers (34% to 21%).

Security is notoriously hard to measure, and many reports that attempt to do so end up with interesting information that's hard to interpret or use. And some reports come from sources whose reliability is widely questioned. On November 2, 2004, mi2g reported on successful digital breaches against permanently connected computers worldwide. They concluded that BSDs (which are usually FOSS) and Apple's computers had the fewest security breaches; on the surface, that sounds positive for FOSS. They also reported that GNU/Linux systems had the most breaches, followed by Windows. That result sounds mixed, but digging deeper it turns out that this ranking is artificial, based on artificial definitions. Their default definition for a security breach only included manual attacks and ignored malware (viruses, worms, and Trojans). After all, why bother with a manual attack when completely automated attacks against broad collections of computers will do more? When they include malware in their calculations for all system breaches, "including the impact of MyDoom, NetSky, SoBig, Klez and Sasser, Windows has become the most breached computing environment in the world accounting for most of the productivity losses associated with malware - virus, worm and trojan - proliferation." Even without malware, in governments "the most breached Operating System for online systems has now become Windows (57.74%) followed by Linux (31.76%) and then BSD and Mac OS X together (1.74%)" (a reversal of their previous rankings).

One of the most dangerous security problems with proprietary software is that if intentionally malicious code is snuck into it, such code is extremely difficult to find. Few proprietary vendors have other developers examine *all* code in great detail - their testing processes are designed to catch mistakes (not malice) and often don't look at the code at all. In contrast, malicious code can be found by anyone when the source code is publicly available, and with FOSS, there are incentives for arbitrary people to review it (such as to add new features or

perform a security review of a product they intend to use). Thus, someone inserting malicious code to an FOSS project runs a far greater risk of detection. Here are two examples, one confirmed, one not confirmed:

Some time between 1992 and 1994, Borland inserted an intentional "back door" into their database server, "InterBase", as a secret username and fixed password. This back door allowed any local or remote user to manipulate any database object and install arbitrary programs, and in some cases could lead to controlling the machine as "root". This vulnerability stayed in the product for at least 6 years - no one else could review the product, and Borland had no incentive to remove the vulnerability. Then Borland released its source code on July 2000 as an OSS/FS project. The "Firebird" project began working with the source code, and uncovered this serious security problem with InterBase in December 2000 (only 5 months after release). By January 2001 the CERT announced the existence of this back door as CERT advisory CA-2001-01. Once this problem was found by open source developers reviewing the code, it was patched quickly. Note that this threat is unfortunately a credible threat to proprietary software, because very few of its users can review the code. This is far less dangerous to FOSS software, due to the worldwide review that's possible (including the ability to see the changes made in each version). (Wheeler, 2007)

CHAPTER III

RESEARCH METHODOLOGY

3.0 Introduction

Research could either be explanatory or exploratory. Explanatory research aims not only to describe but also to provide reasons for the development of certain phenomena or the way people act, or to show the impact of activities on client groups. Explanatory research uses quantitative and experimental designs and methodologies (Neumann, 2003). This research has lots of elaboration and as such could be categorized as explanatory, with a few instances of an exploratory approach.

This chapter therefore, presents a background against which data was gathered and analyzed and reasons why those methods were chosen.

3.1 Research methods used

3.1.1 Research Variables

According to the research title, success of electronic business is a dependent variable in the study. System quality and information quality serve as mediating variables between the

FOSS-based application qualities (reliability, availability, usability and security) and the positive feedback from the developing countries.

3.1.2 Research design

The study adopted a descriptive research design that involved both quantitative and non-quantitative primary/secondary data on the challenges of using FLOSS or Proprietary software in Uganda. Basing on this data, comparisons are then made between the software. This is consistent with the research objectives and questions in chapter one.

3.2 Research population

During the preliminary investigations, it was discovered that a big percentage of the population have mild or no information about Free Open Source Software. This altered the direction or choice of the sample space. Therefore, an alternative was sought in which information-literate knowledge workers were identified. Officially, the views of 52 persons were used in this research.

The following are categories of persons that participated in the above collection of data that involved key informants or personalities and users who interact very closely with the system and the organization: Two systems administrators from National Agriculture Research Centre, IT manager from Kazinga Channel (U) Ltd, four computer instructors, six IT students and 20 staff from computer-oreinted business organizations, 10 Information technology knowledge workers as well as four people from the public at various institutions. In case of usability, 4 participants were selected with mild levels of computer skills, but with no or any Linux experience or background.

3.3 Research instruments

Primary and secondary data was collected. Secondary sources included text books, journals, magazines, company records, seminar presentations, articles from newspapers and internet. Primary sources included tests, experiments, interviews and use of questionnaires.

3.3.1 Interview

Interviews involved face-to-face interaction with the respondents and asking them questions in line with the research question were conducted. The interview involved 12 persons of which 10 were Computer instructors, and 2 were Systems Administrators from Kazinga Channel Ltd and Nafirri. This method was chosen because it was the most effective way for the researcher to get a first-hand account of the information on the study especially on the reliability, availability, usability and security of software.

3.3.2 Literature review

Document review involved intensive researching and reading of different authors of literature about the variables of the study. Several reports availed on the Internet were utilized. This method was used as an additional source of information since the researcher could not get sufficient information on the functionality of the software from the respondents especially on comparing a wide range of software products.

3.3.3 Questionnaire

The questionnaires included both closed-and open-ended questions, was pretested to mitigate any possible misunderstandings, questions from existing validated questionnaires and negatively worded questions to gain validity and reliability were used. 52 questionnaires were distributed and all were received back.

The questionnaire contained both a semantic differential scale and a Likert scale, in order to measure user attitudes and reactions by quantifying subject information.

Semantic differential scale is designed to measure the connotative meaning of concepts. The respondent is asked to choose where his or her position lies, on a scale between two bipolar words, or a range of words or numbers ranging across a bipolar position (for example, `Excellent', `Good', Adequate', `Poor', `Inadequate' etc)

On the other hand, Likert scale is a type of survey question where respondents are asked to rate the level at which they agree or disagree with a given statement. (For example: I find this software easy to use: strongly disagree 1 2 3 4 5 6 7 strongly agree)
A Likert scale is used to measure attitudes, preferences, and subjective reactions. In software evaluation, we can often objectively measure efficiency and effectiveness with performance metrics such as time taken or errors made. Likert scales and other attitudinal scales help get at the emotional and preferential responses people have to the design. Is it attractive, fun, professional, easy?

The well designed open-ended and close-ended questions were handed to the supervisor and modified according to his suggestions so as to increase satisfaction.

Questions to measure satisfaction, interoperability, reliability, scalability, usability and security were included in the questionnaire. All the items under study had one or more respective questions to test the satisfactorily defense of open source software use. Extreme care was taken to provide a friendly questionnaire so as to suit the seemingly more informed persons as well as the newly exposed to the ICT field. (*Refer to appendix iii for a copy of the questionnaire used*)

3.3.4 Test

An experimental test was done to observe challenges and benefits in the application of different software products and their alternatives. An OOBE (Out-Of-Box Experience) test was designed to test the usability of Free Open Source Software.

This section addressed the Out-Of-Box Experience (OOBE) usability issues of Free and Open Source Software (F/OSS) considering outcomes of distributed development process and high number of available product choices.

A methodology is presented below and usability experiments were conducted whose results are discussed later in chapter four. The objective was to determine factors that affect usability of F/OSS during OOBE and first hours of use.

Methodology of test

The test aimed at identifying key factors in OOBE of F/OSS in a laboratory user experiment. For that purpose, emphasis was put on an open source system software. A task list was designed to include installation and basic operations that match OOBE definition.

Designing the Task List

The scenario given in the task list (Table 4.3 below) covered the process from opening the box through using and experimenting the basic productivity tools and software. The task list is given in Table 5. A more explanatory version was given participants as task scenario sheet.
Before the test began, participants were given 20 minutes to warm up and get accustomed to the work environment and a different user interface. The subjects were free to comment on the system and ask for help from the facilitator in this time period.
No hints or assistance were given to complete the tasks.

Table 3.1. Task Descriptions

Task	Explanation
T1	Unpack the Linux
T2	Power on the computer with CD inserted
T3	Install the system including most common components
T4	Reboot and log in into the system
T5	1. Browse the web with a browser. Go to www.google.com,search for an image and save it to the desktop 2. Work with a word processor. Create a document with a heading 12pt and default body font of 10pt. 3. Send an e-mail: Open the mail client and connect to the existing e-mail server if any is available. Send an email to a friend and read any received ones 4. Listen to music: Listen to an artist of your choice by inserting a CD to the driver and adjust the volume if necessary
T6	Log out and power off the machine
T7	Try to get online assistance(if needed)

F/OSS Software: A Desktop Operating System

The facilitator used a desktop centric Linux "Xandros" mainly to fit the testing purposes. Xandros was chosen because of its relatively inflexible installation for new users.

Software components were preselected carefully by the company and for each desktop task, there was only one corresponding software thereby reducing the clutter of ordinary Linux menu system. Moreover, complete set of desktop productivity software was installed in default configuration. The participants were given a full boxed set of Xandros Linux version 2.5. The box included a comprehensive user guide, a bonus CD with applications, games, tools and 60 days of e-mail support.

The system requirements recommended by Xandros OS have been satisfied at a higher level such as 128MB recommended versus 512MB available RAM on the laboratory computer.

Participants and Selection Procedure

During planning stage, the facilitator tried to make sure that the users, tasks and the environment represented the intended context of use. Four (4) participants were selected (2 male and 2 female) with varying levels of computer skills, but with no or any Linux experience or background. All of them were told the reasons, the aim and possible outcomes of the test before the laboratory session begins. and were given time to ask general questions about the study. They were given to complete a pre-text questionnarie asking personal questions and background computer and operating system experience.

Participants of the study (subject group) were occasional users who have heard of F/OSS and Linux before, but none of them had a compelling experience with a Linux distribution. All of them had Internet access at work/school and use computers very often in their everyday life and classified themselves as "casual Microsoft Windows users with the ability to install an operating system and connect to the broadband internet".

During the participant selection procedure, the following items were kept in mind and the participants were screened accordingly: The candidate participant should,

- have Internet and network configuration skills
- be familiar with a graphics package
- have used a word processor and Google search engine before
- have a working e-mail account
- have used an instant messaging application before
- have an excellent knowledge of English computer terminology
- All the participants were part-time or full-time workers, with 22-28 years of age.

3.4 Data Processing and Analysis

This was done by making references to the available literature in order to compare and contrast different opinions as presented by different authors. These opinions and quantitative data were then analyzed to arrive at the conclusions of the study as is reflected later in the findings.

3.4.1 Data Analysis Tools

Non-quantitative data was based on opinions expressed by the respondents, observation and experiments, was analyzed by comparison to achieve the conclusive results of the research. Analysis of collected data was done by use of Microsoft- Excel.

Other tools used include: Hardware such as Laptops, Desktops, Server-ware, Network Interface Card as well as Software like Mozilla Firefox, Internet Explorer, Apache Web Server, Windows 2007, NT, 2003, XP and GNU-Linux products like Xandros.

3.4.2 Data Presentation

The findings and the information in this study were presented by use of quantitative and non-quantitative information through narration and providing the statistical results from the experimental tests. This research often visited the works of other authors, it scrutinized their opinions and thus, compares their analysis to the findings of the research through the experiments, observations and interviews performed by the researcher.

3.6 Limitations of the Study

Uganda being a developing country, identifying a company that employs e-business facilities was next to impossible. Even the few individuals that were interviewed were not so knowledgeable about software details, putting into consideration that our society has limited IT scholars who endeavor to venture in software engineering or programming.

Some respondents were not willing to disclose information especially about their core servers for security reasons and company policy. This was a major problem in the findings, as

the researcher could not get sufficient information which could have gone a long way to shed more light on the study.

It was literally impossible to attain accurate or reliable results from the OOBE test, with Xandros since, the researcher could not provide new intact Xandros packages for the participants.

The researcher could not also get source code to experiment as most software available was not licensed.

The time constraint was another limitation since the researcher had to study, teach and give tests, while compiling this report. But this was solved by proper following of the project time line.

CHAPTER IV

DATA FINDINGS, PRESENTATION AND ANALYSIS

4.0 Overview

The research was collected from sample size 52, composing 56% male and 44% female with the majority between the ages of 25 and 40 years. This age group could be explained by the fact that Information Technology is still an immature field in Uganda and hence very few people of the elder sector are enthusiastic about it. This chapter outlines the summary of the research results as reflected on the questionnaires. The detailed discussion of these results is tackled in chapter five.

4.1 Preference of category of software

As reflected on the graph below, 32 correspondents use both proprietary and Free Open Source Software, 20 predominantly use proprietary software of which Windows XP is the most common. It was also noted that no correspondent uses Free Open Source Software independent of proprietary software, as indicated in figure 3.1

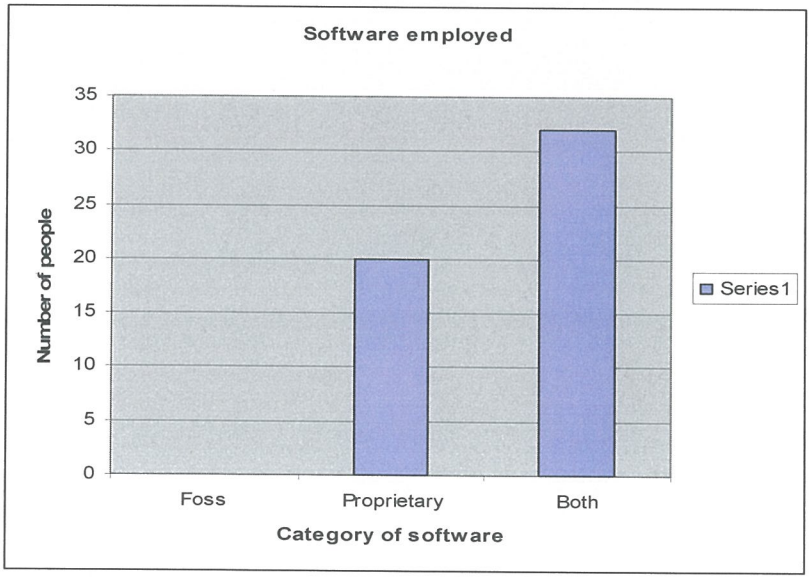


Figure 4.1. Comparing category of software used

4.2 The rate of paying software licenses

After discovering that a large percentage of Software users prefer proprietary material, one would expect that lots of revenue is received from licenses. Unfortunately, many of these people would rather pirate the software other than ay licenses because they claim that its very expensive. Levels of license payments are indicated in table 4.1 as well as the equivalent graph in figure 3.2.

Table 4.1. Payment of software licenses

Software type	Number of license Payers	Non-license payers	Yes/No responses
Proprietary	8	10	2
Foss	0	0	0
Both	0	28	4

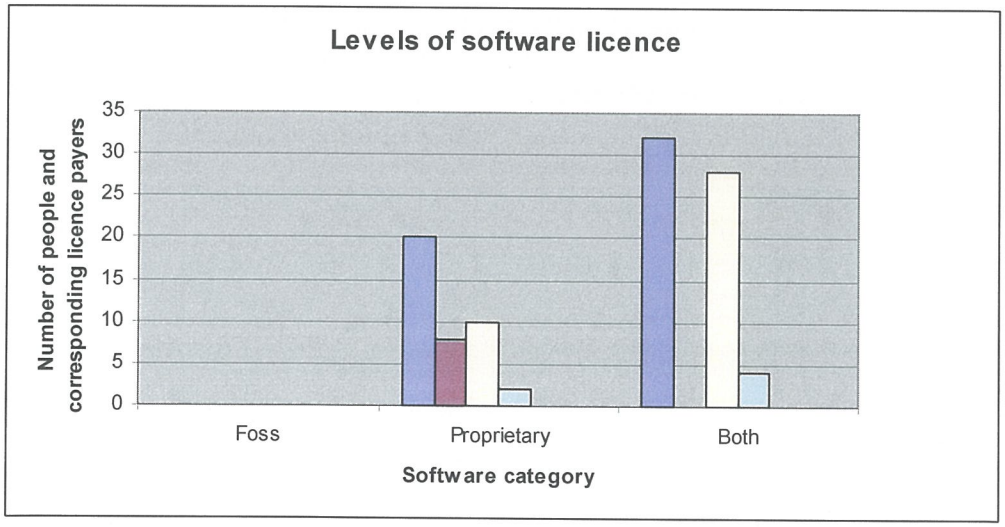


Figure 4.2. Payment of software licenses

Table 4.2. Key for figure 4.2 above

Series 1	software type
Series 2	number of licence payers
Series 3	No licence
Series 4	yes/No

4.3 Degree of Information disintegration about FOSS among users

The research showed that only 31% of the correspondents have some access to information or had an idea about what FOSS is about, while 57% had no knowledge whatsoever about FOSS. 12% had never put any serious thought about this thing called FOSS as shown in 4.3. However, a land slide of 84% admitted that they were willing to adopt FOSS if availed with enlightment about the cons and pros of FOSS. This is recorded figures 4.3 and 4.4 in the pie charts below.

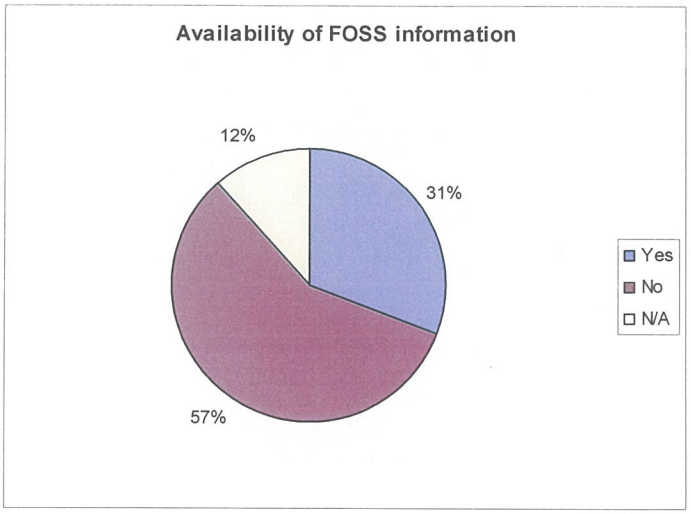


Figure 4.3. Effect of information disintegration on FOSS adoption

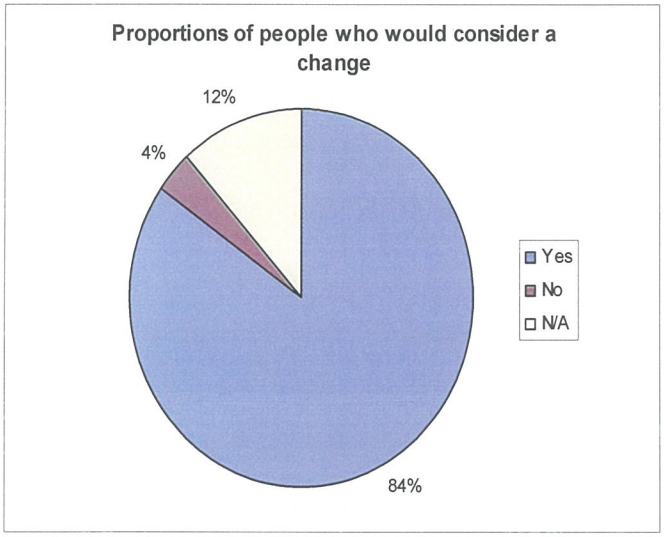


Figure 4.4. Effect of information disintegration on FOSS adoption

4.4 Analysis of commonly used software

Basing on the different software that correspondents mentioned earlier in the Questionnaires as their favorite environment, the researcher spotted some features that could be a promoting factor feature for individual software. The correspondents were asked to weigh them according to the level comfort of use. It's a cumulative frequency that was used to calculate the various weights. Below is an analysis of the different features as graded by the users or correspondents. The software that scored highest and lowest are indicated at the bottom of every graph.

4.4.1 Ease of use

In software engineering, its important that the product offered to the market both satisfies uer's requirements and also the user should be in position to interact or interface with the program comfortably. Therefore, this element of ease of use tested the user friendliness of the different programs employed by the correspondents of this research; as reflected in figure 4.5 below.

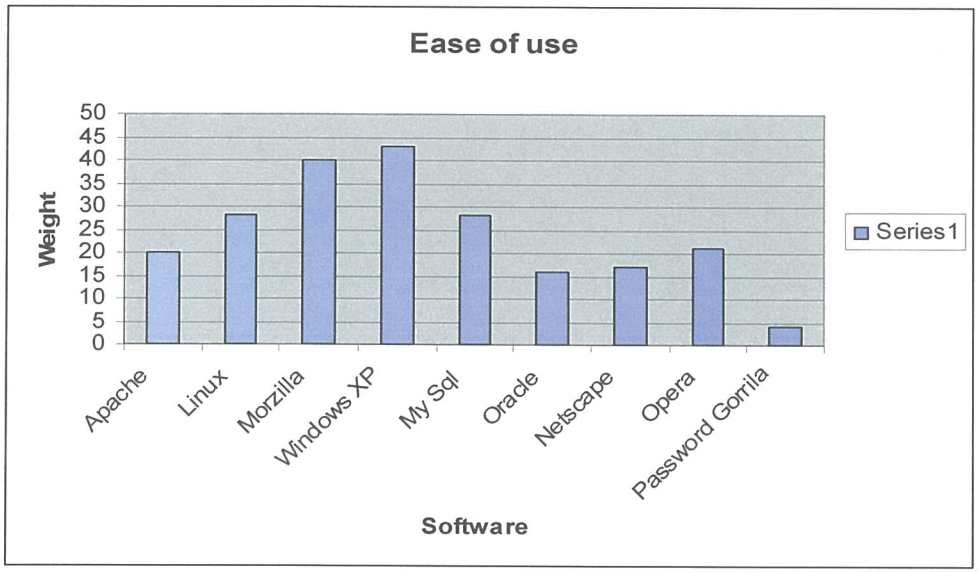


Figure 4.5. Ease of use

Highest: Windows XP

Lowest: Password Gorilla

4.4.2 Program Stability

Stability of a program is usually a function of its scalability. This reflects to what degree a software can function considerably well even when exposed to adverse conditions of which it was not specified for originally. Linux and Windows were graded at the same levels as represented in the graph below in figure 4.6.

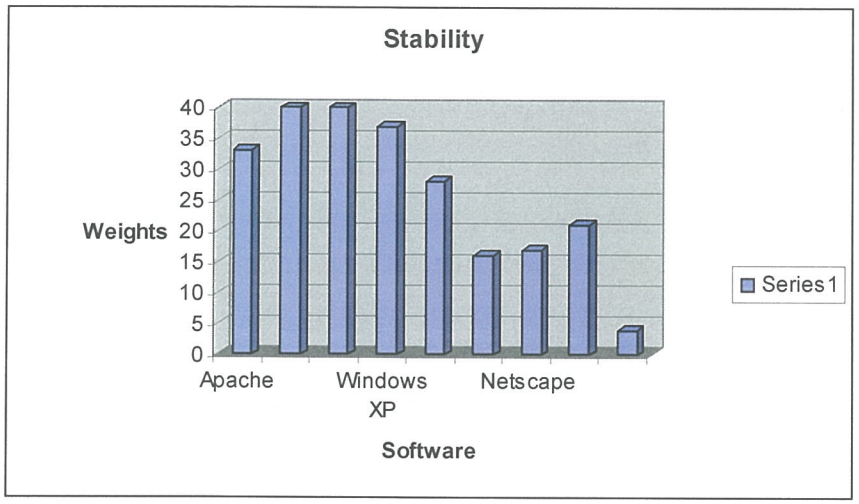


Figure 4.6. Software stability

Highest: Linux

Lowest: Password Gorilla

4.4.3 Start-up ability

Start-up ability tested the rate at which a software takes to log on or display the initial interfaces with the user. The results are reflected in the graph below that is, figure 4.7

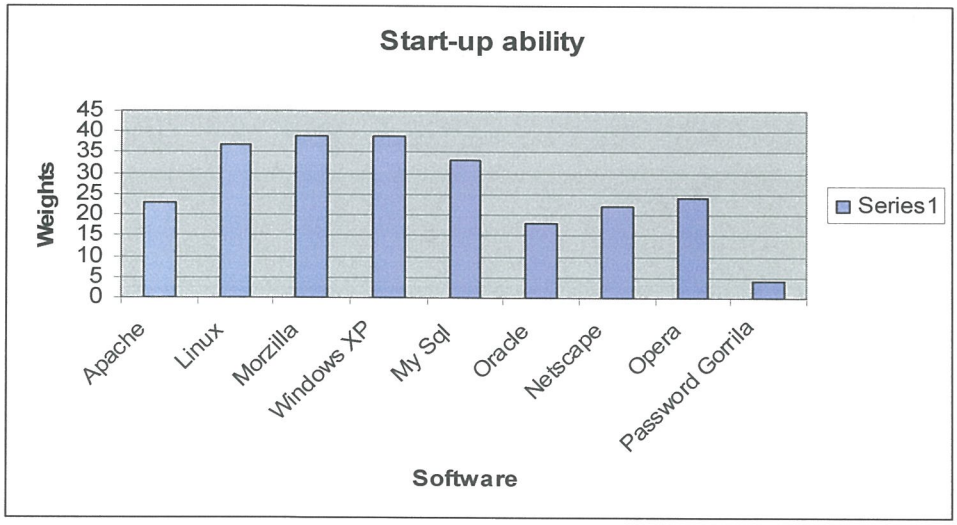


Figure 4.7 Software start-up stability

Highest: Mozilla

Lowest: Password Gorilla

4.4.4. Software reliability

Reliability is popularly measured by how long software can resist crashing or freeze-ups from the time it is installed. Below is figure 4.8reflecting the correspondents views about the various software that are commonly used.

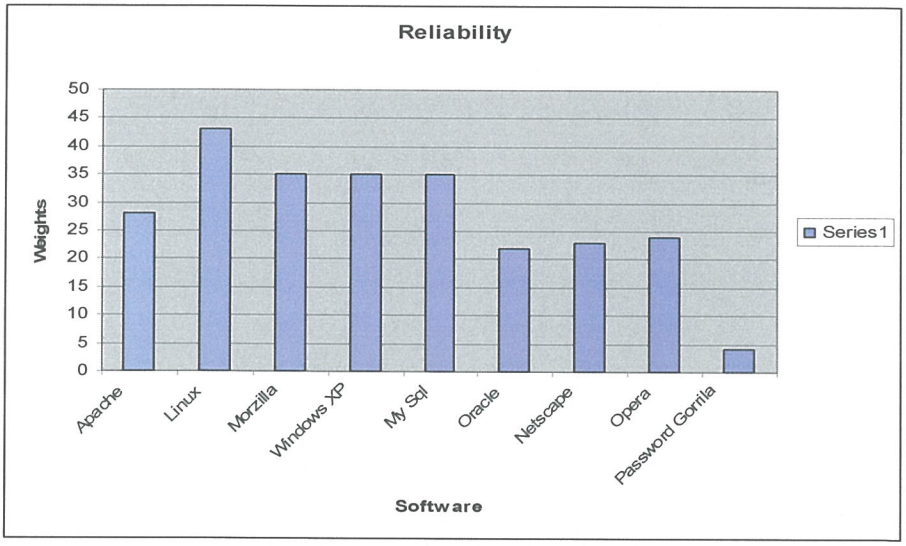


Figure 4.8. Software reliability

Highest: Linux

Lowest: Password Gorilla

4.4.5 Software accessibility or availability

Any excelling business is one that not only offers a buyer a product of sound quality but it must be at the right time and in the right place. This research sought to assess how near the software are to the users. That is, with how much does it take them to get hold of the software or even the information about their software of interest. The results are as shown in figure 4.9 below.

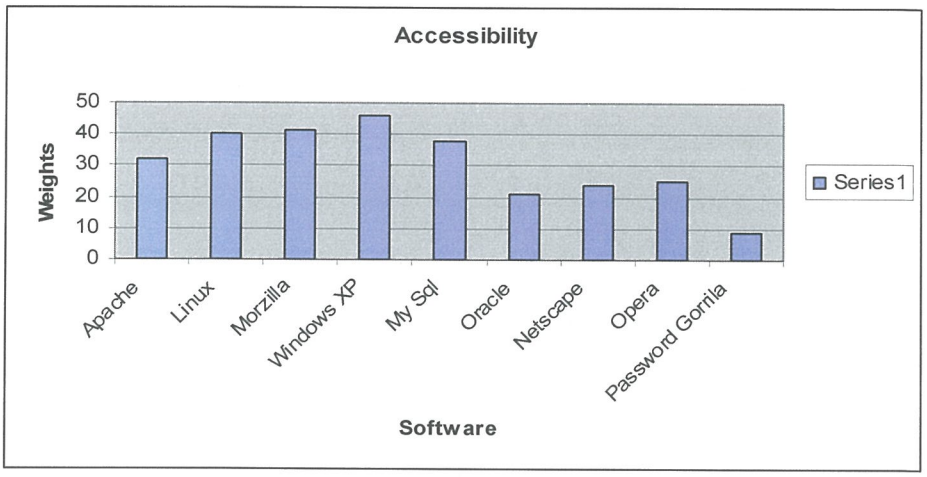


Figure 4.9. Software accessibility

Highest: Windows XP

Lowest: Password Gorilla

4.4.6. Security levels of Software

Quantitatively measuring security is very difficult. Security is often measured by a survey of opinions, from people who are informed and have use used the software because it is hard to measure. Though opinions can be wrong it is very hard to ignore the opinions collected from a large sample space of users who are in the know. The results are as delineated in figure 4.10 and table 4.3 below.

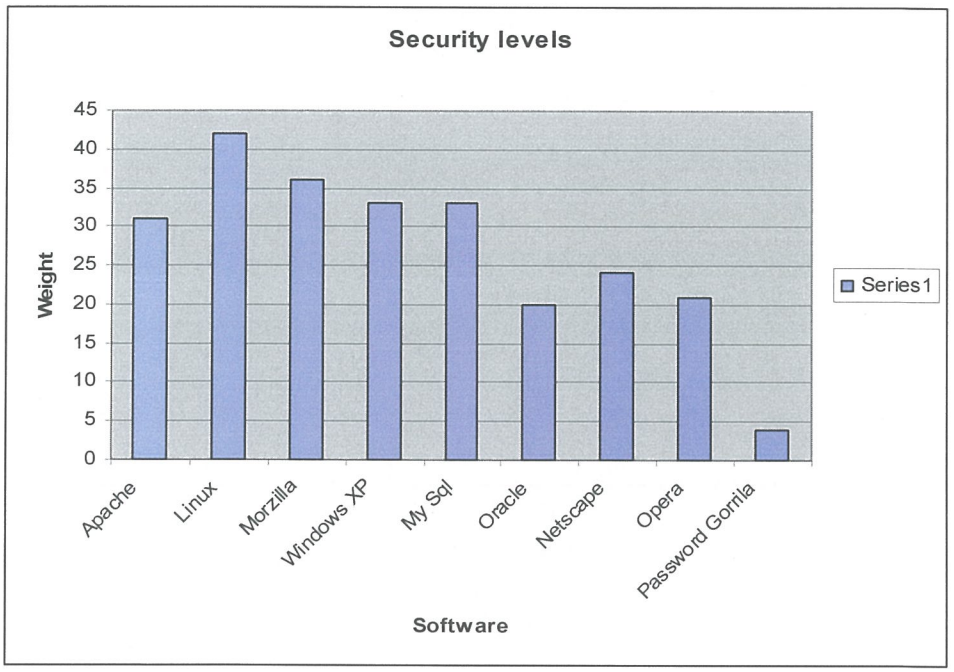


Figure 4.10. Levels of software security

Highest: Linux

Lowest: Password Gorilla

Table 4.3. Comparing the security of Linux and Windows XP

	Windows XP	**Linux**
Very secure/Secure	37.5%	62.5%
Very insecure/insecure	78.2%	21.8%

4.4.7. Interoperability

Interoperability refers to the ability of a software to function concurrently with another software under the same environment or circumstance. For example one would wish to know if Linux can tolerate Oracle or if they can function under similar hardware specifications. Figure 4.11 below expresses the correspondents' view of what software seems more interoperable.

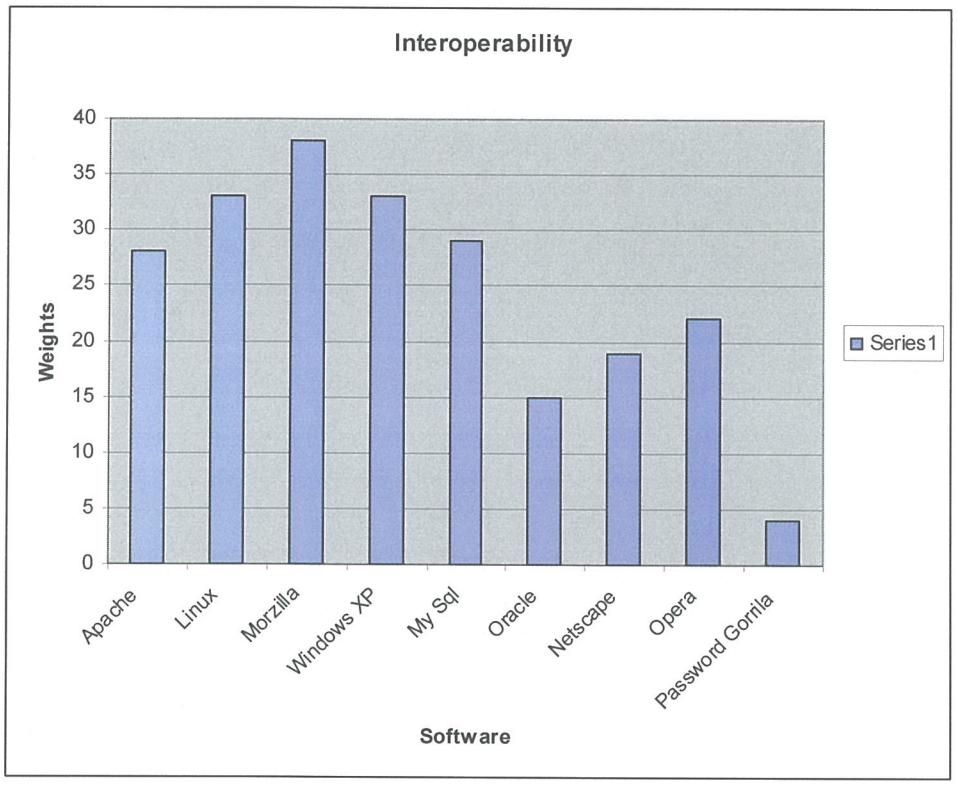


Figure 4.11. Comparing the interoperability of several software

Highest: Morzilla

Lowest: Password Gorilla

4. 5. Usability (results from the test)

Results were obtained by noting down the participants' attitudes and vocal reports, and taking measurements of task completion times. Results for each particular task were examined as follows.

Table 4.4. Key

Participant' Symbol	**Name**
P1	Pauline
P2	Peter
P3	Peninah
P4	Paul

Box Appearance and Appeal

The distributor company chose to design traditional off-the-shelf software box aiming to establish brand recognition same as other branded OS manufacturers. All the participants except P4 could easily open the box and examined the contents. The boxes were closed with an adhesive label where P4 pointed out that the box was tricky to open and take the book and CD's out. None of the participants had examined the back cover containing features and advanced support options. This may result from the lack of sense of possession on the box. P3 was very satisfied with the book's detail level.

Installing the Operating System

The respondents had found the installation to be intuitive, fast and entertaining, without exception. All participants but P4 chose express install, allowing them to install operating system in 5 clicks. One participant P1 requested an installation floppy from past experiences, later realized that the box contained none and inserted the installation CD. Also, only Pauline had created an ordinary account and used this account. P2 tried to insert the application disk, however got no response and had to ask the test coordinator what to do. P3 commented out that it

was effortless to install without a driver disk requirement. The terminology used looked adequate (i.e, not technical),however P1 asked what a /dev/hda2 meant. Only one participant P2 found it interesting to follow the notices appearing on the installation screen about security and stability, and others skimmed the users guide.

The duration of installation process varied from 12 minutes 27 seconds P2 to 31 minutes P4. The reason it took long for P4 was that, he rebooted the machine thinking that the machine hung during the installation initialization process. The mean value for the installation task was 16 minutes and 42 seconds. Different stages of the installation are delineated in figures 4.12, 4.13, 4.14, 4.15 below.

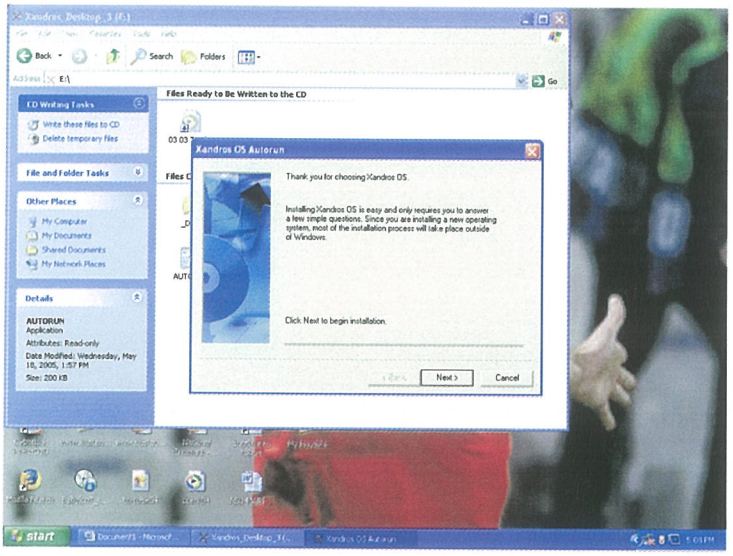
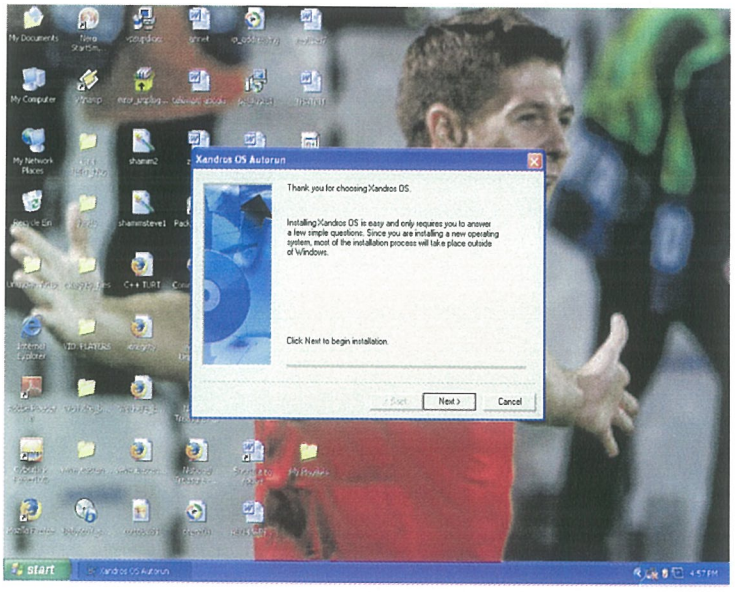


Figure 4.12. Installation Screen



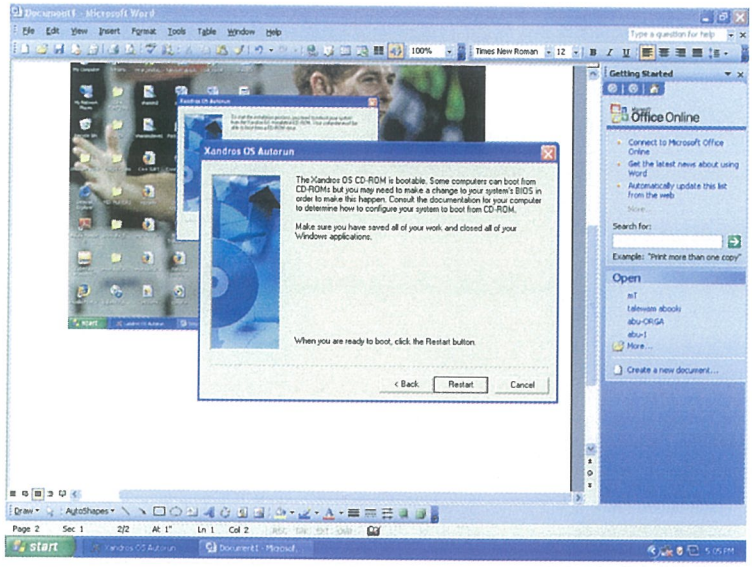


Figure 4.14. Installation Screen

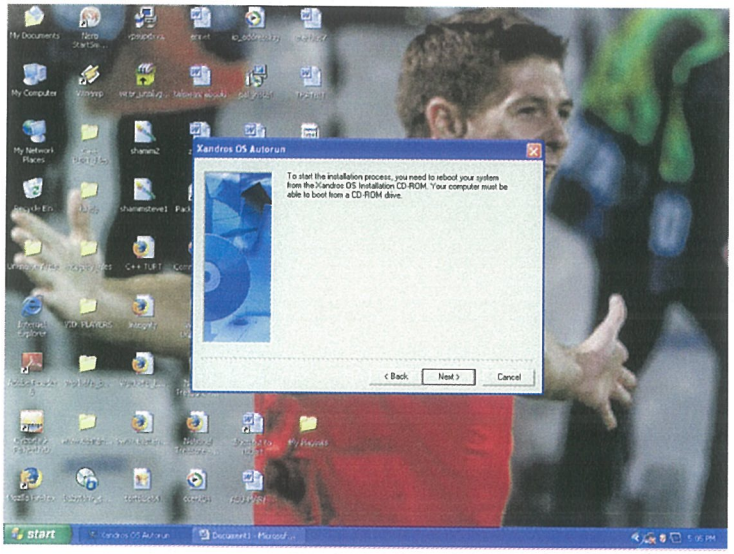


Figure 4.15. Installation Screen

4.6 Findings

With the use of Linux and Morzilla as representatives of FOSS in the above analysis, FOSS tools score highly as reflected in the various graphs. However, the windows environment is still appealing due to its user-friendliness, thus the ease of use. Password gorilla is quite new on the scene of anti- viruses and I think that explains its low scores.

4.7 Answer to research questions

Basing on the several presentations above as collected from the research, one can confidently say that FOSS tools can develop dependable applications and present reliable, secure and user-friendly content for on-line businesses. However, very few businesses have adopted Open Source software tools due to poor information integration about the same and also there is an attitude of assuming piracy is a norm in developing countries of which Uganda is part and thus Kampala International University as well. As much as FOSS tools could be availed to the public, its also necessary that maybe copyright measures be put in place by government to guard the intellectual property of the "Closed" software.

CHAPTER V

DISCUSSION, CONCLUSION AND RECOMMENDATIONS

5.0. Overview

This chapter explains a short overview of the different features of free /open source software that have been the core of this study. Also noted in here are a few factors that must be considered before one assumes that adoption of new ideas or technology could be a walk over. The chapter closes with the researcher's overall perspective about taking on FOSS other than proprietary software.

5.1. Discussion

5.1.1. Usability

When the respondent has previous experience with other similar products, the respondent's expectations, existing factual and procedural knowledge normally affect the initial usability experience to a great extent. When there are no easily reachable alternatives of a particular tool, a user is subjected to its available usability. He eventually has to proceed forward after the initial learning phase regardless of a tool's usability characteristics.

OOBE of F/OSS is significant in software usability perception and possible adoption. User experience, visible structure, consistency and functionality of the interface had significant impact on OOBE and first hours of use.

Basing on the results from the study, in light of the fact that many users are not exposed to FOSS, I strongly believe, OOBE is highly dependent on fundamental usability principles such as visibility, consistency, affordances, locus of control and feedback allowing a user to start using the product immediately. Therefore, keen users won't find any hurdles in adopting FOSS tools for their day-to- day content development.

5.1.2. Reliability

There are a lot of anecdotal stories that Proprietary software is more reliable, this research has provided data confirming that mature FOSS programs are often more reliable. This is clearly evidenced in the high percentages or weights of Morzlla(35), Linux(43) and Mysql(35).Given the fact that Windows Xp is very popular among computer users, it remains quite a mystery to why its was not considered reliable by a great number of research correspondents..

Equivalent FOSS applications are more reliable than proprietary software according to Fuzz report that shows evidence that Windows applications have even less reliability than the proprietary Unix software. Also, according to a 3-month Swiss evaluation, the difference between Netscape (proprietary) and Apache (FLOSS) is statistically insignificant.

IBM studies put Linux to a 10-month test and ran Caldera Systems OpenLinux, Red Hat Linux, and Windows NT Server 4.0 with Service Pack 3 on duplicate 100MHz Pentium systems with 64MB of memory found GNU/Linux highly reliable. The NT server crashed an average of once every six weeks yet none of the Linux server ever went down.

5.1.3. Security

One of the interviewee Julius K from Kazinga channel commented, "on responses to security problems, Linux (an OSS/FS vendor) responded more rapidly than Microsoft or Sun to advisories".

From the data collected from the questionnaires, (*as presented in fig. 4.6, page 60*) Mozilla Firefox fixes its vulnerabilities faster, and has fewer severe vulnerabilities than Internet Explorer.

According to a June 2004 study by Sandvine, 80% of all spam is sent by infected Windows PCs. National Cyber Security Alliance's study of May 2003 reported that 91% of Broadband users have spyware on their home computers running proprietary operating systems.

In contrast, there's no evidence of that this is an issue for FOSS systems. National Cyber Security Alliance's study of May 2003 reported that 91% of Broadband users have spyware on their home computers running proprietary operating systems. In contrast, there's no evidence of this issue for OSS/FS systems.

Microsoft has had far more vulnerabilities than anyone else, according to Security Tracker Statistics paper (March 2002) that analyzed vulnerabilities from April 2001 through March 2002. In late June 2004, the U.S. Department of Homeland Security's Computer Emergency Readiness Team (CERT) recommended using browsers other than Microsoft Corp.'s Internet Explorer (IE) for security reasons because Microsoft had failed to patch a critical vulnerability for 9 months, and IE was being actively exploited in horrendous ways.

5.1.4. Accessibility or Availability

Taking into account all major cost elements, Open Source Software environments are significantly less expensive than proprietary software environments, but proprietary software is still widely in use. This is clearly illustrated by the great numbers of people who are illegally using proprietary software yet are so much willing to switch if given an alternative that is affordable financially. Even the correspondents who admitted that they succumbed to paying licences would prefer cheaper options if given chance to. However, acquisition of the new emerging and rapidly evolving technologies as well as their installation, maintenance and full exploitation is beyond the reach of many businesses in developing countries like Uganda. This could be attributed to infrastructural, technological, and /or financial impediments. There has been, nevertheless, a growing trend in many companies to introduce these technologies at least to levels of fixing stand-alone Pcs or installing simple LANs.

While the problems faced by developing countries are clearly of a different magnitude, and often of a different nature, to those affecting developed countries, there are many issues which are common to them all.

The principal factors with reference to the implications that these obstacles have are:

- **Costs:** ICTs constitute a very expensive resource even in industrialized countries where the necessary infrastructure for their installation is in place. The price of hardware, although constantly decreasing, remains considerable for developing countries as well as software.
- **Lack of professional competence/ Technical Expertise:** Professional competence is considered to be the single most important factor ensuring the successful use of ICTs in business (Dr. Hashim Twaakyondo, 2006).He continues to explain that its importance has tended to be overlooked or underestimated in the development of initiatives for introducing ICT business.

5.2.Conclusion

The above observations are not merely opinions as these effects can be shown quantitatively, using a wide variety of measures. Using the measures above, it was discovered that Open Source Software is often the more reliable software, and has better security, perhaps due to the possibility of worldwide review.

Realizing these potential FOSS benefits may require approaching problems in a different way. This will require an understanding of the differences between the proprietary and FOSS models. FOSS products are not the best technical choice in all cases, even organizations which strongly prefer FOSS generally have some sort of waiver process for proprietary programs.

Before deploying any program one needs to evaluate how well it meets the user's needs, yet most organizations do not know how to evaluate software programs in general, specifically FOSS or proprietary software.

Therefore, its worth noting that most organizations are switching to FOSS because cost is a significant factor driving adoption of open source software; control and flexibility are considered benefits; Implementation of open solutions is evolutionary, not revolutionary; Open source extends across the entire software stack and Product support is not a significant concern.

5.3 Recommendations

Basing on the findings of this study with an edge of reliability, availability, usability and security. OSS/FS options should be carefully considered any time software or computer hardware is needed. Organizations should ensure that their policies encourage, and not discourage, examining OSS/FS approaches when they need software.

Proprietary software is predominantly being used in several Companies, KIU-ICT inclusive instead of Free Open Source Software for which they usually do not pay Licenses. The researcher would therefore recommend the use of Linux OS versions, Morzilla firefox, Opera and any other software that does not promote piracy.

Nevertheless, Proprietary software cannot be completely done away with. For example, the Windows family has proved to be very user friendly which attribute causes a smile on every user. So the researcher recommends that the Government institutes some laws that will protect the intellectual rights of such software by encouraging the society to fulfill any legal obligations before attaining accessibility to the same.

However, the following points need to be considered if FOSS is to be taken on.

i. Innovation is adopted gradually not immediately (Lissoni and Metcalfe, 1994)

When starting this research project , the researcher realized that there is still a wide digital divide in Uganda and hence the different business sectors as well, yet the Internet has already been commercially available for nearly 10 years. So we all need to appreciate that taking on FOSS tools wont happen over- night but time will tell.

ii. Wider spread of information and knowledge influences the adoption and diffusion of innovation (Metcalfe, 1994; Hermann-Pillath and Lehmann-Waffenschmidt, 2001)

A higher level of information and knowledge diminishes not only uncertainty about a new innovation and whether or not to adopt it, but also reduces the level of risk related to the adoption. And the more firms within an economic system adopt the innovation, the broader the diffusion will be. Information about the advantages of a new technology are therefore very important to enable organizations to make the right decision over the longer term. It's therefore, important that people are availed with FOSS information as well as regular training.

iii. *Structure of companies and industry sectors changes long term processes* (Nelson and Winter, 1982)

The long-term processes of changing economic structures is exactly what Ugandan companies in several sectors must deal with at present. The Internet and other emerging technologies (especially those enabling mobile business) and new market entrants force all players in these industry sectors to adapt to the changing structures to ensure their ongoing economic existence. This leads to the core of this study. Internet technology has such a major influence on the success of the companies that they have been shaken to their core.

The Internet calls for quicker reactions in highly competitive complex markets, where less availability of time and money require significant endeavor to succeed; and where competition between products, services, companies, economic systems and stakeholders increases daily. Free Open Source Software products must be adopted by companies to ensure increased profits.

5.3.2 Future areas of study

Due to unlimited time and several un- avoidable occurrences the study was not exhaustively done or it would rather be regarded impartial. Its for this reason that the researcher suggests that further studies be conducted in the following areas:-

- The role of FOSS tools in adoption and innovation of Internet business models. In general, IT enables Internet business models, but it also converts IT innovations into economic value (Chesbrough and Rosenbloom, 2002). Without an adequate business model, an innovation cannot be made profitable.

REFERENCES

A. Books

1. Dix, A., et al (2003).: *Human Computer Interaction.* Prentice-Hall, Englewood Cliffs
2. Gilbert, A.L (2005).: *Personal and Ubiquitous Computing*, vol. 9(4), pp. 198–208.
3. Springer, London (2005)
4. Pirhonen, A(2005).: *Supporting a User Facing Novel Application*: Learnability in OOBE, Personal and Ubiquitous Computing, pp. 218–226. Springer, Heidelberg
5. Weisberg, H.F., Krosnick, J.A., Bowen, B.D. (1996): *An introduction to survey research, polling, and data analysis.* Sage, Thousand Oaks, CA
6. Woodworth, R.S (1961).: *Experimental Psychology*, Revised Edition, Holt Rinehart

B. Journals

1. Bevan, N. (1999): *Common industry format usability tests.* In: Proc. UPA Conference, Scottsdale, Arizona, vol. 5
2. David A. Wheeler, (2007). *Why Open Source Software / Free Software (OSS/FS, FLOSS, or FOSS)?*
3. Justin E. Forrester, Barton P. Miller. (2000, August). *An Empirical Study of the Robustness of Windows NT Applications Using Random Testing*, 4th USENIX Windows Systems Symposium, Seattle.
4. Ketola, P. (2006): *Out-Of-Box Experience and User Support*, Nokia Telecmmunications
5. Kogut, B(2001).: *Open-Source Software Development and Distributed Innovation.* In: Oxford Review of Economic Policy, vol. 17(2), pp. 248–264. Oxford University Press, Oxford
6. Kowalski, L. (2001): *Designing the Out-of-the-box" Experience*: A Case Study. In: Proceedings of STC
7. Mockus, A., et al. (2000): *A Case Study of Open Source Software Development*: The Apache

8. Server. In: 22nd International Conference on Software Engineering

9. Nielsen, J(1993).: *Usability Engineering. Morgan Kaufmann Publishers*, San Francisco

10. Sibusiso Sibisi, et al. (2004). *Free/Libre & Open Source Software and Open Standards in South Africa.*

C. Published and Unpublished materials

1. Li Ge, et al. (2003, December). *Putting Linux reliability to the test.*
2. Paul Murphy. (2005, November). *Unix beats Windows.*
3. Presserelease. (2000, February). *Sites utilizing Microsoft Webserver-Software turn in inferior reliability results in study of Swiss websites.*
4. Robert Lemos. (2004, December). *Linux lasting longer against Net attacks.*

D. Internet resources

1. eWeek. Linux Watch. (October, 2007). *13 reasons why Linux should be on your desktop*

   Website: http://www.eweek.com/article2/

2. FOSSFA: http://www.fossfa.org

3. FOSSFA'S Action Plan:
   http://www.wougnet.org/ICTpolicy/docs/FOSSFA_ACTION_PLAN.rtf

4. Project Database: http://www.fossfa.org/database/

5. Netcraft. (2007). *Netcraft Secure Server Survey.*

   Website: http://uptime.netcraft.com/

6. Richard Gooch. (12 Nov 2006). *The linux-kernel mailing list FAQ.*

   Website: http://www.tux.org/lkml/

7. Wikipedia. Free Encyclopedia. (2007, October) Open-source software

APPENDIX I

QUESTIONNAIRE USED

This questionnaire is conducted as part of a study at Kampala International University focusing on the use of Free Open Source Software (tools) in content development, analyzing their security, usability, reliability and interoperability in e- businesses. There is no commercial interest driving this questionnaire whatsoever and the collected data is intended purely for academic use. Your response shall be treated as confidential; however, the aggregated results would be made available to you if requested for. I would be grateful if you would kindly spend 10 minutes to respond to this questionnaire.

NSTRUCTIONS: Circle the appropriate point or tick besides your choice

Keep structured answers brief

Use the space provided.

Preferably use bold letters or black ink

SECTION A

PERSONAL BACKGROUND

. Name (Optional).....

. Gender:

- Male
- Female

. Age group:

- 20-29 years
- 30-39 years
- 40-49years
- 50-59 years

. Academic qualifications:

- O'level and below
- A' level
- Post secondary education
- University education
- Postgraduate and above

Employment

Are you working? Yes..... ... No.....

. Field of profession

- Education
- Music
- Business
- Political
- Medical
- Student(Specify area of study)
- Other

. How long have you worked with that organization/ Company?

- Less than a year
- 1-2 years
- 3-5 years
- More than 5 years

. What is your staff designation? [Feel free to indicate departmental post e.g. IT manager]

- Support staff
- Staff
- Supervisor
- Manager

SECTION B

GENERAL COMPUTER KNOWLEDGE OR USAGE

ART ONE

a. Have you used computers before? Yes .....No.....
b. In a normal week, how much time do you spend using the Internet?
    - Less than 3 hours
    - 3-8 hours
    - More than 35 hours
    - I don't know
    - I never use it
c. How technical are you?
    - Not very
    - A little less than average
    - More than average
    - Very technical

d. I spend time online mostly for:-

Work .....Pleasure.....

e. What things do you do most when online? (Numerically check your top three-1,2,3
   - o Communicating with others
   - o Work/business
   - o Research for personal use
   - o Games
   - o Education/school work
   - o Web development
   - o Shopping (gathering product information)
   - o Entertainment

f. Which web browser is more appealing to you?
   - o Opera
   - o Netscape Navigator
   - o Mozilla Firefox
   - o Internet Explorer
   - o Other

RT TWO

a. Does your organization invest in IT, Hardware, software, database and networks?.....

b. My organization provides education and training in IT to management, end-users, customers and other business stakeholders.

- o Regularly
- o Rarely
- o Have never

c. Your company needs to go online. I.e. interact with both suppliers and customers via intranets and extranets (Internet).
   - o I strongly agree
   - o I do not agree
   - o Am uncertain

d. Brief defend your view in (c)

.....

.....

.....

e. Integration of IT based systems to manage the regional businesses would:-
   - Positively impact the company (e.g. Increase the number of customers)
   - Negatively impact the company (e.g. increase the company's expenditure)

PART THREE:

To be filled by correspondents dealing directly with the computer sector or applying its technology in their day-to-day activities.

a. Title
   - IT manager
   - Systems Administrator
   - Security Administrator
   - Network Administrator
   - Computer instructor
   - Programmer
   - Student
   - Other

b. How long have you held that position?.....

c. Our Company uses
   - FOSS (Free Open source Software)
   - Proprietary software
   - Both

d. Could you mention some of those software? FOSS or PROPRIETARY

e. Have you purchased licenses for each of your software copies? If not why?
   .....
   .....
   .....
   .....

f. Is information about benefits and shortcomings of both FOSS and proprietary software readily available to you? Yes.....No.....

g. If not, if more information were to be availed to you, would you consider changing your current software and opt for the alternative? Yes.....No.....

h. Is proprietary software fundamentally better supported than FOSS?
   Yes.....No.....

i. Does proprietary software give users more legal rights than FOSS? Yes.....No.....

j. There is ample evidence that FOSS encourages, and not quashes innovation.
   - I strongly agree
   - Do not agree
   - I am uncertain

k. Will FOSS destroy the software industry? Won't programmers starve if many programs become FOSS?
   - Most likely
   - Unlikely

l. Is FOSS economically viable? Yes.....No.....

m. Does FOSS expose you to greater risk of abandonment? Yes.....No.....

n. For each of the following feature, please tell us well does it work on the stated software by using a scale of five(5).

Key:

1-very poor,
2- poor,
3-ok
4-Good
5-Very good
X-Have not used it before

e/Characte ristic	Software								
	Apache server	Linux	Mozila	Windows XP	MySQL	Oracle	Netscape	Opera	Password gorilla
and feel									
ıse									
lay(start up)									
y									
ility									
rability									

o. In your own wording, please tell us if FOSS tools are worth a try in e- businesses and why.....
.....