OPEN SOURCE SOFTWARE AS A VALUE ALTERNATIVE TO COMMERCIAL SOFTWARE

by

FRITS KOK
B.Sc Hons (Computer Science)

SHORT DISSERTATION

submitted in partial fulfillment (25%) of the requirements for the degree of

MAGISTER COMERCII

in

BUSINESS MANAGEMENT

in the

FACULTY OF MANAGEMENT

at the

UNIVERSITY OF JOHANNESBURG

JOHANNESBURG **SUPERVISOR: PROF. N. LESSING**
2005

ACKNOWLEDGEMENTS

After all is said and done, an undertaking of this nature is impossible without the help of many people. Discussions with various industry professionals were also invaluable in shaping this document. In particular, however, the following people deserve my sincere gratitude:

- Prof. Nic Lessing from the University of Johannesburg who was always willing to listen and provide guidance as well as motivation.
- Me. Rika Venter, secretary at the M.Com Business Management department at the University of Johannesburg, for her friendly and dynamic personality and her willingness to help whenever she could.
- My wife for her understanding and not allowing me to have a pity party when the pressure is on.
- My dad, for encouraging me regularly, and to ensure that I maintain focus and not spend all my time procrastinating.
- My mother, for her active interest in my wellbeing.
- My brother for ensuring that I register for the M.Com degree four years ago even though I had many excuses.
- And last, but definitely not least, to the Lord for so many blessings.

SYNOPSIS

NAME:	Frits Kok
DEGREE:	M Com (Business Management)
UNIVERSITY:	University of Johannesburg
TITLE:	Open source as a value alternative to commercial software
SUPERVISOR:	Professor N. Lessing
DATE:	31 October 2005

The study aims to provide IT decision-makers with information on the issues to consider when looking at open source from a value perspective when compared to commercial software.

Software value-add should be considered through all the phases in the software life-cycle. The concept of the value chain should be the starting point to identify those aspects of the value chain that would benefit the most. Other initial considerations, such as risks involved in open source, should also be taken into account before deciding on the software to acquire.

The concept of total cost of ownership can be applied to the software employed within an organisation – in other words, considering the costs involved in software from initial purchase through to support and maintenance. In this regard, both open source and commercial software offer different levels of cost in separate areas of the life-cycle. An informed decision regarding the complete software life-cycle enables the decision-maker to justify and substantiate the decision to go the open source or commercial software route - based on the value gained from the software through this life-cycle.

The empirical research aims to highlight those issues of the software life-cycle that are considered most important with regard to open source and commercial software as well as to identify the current perceptions regarding open source in the market today from different points of view.

SINOPSIS

NAAM:	Frits Kok
GRAAD:	M Com (Ondernemingsbestuur)
UNIVERSITEIT:	Universiteit van Johannesburg
TITEL:	Vrye bronkode sagteware (“Open source software”) as ‘n waarde alternatief teenoor kommersiële sagteware.
STUDIELEIER:	Professor N. Lessing
DATUM:	31 Oktober 2005

Die studie poog om IT besluitnemers van inligting te verskaf rondom tersaaklike kwessies om in ag te neem wanneer daar gekyk word na vrye bronkode sagteware (“Open source software”) as ‘n alternatief tot kommersiële sagteware vanuit ‘n waarde-oogpunt.

Waarde-toevoeging deur die sagteware moet in ag geneem word deur al die fases in die sagteware lewens-siklus te evalueer. Die waarde-ketting (“value chain”) behoort die beginpunt te wees met die oog op identifisering van aspekte wat die meeste baat sal vind by die gebruik van ‘n sagteware produk. Ander kwessies, soos die risikos met betrekking tot die gebruik van sagteware, moet ook in ag geneem word voor ‘n besluit geneem word om spesifieke sagteware aan te skaf.

Die konsep van totale koste van eienaarskap (“total cost of ownership”) kan ook toegepas word op die sagteware wat in ‘n organisasie gebruik word – met ander woorde, van die aankoop prys regdeur tot by die koste van ondersteuning en instandhouding moet in ag geneem word. Ter saaklike inligting aangaande die totale sagteware lewens-siklus stel die besluitnemer in staat om ‘n keuse met betrekking tot vrye bronkode sagteware of kommersiële sagteware te regverdig en te substansiëer. Die besluit word gebaseer op die waarde wat toegevoeg word in al die komponente van die lewens-siklus.

Met behulp van empiriese navorsing word gepoog om die komponente van die sagteware lewens-siklus uit te lig wat as van die grootste belang geag kan word met betrekking tot die keuse tussen vrye bronkode sagteware of kommersiële sagteware. Tans geldende sienings ten opsigte van vrye bronkode sagteware word ook vanuit verskillende oogpunte bespreek.

DIVISION OF CHAPTERS

		Page
List of figures		ix
Glossary		xi
List of acronyms		xiv
CHAPTER 1	Orientation	1-1
CHAPTER 2	Software value using the software life cycle	2-15
CHAPTER 3	Total cost of ownership	3-33
CHAPTER 4	Research methodology and research findings	4-57
CHAPTER 5	Interpretation and recommendations	5-78
APPENDIX A	Online survey as made available to respondents	90
BIBLIOGRAPHY		99

CHAPTER 1 ORIENTATION

		Page
1.1	Background of the study	1-2
1.2	Statement of the research problem	1-5
1.3	Study objectives and components of study	1-7
1.4	Research methodology	1-9
1.5	Scope and limitations of study	1-11
1.6	Contribution	1-13

CHAPTER 2 SOFTWARE VALUE USING THE SOFTWARE LIFE CYCLE

		Page
SYNOPSIS		2-16
2.1	Introduction	2-17
2.2	Towards software value: the customer activity cycle	2-18
2.3	The value chain	2-21
2.4	The value delivery network	2-24
2.5	Risks	2-26
2.6	Closure	2-31

CHAPTER 3 TOTAL COST OF OWNERSHIP

		Page
SYNOPSIS		3-34
3.1	Introduction	3-35
2.2	The purchase price	3-37
2.3	The cost of implementation	3-41
2.5	The cost of migration	3-45
2.6	The cost of support and maintenance	3-48
2.7	The cost of training	3-53
2.8	Closure	3-56

CHAPTER 4 RESEARCH METHODOLOGY AND RESEARCH FINDINGS

		Page
SYNOPSIS		4-58
4.1	Introduction	4-59
4.2	Research methodology	4-59
4.3	Research findings	4-65
4.4	Closure	4-76

CHAPTER 5 INTERPRETATION AND RECOMMENDATIONS

		Page
SYNOPSIS		5-79
5.1	Introduction	5-80
5.2	Interpretation of study	5-81
5.3	Recommendations	5-84
5.4	Closure	5-88

LIST OF FIGURES

		Page
FIGURE 1.1	Apple II computer and IBM PC	1-3
FIGURE 2.1	Gartner hype cycle for open source	2-17
FIGURE 2.2	The consumer activity cycle	2-18
FIGURE 2.3	IBM consumer activity cycle	2-19
FIGURE 2.4	Value add for consumer activity cycle	2-20
FIGURE 2.5	Layout of value assessment	2-21
FIGURE 2.6	The value chain (adapted)	2-22
FIGURE 2.7	Perceived weaknesses of Linux	2-27
FIGURE 2.8	Corporate concerns over legal indemnification	2-30
FIGURE 3.1	Different perspectives on total cost of ownership	3-35
FIGURE 3.2	Open source products licensing type usage	3-40
FIGURE 3.3	Spiral development model including user input	3-43
FIGURE 3.4	The free enterprise stack	3-47
FIGURE 3.5	Challenges faced with open source	3-49
FIGURE 3.6	Support costs for medium businesses	3-49
FIGURE 3.7	Commercial Linux support purchases	3-50
FIGURE 3.8	SourceForge development statuses	3-52
FIGURE 4.1	Organisational roles with corresponding technical roles of respondents	4-66
FIGURE 4.2	Environment in which value is contributed by open source	4-68

FIGURE 4.3	Interoperability: Open source and commercial software	4-69
FIGURE 4.4	Perceived risks when using open source	4-70
FIGURE 4.5	Ratings assigned to aspects of total cost of ownership	4-72
FIGURE 4.6	Open source pros and cons	4-73
FIGURE 4.7	Current product usage	4-75

GLOSSARY

An explanation of the Key Concepts is placed in the beginning of this document to serve a dual purpose:

1. Firstly, some of these terms might be new and unfamiliar to the reader and will act as a hindrance in approaching this document.
2. Secondly, these terms can take on different meanings in different contexts, and the meanings of these terms within the context of this document are therefore elaborated on at this point. The terms are explained as they are understood, interpreted and applied by the author, and might differ from other explanations of these terms.

Business software	This includes various software applications that form the administrative/financial/commercial etc. back-office of an organisation and can run either on the desktop or on backend servers.
Commercial software	Also known as proprietary software. Refers to software applications which are made available for purchase by a software vendor or software development house. Examples of commercial software include: Microsoft Windows, Solaris, SCO Unix, SAP, Microsoft Excel and Oracle, with Microsoft Windows being the most apparent competitor to Linux.
Customer activity cycle	Addresses three phases – *pre, during* and *post* in which consumer activities can be mapped towards a specific task. Value adding activities can then be mapped to each of these phases in order to add value to the consumer activity. The activity cycle can also be used in order to address value add from a decision-making

	perspective throughout the complete life-cycle of an activity.
Desktop software/ Client software	Refers to software that normally runs on a desktop machine in a user's office/home environment.
Infrastructure software/Server software	Refers to software that normally runs within an organisation's back-end environment. Examples of software in this category include: mail servers, firewalls, database servers, web servers and server operating systems.
Open source	Refers to software applications which are freely available for use. These software applications also make their source code freely available and modifications to the source code are allowed under certain licensing conditions. Examples of open source software include: Linux, MySQL, PostgreSQL and Apache.
Software development life-cycle	The stages through which a software product moves during software development. In other words, all the activities involved in the creation of a software product.
Software life-cycle	Stages of software usage from initial purchase, to training and maintenance. The software life-cycle comprises of all the activities that form part of the software usage cycle.
Source code	Refers to the computer language instructions in which a software product has been developed. These computer-based instructions for larger applications can extend into millions of lines of functions and programming logic.
Total cost of	Total cost of ownership (TCO) for software can be

ownership (TCO)	defined as "the total price in money, time, and resources of owning and using software." [Netc.org 2005:Internet]

LIST OF ACRONYMS

BSD	Berkeley Software Distribution. An open source licensing type indicating that the source code can be modified and imposes no restrictions on licensees making use of the source code.
GNU	A recursive acronym standing for "GNU's not UNIX". An ongoing project of the Free Software Foundation to create a complete, freely available computing environment.
GPL	General Public License. An open source licensing type indicating that the source code can be modified, but imposes more restrictions on licensees such as requiring that any changes to the source code are made available as well. Various GPL-type licenses exist and each type of license requires licensees to make use of the same type of license for derivative software products.
TCO	Total cost of ownership. See explanation in Glossary.

CHAPTER 1

1 ORIENTATION

Page

1.1 BACKGROUND OF THE STUDY 1-2

1.2 STATEMENT OF THE RESEARCH PROBLEM 1-5

1.3 STUDY OBJECTIVES AND COMPONENTS OF STUDY 1-7

1.4 RESEARCH METHODOLOGY 1-9
1.4.1 Literature study 1-9
1.4.2 Internet 1-10
1.4.3 Empirical research 1-10

1.5 SCOPE AND LIMITATIONS OF STUDY 1-11
1.5.1 Scope 1-11
1.5.2 Limitations 1-12

1.6 CONTRIBUTION 1-13

1.1 BACKGROUND OF THE STUDY

In the 1950s and 1960s the first large scale computers, such as those supplied by IBM [Gonzales-Barahona 2000:Internet], were bundled with software that included the source code and could be modified as such. The hardware itself was sold for a hefty fee, and the software was almost seen as an extension of the hardware – a requirement to "make the hardware work". The early mainframe computers, such as the System/360 required an initial investment of $5 billion and only large corporations and government institutions could afford to purchase or hire system time on these systems.

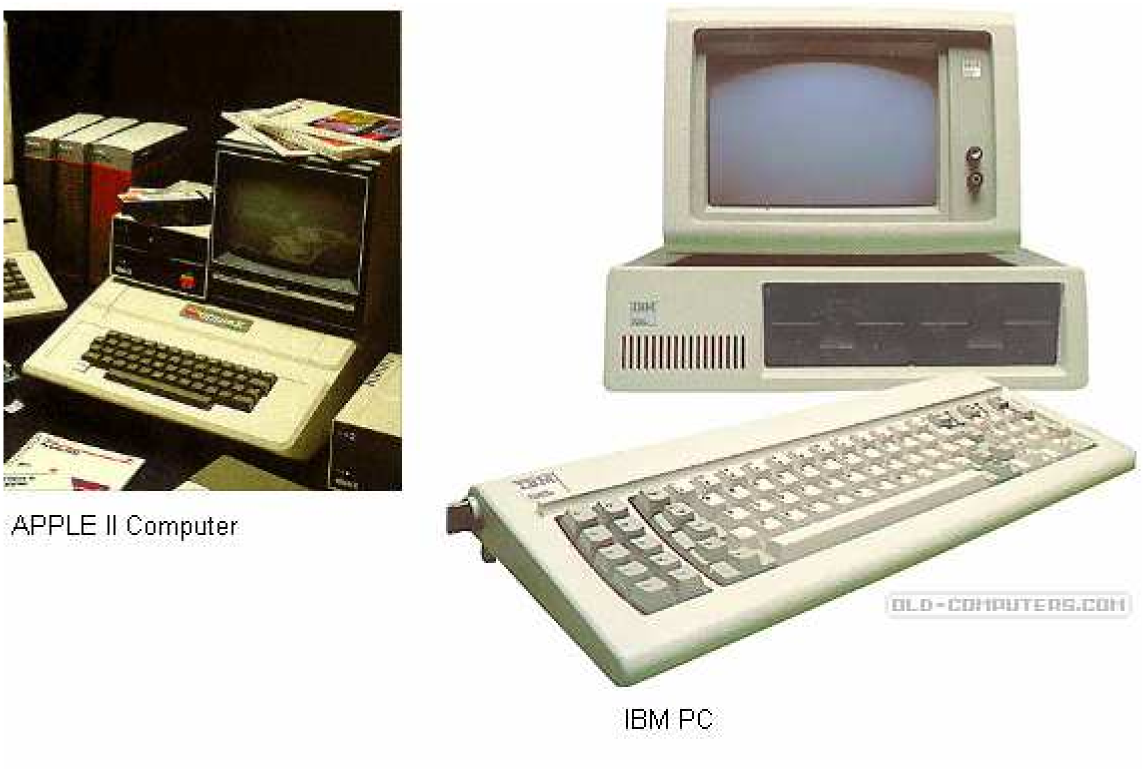
In 1957, FORTRAN (Short for FORmula TRANslator) was introduced as a programming language capable of specifying repetitive tasks via loops instead of specifying the same tasks several times. Two years later, a team drawn from several computer manufacturers as well as the Pentagon introduced COBOL (Common Business Oriented Language) that aspired to improve computer program readability and be machine independent. However, mainframes were still extremely expensive to obtain and time-sharing became commonplace as programmers shared computing time on these costly platforms. Another programming language, BASIC (Beginners All Purpose Symbolic Instruction Code), developed by Thomas Kurtz and John Kemeny, was introduced in 1964 to make use of the Dartmouth Time Sharing System at Dartmouth College specifically for the purpose of time sharing in order to efficiently make use of the expensive mainframe platforms of the day.

As computer programming languages became more common and user-friendly, computers became smaller and less expensive. In 1975 the first Apple computer, the Apple I, designed by Steve Wozniak, was made available for $500 - an order of a magnitude less expensive than the earlier mainframe systems. In 1977, the

Apple II became an instant success which, when hooked up to a television, could produce dazzling colour graphics. In 1978, IBM introduced its Personal Computer (PC) and jump started the growth in the personal computer industry [Computer History 2004:Internet].

Figure 1.1 indicates some of the early personal computer hardware of the 1970's.

Figure 1.1 Apple II Computer and IBM PC



Source: Computer History Museum 2004:Internet
Old Computers 2005:Internet

As computers became more affordable, they also became more standardised. The increase in software development tools and programming languages also allowed more companies to provide software for these systems. Two of the most common software applications were MS DOS (Microsoft Disk Operating System) for the IBM PC and Visi-Calc [Bricklin 1999:Internet] – a spreadsheet program first introduced for the Apple and later for the IBM PC and a precursor to Lotus 123. These software applications were developed by skilled application programmers and were sold for commercial gain. As such, a user requiring the functionality of these commercial applications was required to purchase a license in order to make use of it.

However, through the late 1970s and early 1980s two separate groups at MIT and at the University of California at Berkeley were also establishing the roots of the open source movement [Gonzales-Barahona 2000:Internet]. Richard Stallman resigned at MIT to launch the GNU (a recursive acronym stating "GNU is NOT UNIX") and the Free Software Foundation. The aim of the GNU project was to create a free operating system and associated utilities.

The Computer Science Research Group (CSRG) at Berkeley was developing UNIX enhancements and utilities. In the late 1980s the operating system and utilities that started out being developed by the CSRG and later included the assistance of various UNIX "hackers", was released under the BSD license – one of the first open source licenses.

Into the 1990s the wide adoption of the Internet opened the doors for various geographically dispersed collaboration efforts – one of them being in the software industry. Up to this point, most enterprise-wide or commonly used software applications were developed and distributed by commercial software houses aiming to make a profit out of their development efforts and expertise in a specific software application environment. Although open source software had existed during this time for as long as commercial software, the broad adoption of the global network that the Internet offered proved to be fertile breeding ground for open source software development.

The first truly global success of the capability of open source came with the release of Linux [Linux.org 2005:Internet] – an operating system similar to Unix. Contrary to the commercial operating systems such as UNIX (Sun, SCO) or Windows (Microsoft), Linux was completely free. Furthermore, the Linux source code was also freely available and many individuals from various countries were invited to contribute to further its progress. Although previous open source operating systems and applications existed, the Linux development process and its contributions from various countries caught the "Internet wave" and made use of the global communications network to produce a large scale commercially competitive operating system and suite of utilities.

In many respects, into recent times, open source has touted alternatives to the traditional commercial offerings. This has suitably upset commercial software houses as a potential source of competition and even cannibalisation of the software industry. Furthermore, companies around the globe are faced with increasingly difficult decisions with regard to software application choices between open source and commercial offerings and the value proposition that each brings to the table.

1.2 STATEMENT OF THE RESEARCH PROBLEM

Governments and businesses around the globe are faced with progressively more difficult decisions when it comes to software selection and implementations. Open source and commercial software are becoming increasingly interwoven and the uptake of open source software in the public sector is on the rise [Postnote 2005:Internet].

The initial free price tag of open source is enticing to all those that need to adhere to an Information Technology budget or those organisations with very limited financial resources but with a definite information technology requirement. In fact, most technology decision-makers will be hard pressed not to be interested in the "free" alternative that open source offers.

Therefore, as applications in the open source industry mature, decision-makers need to be made aware of the issues to take into consideration when deciding on which software to employ for a specific purpose within an organisation. More specifically, which activities and aspects contribute to value with regard to software systems.

The value proposition that a software vendor offers can be defined as "an implicit promise a company makes to its customers to deliver a particular combination of values" [McBryde 2004:Internet]. This research will consider the software value proposition using the customer life-cycle perspective, which is divided into three phases – *pre, during and post*. In the *pre* phase, initial considerations - in other words issues to consider before the software purchase is made will be addressed, such as value chain and risk considerations for open source and commercial software. In the *during* and *post* phases the issue of total cost of ownership will be addressed. Total cost of ownership (TCO) for software can be defined as "the total price in money, time, and resources of owning and using software. TCO is widely considered to include the purchase price, and the cost of hardware and software upgrades, maintenance and technical support, and training (or re-training). TCO is an essential part of technology decision-making since a product may be cheap or free to get, but involve excessive maintenance or training" [Netc.org 2005:Internet].

Problem statement: Technology decision-makers are increasingly faced with the dilemma of making an informed choice between open source and commercial software. The issues of value to consider are not always clear and not all stages in the software life-cycle are evaluated.

1.3 STUDY OBJECTIVES AND COMPONENTS OF STUDY

The mission of this study is:

To elaborate on the issues to deliberate on when deciding on a software solution. This includes a discussion of the value, risk and total cost of ownership aspects to take into consideration from a commercial and open source software perspective through the entire software life-cycle.

Since new software products are continuously entering the market and open source are playing a more prominent role in the software industry, it is essential for decision-makers to recognise the many issues that need to be considered in order to make an informed decision.

In this respect open source is considered as an alternative to commercial software with regard to the *value* it brings to the table. These issues to consider already begin before a software acquisition is made and start by evaluating the software's impact within the organisational value chain and risk exposure. Ultimately a set of value aspects are identified throughout the software life-cycle. These value aspects can then be considered in the decision to either use open source or commercial software in a given situation.

The objectives of the study are as follows:

Objective 1: To provide an overview of the value concept when evaluating open source software from a customer activity cycle perspective and to discuss the issues to consider before making a software acquisition decision.

This is necessary in order to understand the life cycle considerations that should ultimately affect the decision to make use of open source or commercial software in a given situation.

Chapter 2 provides an overview of the current open source environment and explains the three phases, which form part of the life cycle in which software value contributions are made, namely *pre, during* and *post* by using the customer activity cycle. The first phase (*pre*) is then elaborated upon in order to explain pre-purchase value aspects to consider, namely the value chain, the value delivery network and risk aspects.

Objective 2: To discuss the value adding issues to consider by using the concept of total cost of ownership (TCO).

Decision-makers need to look past the initial purchase price with regards to commercial software and the free price tag of open source software to evaluate the total running costs of a specific software solution.

Chapter 3 addresses the *during* and *post* phases, as identified in **Chapter 2**, by discussing the value aspects in these phases using total cost of ownership to identify software life-cycle activities that can either contribute or detract from value, namely the purchase price, the cost of implementation, the cost of migration, the cost of support and maintenance and finally by looking at the cost of training.

Objective 3: To provide information on the methodology employed in the empirical research.

The methodology of the empirical research is elaborated on in **Chapter 4**. **Chapter 4** will also discuss the research findings, namely:

1. The characteristics of respondents.
2. Perceived value, interoperability and risks.
3. Software life cycle considerations.
4. Pros and cons with regards to open source.
5. Current product usage of respondents.

Objective 4: To provide some recommendations to the steps an organisation should follow when evaluating open source alternatives.

Open source is here to stay. The question is rather how an organisation can make a fact-based, informed decision on which product to use from the open source stable or from a commercial vendor. **Chapter 5** aims to provide decision-makers with guidelines in the decision-making process. The software life cycle, total cost of ownership and research findings will be discussed with the aim to provide recommendations to the industry as well as recommendations for further research.

1.4 RESEARCH METHODOLOGY

A literature review is included in the research with the Internet forming the main source of information. Empirical research is employed to test the current perceptions on open source from different roles within an organisation.

1.4.1 Literature study

Although open source has been around since the late 1970s, the amount of literature do not abound. However, certain principles from the field of management can also be applied to the software arena. These principles are widely recognised and formal types of literature, such as textbooks, are used.

Government publications are also becoming more prevalent with regard to open source and are also employed in this study. Sources have been selected based on their content with regard to issues involving open source or commercial software and their relevance to the topic of this study. In this regard the literature review is a central component to the study.

1.4.2 Internet

Open source's proliferation can in part be attributed to the wide adoption and tremendous growth of the Internet. It therefore serves as a very valuable resource in the latest developments and current approaches to open source within an organisation. As the Internet also acts as the main distribution channel of the latest open source software as well as providing frameworks for collaborative development, it forms the main source of information. Although the Internet can be questionable with regard to the validity of information, the author has attempted to use information from recognised sources such as government publications and well known research organisations.

1.4.3 Empirical research

The empirical research aims to identify current perceptions of individuals in an organisational context regarding open source and commercial software using value aspects in the software life-cycle as identified in the literature review.

1.4.3.1 Questionnaire

The sample of respondents consists of various types of software users, including end-users, implementers and developers/programmers (technical role). Furthermore, the respondents have different roles within the organisation (organisational role) and their perceptions or current attitudes towards open source and commercial software can also be linked to these roles. The survey process made use of an online survey provider and the questionnaires were all completed anonymously and securely online.

The aim of the questionnaire is:

To identify the current perceptions of individuals in an organizational context surrounding open source and commercial software issues as identified in the literature review based on different organisational roles and level of software usage.

Open source are becoming more mainstream as various organisations and even governments are considering it for large projects. A decision-maker needs to understand how to identify aspects of value within this environment in order to keep up to date with the latest developments and options available for governments and businesses alike.

1.5.1 Scope

This study aims to outline the considerations for the applicability of open source in a given environment. In this regard, it endeavours to provide certain generic value aspects throughout the software life-cycle, instead of focusing only on the cost of purchase. Where possible, the study highlights aspects of open source and commercial software as a whole, but in some cases will specifically mention differences between two of the major alternatives in the industry – Linux (open source software) and Microsoft (commercial software).

1.5.2 Limitations

With a study of this nature limitations are to be expected. The main limitations of the study are that it is of a general nature overview and some information provided in this study is based on confirmed industry opinion, rather than hard statistics.

1.5.2.1 General overview

The sheer breadth of aspects to consider makes it difficult to evaluate all the issues involved or to get all the pertinent information with regard to current developments within the open source and commercial software environment. Although the aspects of value is of a constant nature and can even be applied to developments that have not yet taken place, quantifiable measures for these aspects are not provided.

The *implication* of this limitation is that although the value aspects defined within this document are of a constant nature and therefore applicable in any software decision, the lack of quantifiable measures for some value aspects make it difficult to compare alternatives on a purely measurable basis.

Although this limitation in and of itself is not overcome in this document, it by no means diminishes the collection of value aspects as identified in **Chapter 2** and **Chapter 3**, and can be used in fact-based decisions using these value aspects.

Quantifiable measurements for the value aspects as identified in this document can also form the basis for future studies in this regard.

1.5.2.2 Not enough statistics

Because of the pace of change in open source industry, and the IT industry for that matter, statistics can quickly become outdated and not applicable and reliable anymore. Another outflow of this is that opinions are quickly formed because of the rate at which information travels and which results in some statistics being based on opinion rather than evaluated fact.

Another outflow is that because this study is of a pioneering nature, statistics regarding the value aspects as identified in **Chapter 2** and **Chapter 3** are not readily available.

To counteract the first limitation of opinion-based statistics, statistics have been gathered from reputable research institutes with credible track records in this regard, in an effort to provide trustworthy information with regard to open source and commercial software value aspects.

To overcome the second limitation of limited availability of statistics for the value aspects as identified in **Chapter 2** and **Chapter 3**, the research as laid out in **Chapter 4** aims to contribute statistics in this regard for scientific use.

1.6 CONTRIBUTION

The aim of this research is to contribute in a two-fold manner:

Firstly to add to the existing body of knowledge as it pertains to software selection, specifically with regard to value-based decision-making and identifying value aspects to consider in the three phases of software selection and application as identified by the customer activity cycle.

Secondly to aid industry players in addressing the relatively new challenge of alternative selections between open source and commercial software by looking past the "hype" and making informed, fact-based decisions by applying the value aspects as identified in **Chapter 2** and **Chapter 3**.

CHAPTER 2

Value: An amount, as of goods, services, or money, considered to be a fair and suitable equivalent for something else; a fair price or return [The Free Online Dictionary 2005:Internet]

2 SOFTWARE VALUE USING THE SOFTWARE LIFE CYCLE

		Page
SYNOPSIS		2-16
2.1	INTRODUCTION	2-17
2.2	TOWARDS SOFTWARE VALUE: THE CUSTOMER ACTIVITY CYCLE	2-18
2.3	THE VALUE CHAIN	2-21
2.4	THE VALUE DELIVERY NETWORK	2-24
2.5	RISKS	2-26
2.5.1	Open Source Risks	2-26
2.5.2	Commercial Software Risks	2-28
2.5.3	Indemnification	2-29
2.6	CLOSURE	2-31

SYNOPSIS

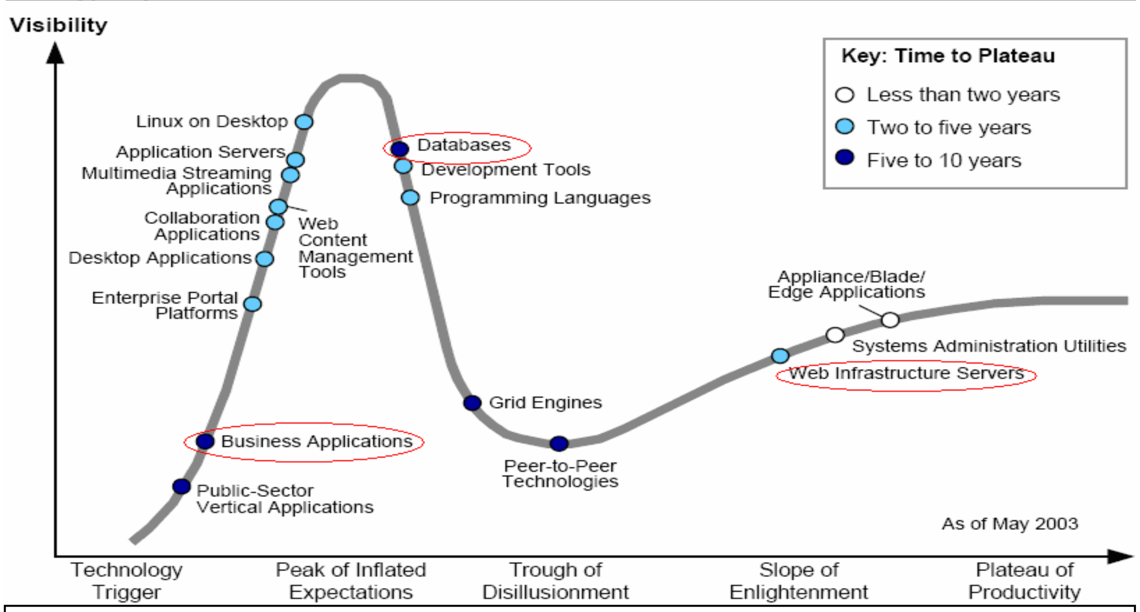
Chapter 2 starts by discussing the approach to take when evaluating open source software from a value perspective, in other words, to look past the "excitement" in order to make an objective, fact-based decision. To identify value aspects, the customer activity cycle is used. This cycle defines three phases of activity – *pre, during* and *post*, with value contributions possible at each phase. The *pre* phase is then elaborated upon in this chapter – a phase where the decision on "what" is made. In other words, what product is most necessary for an organisation based on its current requirements in the value chain, the value delivery network as well as minimising the type of risks involved.

2.1 INTRODUCTION

AS organisations are faced with increasing publicity surrounding open source, decision-makers today feel compelled to include open source software within their organisation, sometimes simply because it is the "in-thing" to do. Companies are bombarded by buzzwords yet many times lack unbiased, fact-based information to direct their choices.

Gartner [2003:Internet] coined the term "hype-cycle" to describe the uptake of new technology. In this regard, Gartner describes these technologies as climbing a steep "hype" curve as pundits describe the benefits of the technology. After the initial uptake, users are disillusioned with the technology and it falls into a "trough of disillusionment". In the final phase, users are in a better position to understand the technology and benefit from it as stable offerings emerge. Gartner also created a "hype-cycle" for open source technologies with their predicted technology applications through this cycle, including the time frames for each type of technology application, as illustrated in Figure 2.1:

Figure 2.1 Gartner hype cycle for open source


Visibility
Key: Time to Plateau
Less than two years
Two to five years
Five to 10 years
Linux on Desktop
Application Servers
Multimedia Streaming Applications
Collaboration Applications
Desktop Applications
Enterprise Portal Platforms
Web Content Management Tools
Databases
Development Tools
Programming Languages
Business Applications
Public-Sector Vertical Applications
Grid Engines
Peer-to-Peer Technologies
Appliance/Blade/ Edge Applications
Systems Administration Utilities
Web Infrastructure Servers
As of May 2003
Technology Trigger
Peak of Inflated Expectations
Trough of Disillusionment
Slope of Enlightenment
Plateau of Productivity

Source: Gartner 2003:Internet

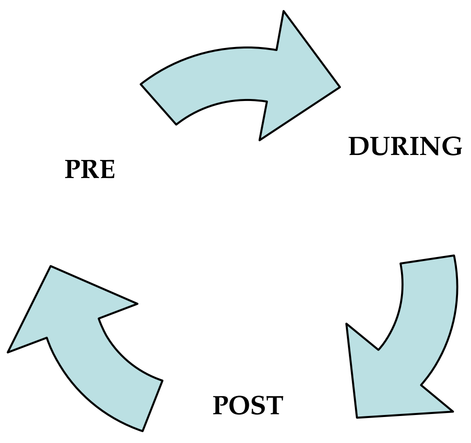
As indicated in red in figure 2.1, open source business applications were only approaching the hype-cycle in 2003, with open source database applications approaching the "trough of disillusionment" and web infrastructure servers reaching the "plateau of enlightenment" and ready for mainstream use.

In essence, these open source technologies offer new alternatives to the IT decision-maker, and should simply be considered as another option in the technology arsenal. **Chapter 2** starts by discussing value aspects to consider before the decision to purchase is made.

2.2 TOWARDS SOFTWARE VALUE: THE CUSTOMER ACTIVITY CYCLE

Vandermerwe (1999:94) discusses a way of mapping consumer behaviour toward a particular task. The mechanism for mapping consists of three phases as shown in figure 2.2:

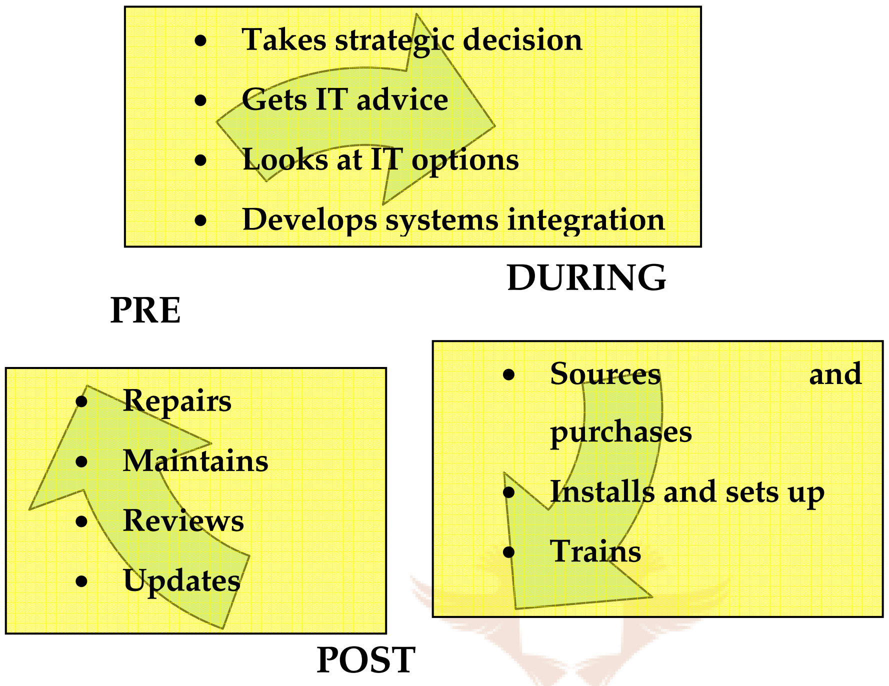
Figure 2.2 The consumer activity cycle



Source: Vandermerwe 1999:94

Each of these phases contains certain activities which a consumer follows for a specific task. The challenge for an organisation is to add **value** at each stage in this cycle. This cycle can also be used to map out the three phases in which value is extracted from a task, activity or entity. The activity cycle for IBM customers in the global electronic networking market space can be depicted in figure 2.3:

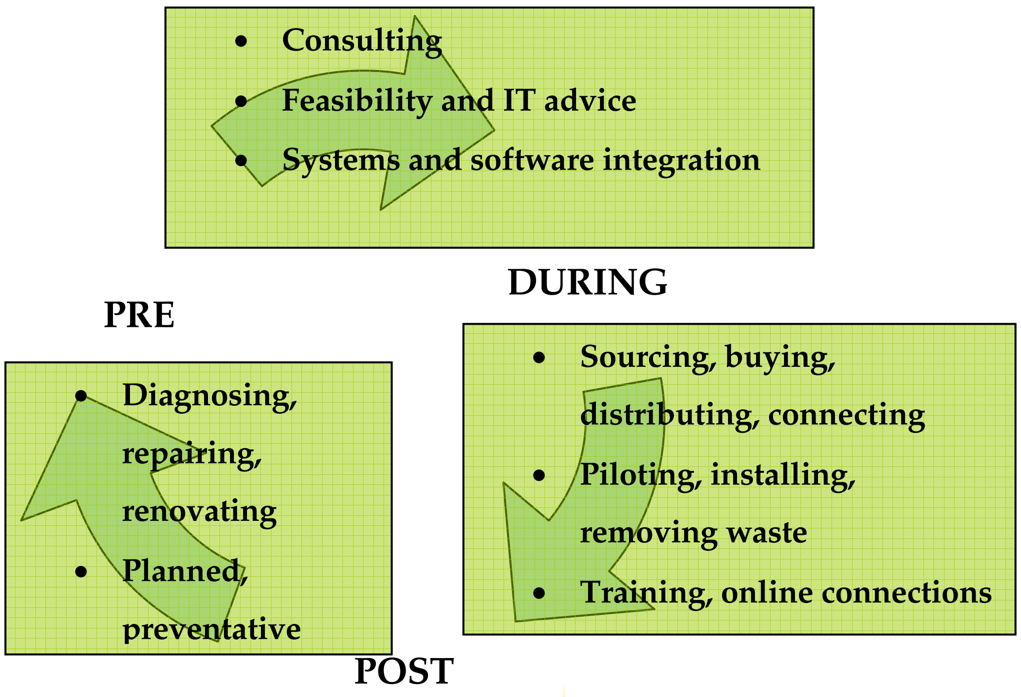
Figure 2.3 IBM consumer activity cycle



Source: Vandermerwe 1999:92

IBM took the tasks that are normally associated with the networking market space and devised activities which will add value each step of the way. Figure 2.4 illustrates all the value-adding activities that IBM partakes in for this consumer activity cycle:

Figure 2.4 Value add for consumer activity cycle

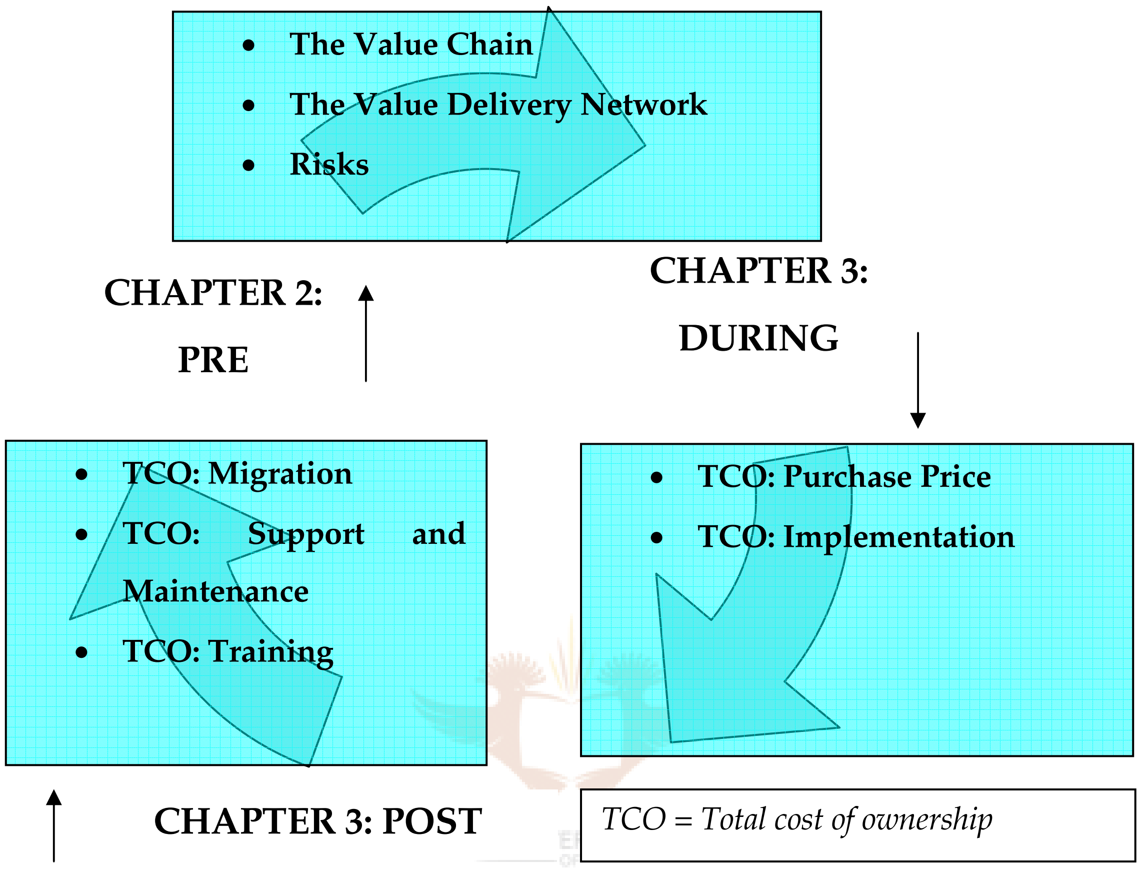

Consulting
Feasibility and IT advice
Systems and software integration
DURING
PRE
Diagnosing, repairing, renovating
Planned, preventative
Sourcing, buying, distributing, connecting
Piloting, installing, removing waste
Training, online connections
POST

Source: Vandermerwe 1999:94

In this way IBM attempts to provide the consumer with value at each stage in the process. The more value adding activities, the higher the probability of creating "customer lock-in" where a customer is essentially forced to make use of a specific supplier simply because the value gained is superior to any alternative.

This document proceeds to use the "Pre, During and Post" model to discuss the issues that should be investigated when considering a value proposition for a software application in order to provide a guide for software selection that maximises value. The structure followed in this chapter (**Chapter 2**) and the next chapter (**Chapter 3**) are indicated in figure 2.5:

Figure 2.5 Layout of value assessment



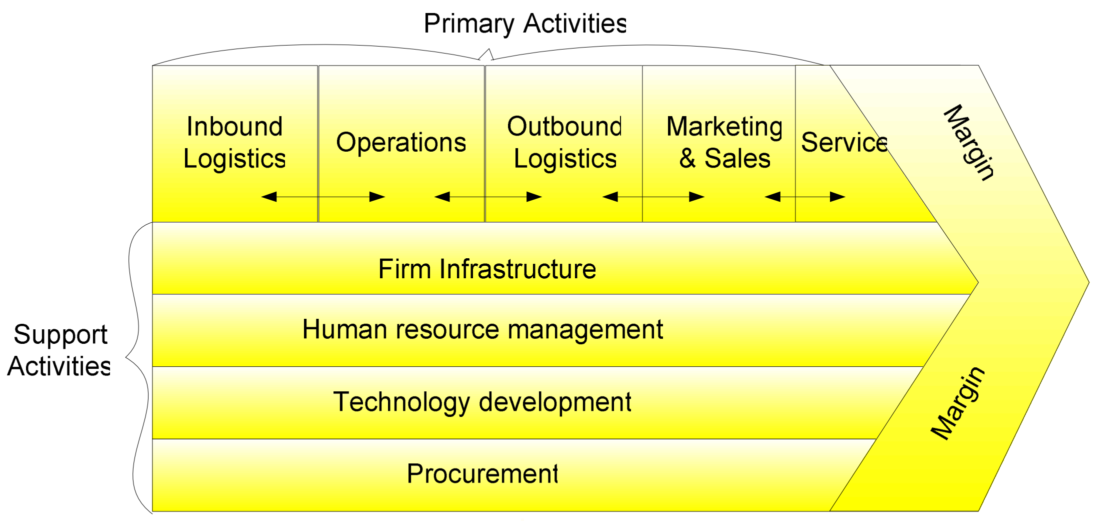
Source: Chapter layout

This chapter will focus on the *pre* phase of software selection and will start by discussing the concept of the **value chain** (Porter 1985:11-15). This is an important value aspect to consider when evaluating the type of software that will offer the best value contribution with regard to activities within the organisational value chain – essentially deciding on "what" software application is best suited to an organisation at a specific moment in time.

2.3 THE VALUE CHAIN

Figure 2.6 illustrates an adaptation of the value chain as provided by Porter (1985:11-15).

Figure 2.6 The value chain (adapted)



Source: Adapted from Porter 1985:11-15

Porter (Porter 1985:11-15) identified five primary and four secondary value-creating activities that is part of every firm:

1. Primary activities: These activities represent the sequence of bringing materials/services, converting them into final products or services, shipping or delivering them, marketing them and finally, supporting them. In essence, these activities are the primary value adding activities of a firm.
2. Support Activities: These activities support the primary activities and in so doing are contributing to the value-add of a firm and consist of activities supporting the firm infrastructure, activities within human resource management and technology development and research in order to improve efficiency and/or effectiveness of the organisation.

The value-chain is important in software selection and in identifying software value for two reasons:

1. **Efficiency** - As fig. 2.6 indicates, the primary activities logically follow on each other. Each activity may rely on a seamless bidirectional flow of information to and from the follow-on activity in order to provide maximum value-add. These activities are increasingly managed by sophisticated software systems. Examples could include:
    a. The "Operations" activity being able to inform the "Inbound Logistics" activity of scarcity of certain materials and a requirement for replenishment, such as the Just-In-Time delivery system (Coyle, Bardi & Langley 2003:680]
    b. Tracking product sales and being able to relay this information to the "Operations" activity in order to focus on the most profitable items or on items that require stock replenishment at vendor outlets.
    c. Service activities are closely monitored and a one-to-one business-to-consumer relationship can be maintained, for example by using CRM (Customer Relationship Management) systems. [CRM Systems 2005:Internet]
2. **Effectiveness** – The "Technology development" supporting activity entails technical efficiencies, innovations and capabilities which enable an organisation to improve the transformation from inputs into outputs and which results in cost-savings, time-savings, the introduction of additional products/services, the ability to serve the customer better and a multitude of other advantages. This activity does not only serve the firm in "doing things right", but also in understanding what the "right things" are.

Understanding the concept of efficiency and effectiveness enables decision-makers to prioritise and include these aspects in any software acquisition decision.

A few criteria emerge from this discussion for use in the generic value proposition for software selection:

1. Software that assists or supports the primary activities and in this way increase the efficiency and/or effectiveness of the business should receive priority. In other words, software which assists in activities in the value chain that have been earmarked as strategically important or underperforming, should be higher on the priority list.
2. Software selection for a new project should take cognisance of its role as part of the whole within a firm. Therefore, the potential integration of new software or lack thereof within an organisation's existing software environment should be taken into consideration.
3. Software selection that can increase the margins and therefore the profit of a firm should receive priority. The higher the potential margin increase, the higher the selection priority.

These aspects can greatly aid in identifying software which will contribute the most value to an organisation at a specific point in time.

2.4 THE VALUE DELIVERY NETWORK

Firms co-exist with other businesses in the marketplace. In many cases, firms have partnerships with other firms in order to increase their ultimate value proposition for products or services to create a superior value delivery network (Kotler 2003:71).

As each of these firms rely on efficient and effective communication between business partners it is essential that their systems, more specifically their software systems, can communicate with each other. Examples of such communication include:

- Being able to notify supplier partners of replenishment requirements for specific raw materials.
- Electronic documentation handling between business partners such as invoicing and ordering.

Frequently a firm do not exist alone within a marketplace and partnerships are forged to improve competitiveness. Incompatible software systems between partners lead to inefficiencies in the value delivery network. An additional criterion for the generic software value proposition should therefore be added:

- Software that can interact with business partners' existing software systems should receive priority over those that do not integrate well with these systems.

In various software arenas standards have been developed which streamlines inter-application communication and allows software from different vendors to interact. Examples include:

- Electronic Data Interchange (EDI) – an efficient standard of communication between software systems which a firm can use to notify its suppliers of required materials or relay other information. (Coyle, Bardi, Langley 2003:675)
- ebXML [EBXML.ORG 2005:Internet] – a standard that allows enterprises to conduct business using the Internet.

Standards within a specific software arena should be identified and software systems which implement standards and standard interfaces should receive priority over those that implement proprietary interfaces.

As seen from the abovementioned discussion, various software solutions do not exist on their own, but form part of a larger software network supporting organisations and their business partners. The above considerations should form part of a value-based decision with regard to the software system to employ whether it is open source or commercial software.

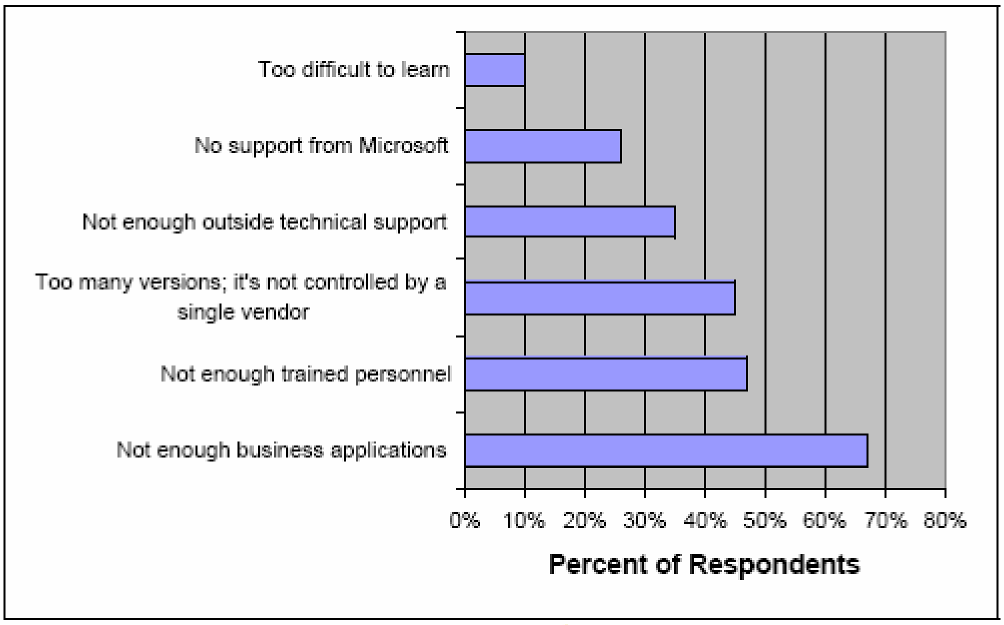
2.5 RISKS

When considering choosing either an open source or a commercial software solution, various risks need to be investigated.

2.5.1 OPEN SOURCE RISKS

In an *Information Week* survey done in January 2000 [Kenwood 2001:Internet], with some relevant risk-related statistics provided in figure 2.7, almost seventy percent of the respondents indicated that their greatest concern surrounding the use of Linux was the lack of business applications. The second highest perceived weakness as identified by respondents was the shortage of trained personnel.

Figure 2.7 Perceived weaknesses of Linux



Source: Kenwood 2001:Internet

If a decision regarding open source is made, these risks need to be mitigated. Options to do so include:

- Outsourcing support to an established Linux support centre that is affiliated with one of the recognised Linux distributions.
- Understanding the requirements surrounding business applications for a specific organisation and identifying potential software solutions in the open source arena which can fit the bill.
- Categorising training requirements and ensuring that the training material is adequate, either by using in-house expertise or handing the training over to a recognised Linux training centre.
- Recruiting individuals with expertise in the required open source applications to ensure that the necessary skills exist within the organisation to effectively utilise and operate the software solution.

The level to which each of these risks can be addressed should be a deciding factor in the move to using open source.

2.5.2 COMMERCIAL SOFTWARE RISKS

The use of commercial software is also not without its risks. Whereas open source provides the source code for free, and therefore to some extent ensures the longevity of a certain product's use within an organisation, the same cannot be said for commercial software. An organisation does not necessarily have any guarantees regarding the long term viability of a commercial software vendor, which ultimately impacts on the business of the organisations using its software. It is important therefore to look at the track record of a software vendor with a view to how it conducts business. Feedback from existing customers of a software vendor is also a useful gauge in deciding whether to go with a vendor or not – positive feedback can at least provide some assurance that a software vendor has a proven track record.

Clapp (2001:5) states that commercial software can take away the manager's control over which features will be improved upon and when new upgrades to the software will be made available. However, this risk is just as prevalent with open source software, or even more so, as there is no monetary motivation from an open source perspective to introduce certain features specifically for one customer. Furthermore, essentially there is no single vendor to contact when it comes to requiring additional features. This said, open source at least offers the knowledgeable organisation the opportunity to modify the source code to suit their needs. In general, however, this is not often done.

Another potential risk is vendor lock-in. If an organisation makes exclusive or extensive use of software that is developed and maintained by only a single vendor, the risk always exist that the organisation becomes over-reliant on the software vendor. This can lead to a situation where a software vendor can make demands in terms of increased pricing, or where the organisation's continued existence is dependent on the software vendor's continued existence.

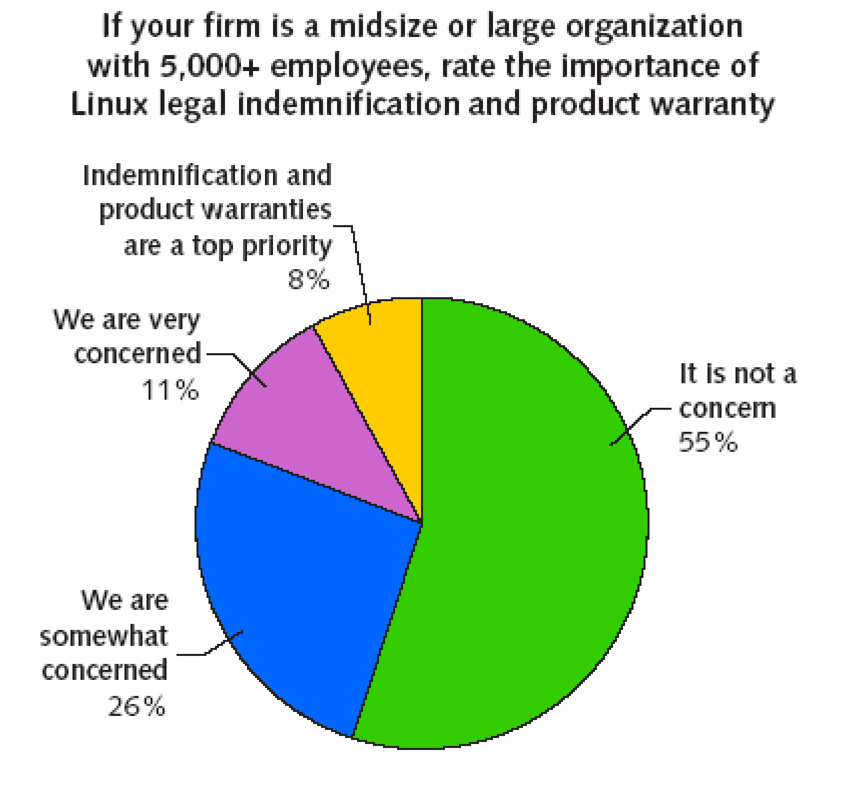
As can be seen from the above discussion, commercial software is also not without its risks and these should be identified and evaluated in order to understand the short and long term value implications they will have.

2.5.3 INDEMNIFICATION

With intellectual property forming the basis of the software industry in order to maintain a competitive advantage, the illegal use of this property can have dire financial consequences for the guilty parties. In recent lawsuits surrounding the misuse of intellectual property (as cited in Graeme & Boariu 2004:2), Microsoft agreed to pay Sun Microsystems $900 million to resolve patent issues, Sun Microsystems agreed to pay Kodak $92 million for a patent infringement and Research in Motion is currently in a lawsuit against NTP Inc. for $53 million for patent issues.

However, until recently, end-users were not involved in the crossfire between different software vendors. On the 6th of March, 2003, SCO filed a lawsuit against IBM listing various misappropriations with regard to intellectual capital. SCO did not stop with IBM and issued letters to Linux customers warning them about the current law suit and that end-users may be liable. On 19 December SCO sent out a cease-and-desist letter, in other words to stop making use of certain open source applications, to Fortune 1000 companies charging them with illegally using SCO intellectual property. This action by SCO has left many end-users of open source, especially larger organisations, with doubts about the potential legal wrangles surrounding intellectual property. The awareness surrounding potential intellectual property have increased, and figure 2.8 highlights the current corporate concerns regarding indemnification on this issue:

Figure 2.8 Corporate concerns over legal indemnification



Source: Didio 2004:1

Indemnification can be defined as "an act of compensation for actual loss or damage or for trouble and annoyance" [WordReference 2005:Internet]. As a result of the increase in intellectual property cases with regard to software, many companies have introduced indemnity protection programmes in the event that a customer is caught in the middle of a legal battle surrounding intellectual property concerns. IBM, one of the larger open source proponents offers no indemnification programme even though it is one of the companies targeted for intellectual capital infringement. Instead, IBM is taking the position that the responsibility of indemnity protection lies with the originator of the Linux distribution (Graeme & Boariu 2004:3). Microsoft and Sun, however are offering very comprehensive indemnification programmes for their end-users (Didio 2004:5).

When considering open source as an alternative therefore, it is important to understand the risks involved. Some of the issues such as the current lack of business applications of the standards of commercial software or the shortage of skilled support should be evaluated against the free (or nearly free) price tag. A new risk that is surfacing is the issue of indemnification, and the importance of an indemnification programme should form part of the risk evaluation.

Commercial software is also not without its risks and issues such as a proven track record and customer lock-in are but a few of the potential issues that must be investigated.

2.6 CLOSURE

The objective of this chapter is to identify the phases of the software life cycle in which value contributions can be made and to discuss the issues that need to be considered initially in order to make a value-based decision. In this regard, the chapter discusses the *pre* phase and examines the value chain, the value delivery network and the risks commonly associated with open source or commercial software.

Using the concept of the *value chain* it is possible to identify potential areas in the value chain in which the introduction of a software system will enhance the efficient and/or effectiveness in that area. Understanding the *value delivery network* enables the identification of interoperability standards that selected software need to adhere to in order to enable streamlined communication between business partners. The last aspect identified to be considered in the "pre" phase is *risks* that can generally be associated with the use of open source and commercial software, such as lack of support and vendor lock-in.

Aimed with these considerations, a decision-maker is well prepared to understand which software offering is most applicable in a given situation.

CHAPTER 3

"Linux is never really going to be a rich sell." – Linus Torvalds [Mundie 2001:Internet]

3 TOTAL COST OF OWNERSHIP

		Page
SYNOPSIS		3-34
3.1	INTRODUCTION	3-35
3.2	THE PURCHASE PRICE	3-37
3.2.1	Open Source Licensing Types	3-38
3.3	THE COST OF IMPLEMENTATION	3-41
3.4	THE COST OF MIGRATION	3-45
3.5	THE COST OF SUPPORT AND MAINTENANCE	3-48
3.5.1	Stability and reliability	3-51
3.6	THE COST OF TRAINING	3-53
3.7	CLOSURE	3-56

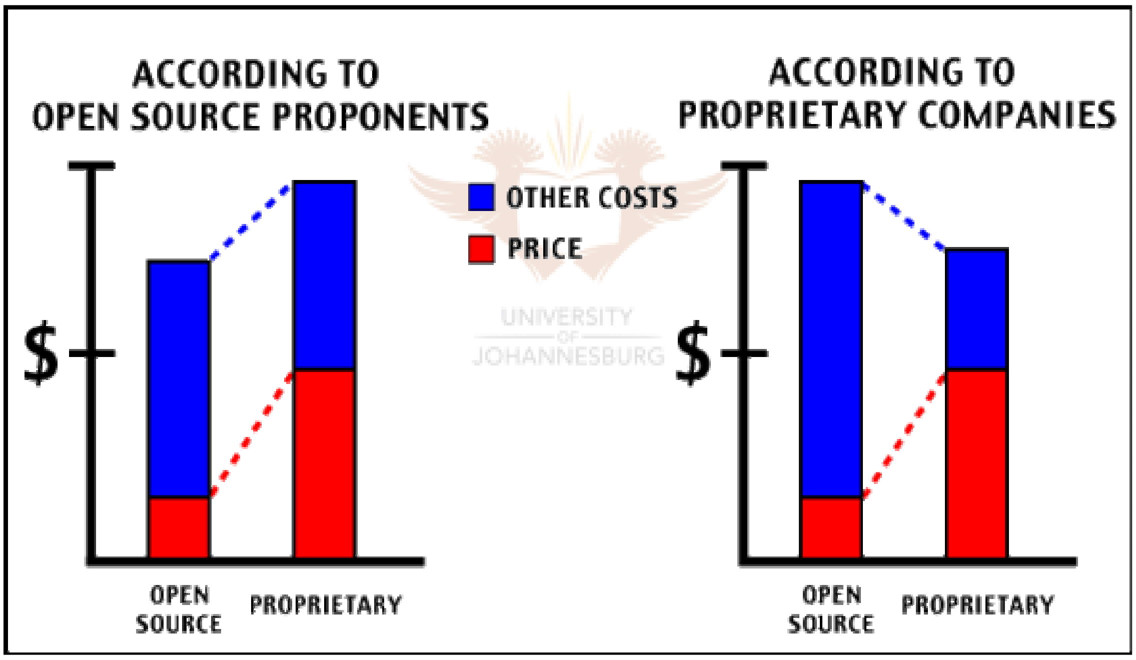
SYNOPSIS

After evaluating the issues to consider in the "pre" phase of the customer activity cycle with regard to software value, it is necessary to consider the "during" and "post" phases. **Chapter 3** looks at the "during" and "post" phases at the hand of Total Cost of Ownership or TCO. Five aspects, from the purchase price to the training on the software is identified and elaborated upon specifically with regard to open source and commercial alternatives. Ultimately, knowledge of all the aspects involved in the use of software within an organisation enables a more thorough investigation into software products in order to make an informed decision between the alternatives.

3.1 INTRODUCTION

In a Forrester survey of 140 North American companies on their use of open source [Giera 2004:Internet], responses indicated that the biggest motivation behind the move to open source was low acquisition costs. The second biggest reason cited was an expected lower total cost of ownership (TCO) than existing commercial software. Open source proponents and commercial software vendors differ in opinion as figure 3.1 indicates:

Figure 3.1 Different perspectives on total cost of ownership



Source: OpenOptions 2005:Internet

However, of the companies that kept detailed metrics regarding their open source software spending, Linux was 5% to 20% more expensive than current Microsoft environments [Giera 2004:Internet], although the companies indicated that they expected these costs to come down as their proficiency in open source technology increase. To understand the costs involved in using software therefore, it is useful to look at the concept of total cost of ownership in more detail.

Total cost of ownership can be defined as "the systematic quantification of all costs generated over the lifetime of a project." [Odellion Research 2005:Internet]. In relation to information technology, Gartner [AVST 2004:Internet] defines it as "the total of [IT] costs over time". Chou [2003:Internet] states that to understand the cost of software one must track all related expenditures through the software life-cycle. The point being that the cost of software does not simply entail the purchase price and that the complete software life-cycle should be considered when evaluating software cost. The first task therefore is to define the activities that make up the software life-cycle in order to calculate total cost of ownership.

AVST [2004:Internet] mentions capital investment, license fees, leasing costs and service fees as potential sources of software costs. They continue to state that additional costs such as direct and indirect, or budgeted and unbudgeted, expenses are also potential cost elements. Coleman [1998:Internet] states that the following six elements should be included in a TCO calculation:

1. The purchase price for all hardware and software
2. The training cost
3. Application change(s) cost to maintain compatibility
4. The cost of maintenance and support
5. Environmental changes required to permit connectivity
6. Technical support contract costs

Many organisations also have existing software installations which will require to be migrated to the new systems. This taken into consideration the following elements will be discussed in more detail with regard to the software life-cycle:

1. The purchase price
2. The cost of implementation
3. The cost of migration
4. The cost of support and maintenance
5. The cost of training

From discussions it will become apparent that these issues are interrelated and that they should be considered as a whole when considering the total cost of ownership of a software solution.

3.2 THE PURCHASE PRICE

Apart from the cost of actual distribution material (CD's/DVD's) on which open source Software is made available, no other initial cost is usually incurred to obtain the product. Commercial Software on the other hand, normally does have an initial price tag associated with it.

However, not all commercial software has a large initial price tag. Some commercial software providers are making their software available via lease agreements [CIMNET 2004:Internet]. Others are using third parties to provide financing for leasing options[INGRAM MICRO 2005:Internet]. There are also independent software leasing companies that provide leasing alternatives for many popular commercial packages[ASAP 2005:Internet]. A number of software vendors also offer organisations the ability to make use of transaction-based charging schemes (Taub 2001:9). These payment mechanisms offers organisations a method of only paying for what they use with regard to the software they employ.

Not all software starts out as commercial software, however. Some applications have also gone the commercial route after gaining widespread adoption offering initial versions of the product at no charge. Whereas Sun Microsystems' Star Office 5.2 was freely obtainable by anyone (as a free download from Sun's website), Star Office 7.0 [Star Office 2005:Internet], it's latest incarnation, is only available for purchase if not being used for educational purposes[Broersma 2002:Internet].

3.2.1 Open source licensing types

When a new software product is acquired, care should therefore also be taken to understand possible licensing implications and potential future changes to the price tag associated with the product. Certain licenses prohibit any future commercial gains if the source code is used in a product (or to at least make all additional source code available as well), whilst other licensing schemes contain no such exclusivity statements.

Rosen [2001:Internet] mentions various types of software licenses along with the considerations to bear in mind for a specific product. He elaborates on BSD-type licenses which impose no restrictions on licensees, for example to create proprietary versions of applications for commercial gain, and GPL-type licenses which require licensees to make use of the same license as the original software product for any altered software as well.

An example of the different type of licensing models [Conrad 2005:Internet] can be found in two popular open source Relational Database Systems: MySQL [MYSQL 2005:Internet] and PostgreSQL [PostgreSQL 2005:Internet]. **MySQL** makes its product available under two different license types:

- GNU General Public License (GPL): For software projects that are 100 percent GPL, a form of open source license, this license may be used, and no payment is required for MySQL usage. The new software project must therefore also make all its source code available.
- Commercial License: For software products developed for commercial gain, this license must be used and the business division for MySQL implementations will receive remuneration for the implementation of MySQL in the commercial product.

PostgreSQL has a much simpler license and allows software products to make use of it whether for commercial or non-commercial implementations and without requiring making the source code available.

Warrene [2005:Internet] elaborates on four distinct categories of open source licenses:

- Academic licenses: Places no requirements whatsoever on the user of the license. Examples of licenses in this category include BSD type licenses.
- Reciprocal licenses: Derivatives of the software must be released under the same license as the original software product. Examples include Linux and MySQL.
- Standards licenses: A license type that seeks to create a base standard of software and documentation in order to preserve a "core" piece of software whilst allowing modifications through "plug-ins".
- Content licenses: Covers elements aside from code, such as art.

An open source product therefore falls within one or more of the abovementioned licensing categories. Figure 3.2 indicates the license usage summary from a popular open source development site of all the products as of May 2005 compared to January 2005 and June 2004:

Figure 3.2 Open source products licensing type usage

License	**June 2004**	**January 2005**	**May 2005**
GPL	**32,234**	**40,321**	**43,955**
LGPL	4,834	6,463	7,185
BSD	3,210	4,177	4,602
Artistic	1,055	1,191	1,219
MIT	795	1,162	1,167
Apache	784	1,041	1,372

Source: SourceForge 2005b:Internet

As seen in the figure 3.1 the GPL, a reciprocal type license, is by far the most popular license type for open source development currently, with 43,995 products making use of this license type in May 2005. This would indicate that the majority of software applications require licensees that release derivative software to release it under the same license agreement as the original software product.

Looking at the first aspect of total cost of ownership, the *cost of purchase*, it is apparent that there are various issues to consider including the following four aspects:

1. The purchase price is the first element to consider when looking at the total cost of ownership of a software product. Commercial software offers various alternatives to paying a simple once of price, including leasing and transaction based payment mechanisms.
2. Although open source usually requires no initial purchase fee (apart from the cost of distribution material), it is important to understand the various licensing implications. It helps in identifying the possible future direction

which a software product may take with regard to its licensing – for example making the product available for free at a certain point in some cases do not guarantee its price tag in the future.

3. License implications should also be considered when choosing an open source application with a view to modify it to suit an organisation. When a product with a reciprocal license is chosen, the same license must also be used for the modified project.
4. Finally, the open source alternative also places the buyer in a stronger negotiating position when discussing pricing with commercial software providers which can also be used to reduce the total cost of ownership for current commercial investments.

When evaluating the initial purchase in the software selection decision it is important to understand the issues involved for both open source and commercial software including the different mechanisms of payment available for commercial software as well as licensing implications with regard to open source.

3.3 THE COST OF IMPLEMENTATION

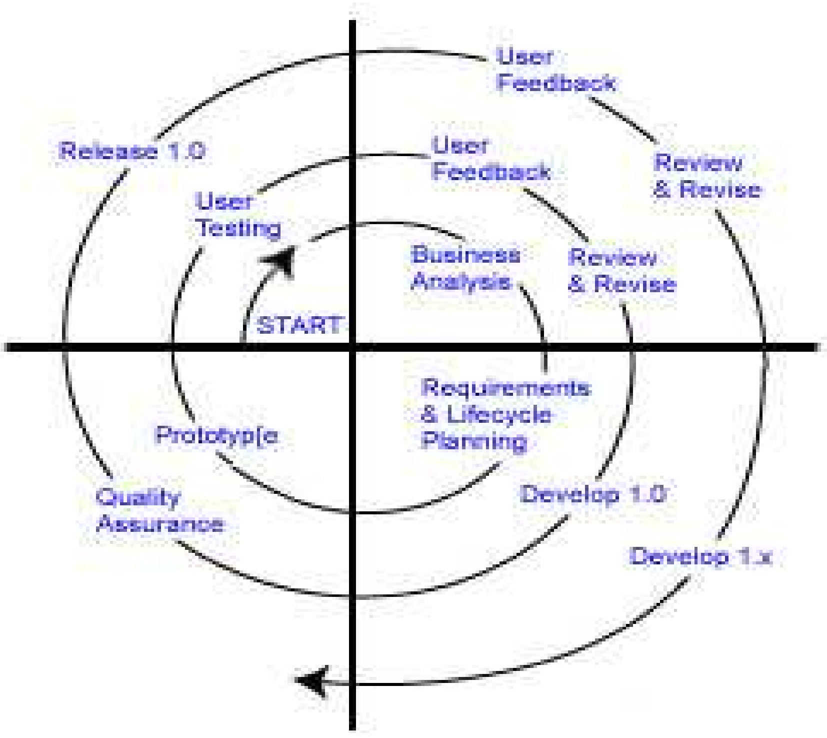
The initial purchase of a software product is only the first step in a series of steps which need to be performed in order to extract value from the product. DeGiglio [2004:Internet], principal analyst at the Robert Frances Group [RFG 2005:Internet] is of the opinion that the software implementation phase can be up to four times that of the software purchase cost.

Organisations may find that there are a lot of resistance to an attempt to change the status quo with regard to the current software install base. Employees used to a specific work environment may be against the change to open source, especially if the software user interfaces differ from what they are used to. A well devised change management plan can reduce the cost of disruptions while implementing the new software.

Following proper change management practices can allow mediocre software solutions to succeed, whilst the lack of a change management programme can cause the best technologies to fail [Carey 2002:Internet]. Issues to consider during the implementation process, which will ultimately result in a less costly exercise include the following six points:

1. Gain buy-in from the users at the implementation's earliest stages. This could include explaining the software implementation process as well as the benefits it might offer the users.
2. Address employees' concerns up front, for example employees might feel that the software would make them redundant or marginalised. These concerns should be surfaced and fully discussed.
3. Explain the rationale behind the decision. This would involve discussing the reasons for the software implementation as well as the expected gains from it.
4. Get buy-in from senior management. Without senior management buy-in a project is bound to fail. Having buy-in also provides a strong external motivation for employees to go along with the implementation process.
5. Start with smaller projects. In other words, if possible opt for an incremental rollout instead of a global once off approach.
6. Leverage methodologies and best practices found in the realm of software development [Carey 2002:Internet], as figure 3.3 indicates:

Figure 3.3 Spiral development model including user input



Source: Carey 2002:Internet

The above spiral model includes user involvement from the early stages and defines chronological activities which can be functional during the implementation stage as well. The process starts with a thorough business analysis to understand the business rationale for implementing the software product. The next step is to conduct a requirements analysis to ensure that the software implementation will meet the user and business requirements. Thirdly a prototype implementation can be done with a small subset of users in order to understand the dynamics of the application and the product acceptance from the users. After user feedback and review takes place, implementation can be done on a larger scale. As the circles in the spiral widens, the implementation knowledge and install base grows.

Novell Consulting [Novell 2005:Internet] mentions that organisational structure, user categorisation and application dependencies should be well understood before starting with the implementation process. Low-cost, low-risk pilot projects, such as deploying infrastructure software (web, email or other back office services) should be considered initially.

The cost of implementation can also be reduced by making use of consultants specialising in open source implementations throughout the enterprise [NOVELL 2005:Internet]. Making use of external expertise that has been gained from several implementations should speed up the implementation process and can aid in lending credibility to the change management programme.

In a report released by the UK Office of Government Commerce [UK Office of Government Commerce 2002:2] it is stated that the UK Government will consider open source software (OSS) solutions alongside commercial offerings on a value for money basis. This report continues by declaring that the UK Government will only implement future products that support open standards and specifications. This is an important consideration for the implementation process.

Products, whether commercial or open source, should adhere to the standards as applied in that product space. This simplifies the implementation process and allows for interoperability, which, if not provided by the software, can greatly increase the technical hitches experienced and therefore the costs involved.

When addressing the issues involved in the implementation phase it is apparent that a suitable change management plan is needed to ensure a streamlined software introduction process. Activities to perform in this phase in order to ensure efficient and effective uptake can be similar to those found in the software development life-cycle, these include:

1. Organisations with little current open source implementation experience can expect a higher expenditure (at least initially) compared to commercial software implementations.
2. The implementation of standards can reduce the implementation burden.
3. Change management plans are one mechanism for lowering the overheads during the implementation process.
4. Certain companies outsource implementation expertise and can assist organisations in streamlining this aspect of Total Cost of Ownership.

Ultimately a well planned implementation process reduces the Total Cost of Ownership of a software application and helps to ensure efficient and effective uptake within the organisation.

3.4 THE COST OF MIGRATION

Switching over, or migrating, from one software solution to another can be a daunting task. Various open source distributions such as Red Hat [2005:Internet] and SUSE [2005:Internet] attempt to smooth the migration from a commercial software environment to an open source environment. These distributions provide a collection of commonly used desktop and server software products such as mail servers, windows network interconnectivity, web servers, office applications and other productivity tools that are found in the Microsoft Windows environment.

However, finding capable open source alternatives to some business applications is still problematic. The availability of accounting and enterprise resource planning (ERP) applications and the functionality available in these products are still not as comprehensive as found in commercial counterparts. In a study on the viability of open source in the UK Office of Government Commerce [Office of

Government Commerce 2004:Internet] it was found that business applications are still immature and the scope of those available are still limited.

In the desktop category (including "office" personal productivity suites), this study also found that although these products have been under development and are growing in functionality there has been little significant widespread implementations. A report by Gartner [Silver 2005:Internet] indicated that compatibility and functionality issues make open source alternatives thorny to adopt across the complete organisation. The report continues by stating that through 2008 open source office products will not be suitable for all users in 80 percent of companies. In other words, most companies will be required to support two products: one open source and one commercial. Ultimately, companies already running commercial office products feel that "already bought and installed is cheaper than free" [Silver 2005:Internet].

Other reports indicate that open source software is a feasible and credible alternative for meeting the requirements of the majority of desktop users [UK Office of Government Commerce 2004:Internet]. In an article on Infoconomy [Bradshaw 2005:Internet], it is stated that although open source products lack some of the advanced features found in commercial counterparts, this may in fact be an advantage and that organisations are sold "too much software" and do not use most of the features available in these products.

Infrastructure software, on the other hand, which includes firewalls, mail servers, application and file servers as well as databases are rated as much more mature. On 2 September 2005, the Lower House of the German Parliament announced that it had completed its migration of all servers to the Linux operating system [IDABC 2005:Internet]. The 100 servers have been migrated from Microsoft Windows NT to Linux with all the desktop computers (5000) still running Microsoft Windows XP. The project was not without its delays, however. After the initial launch of the project in May 2003, various compatibility issues between the Linux servers and desktop machines hampered the progress. An interim report has indicated that the operational costs of running Linux could be marginally less than those of running Windows. This experience has also led to the German public sector establishing an open source competence centre aimed at facilitating the spread of best practices in this environment.

Migration efforts from commercial Unix to Linux, however, are much less complex than those of migration from Microsoft Windows environments. Red Hat [2005:Internet] provide detailed white papers regarding Unix to Linux migration considerations. Ultimately, users and support engineers comfortable with the Unix environment will feel much more at home in the Linux environment than those only comfortable within the Microsoft Windows environment.

As of September 2005, figure 3.4 can represent an open source enterprise stack, in other words, the software required to run the average enterprise from infrastructure software through to desktop software:

Figure 3.4 The free enterprise stack

Application Server - JBoss, Zope
Browser - Mozilla, Konqueror, Galeon
Business reporting - BIRT
Content Management System - Mambo, OpenCMS, Red Hat CCM, PHP-Nuke, Plone
Database Systems - PostgreSQL, Ingres, Cloudscape, MySQL, SQLite, FirebirdSQL
Desktop Environments - KDE, Gnome, Ximian
Email - KMail, Mozilla, Ximian, Mutt
Finance - Jcash, Jmoney, GnuCash
Groupware - Opengroupware.org
Instant messaging - Jabber, Fire, Exodus, Kopete
Internet voice - Catzilla, Speak Freely
Office Applications - OpenOffice, KOffice, Gnome Office, AbiWord

Operating Systems - Linux, FreeBSD, FreeDOS, Haiku, Menuet
Password management - Password Safe, PINs
Personal information managers - Kaddressbook/KNotes/KOrganizer, Mozilla Calendar
Security - Open Antivirus, NeoCrypt, Coyote Linux Firewall
Web Server - Apache
Windows managers - Blackbox, Openbox, Window Maker, FVWM

Source: Bradshaw 2005:Internet

Conclusions drawn from the study done by the UK Office of Government Commerce [UK Office of Government Commerce 2004:Internet] indicated that although open source software is a viable alternative to commercial software for infrastructure and the majority of desktop users, careful consideration is required to understand all the interoperability issues that could be faced such as the required interaction with other applications for day to day usage.

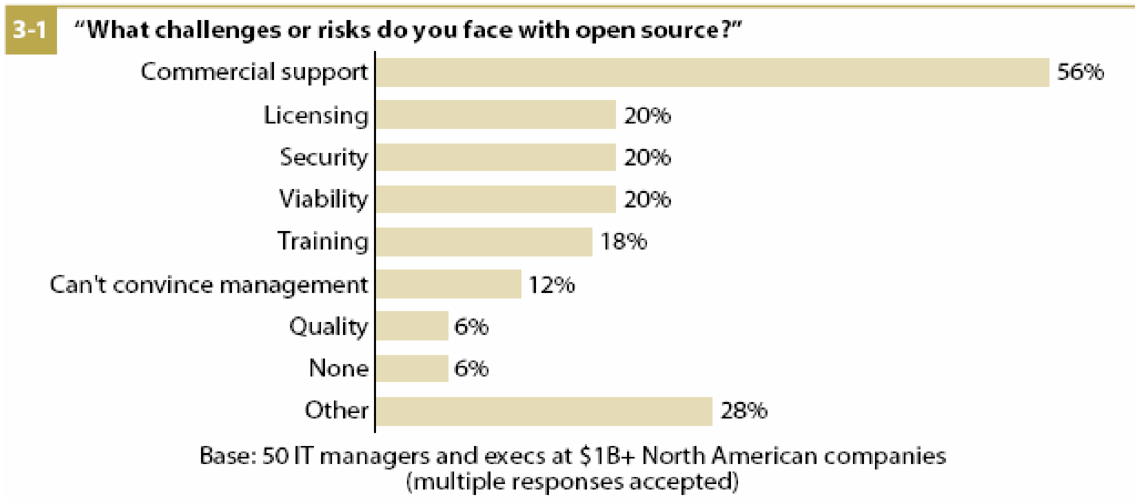
Migrating from commercial software to open source requires careful consideration and investment in planning. Although open source software provides alternatives for various software applications, these should be fully understood before taking the bull by the horns.

Some arenas, such as infrastructure, contain a lot more mature open source products than others such as business applications. Office productivity applications are available and can suit a large proportion of the workforce that does not need advanced features offered in commercial counterparts. Some consideration should also be given to situations where dual product support between commercial and open source software will be required.

3.5 THE COST OF SUPPORT AND MAINTENANCE

Ongoing product support and maintenance is one of the key activities during the software life-cycle. Figure 3.5 indicates that 56 percent of IT managers rated commercial support as a risk or challenge with regard to open source decisions:

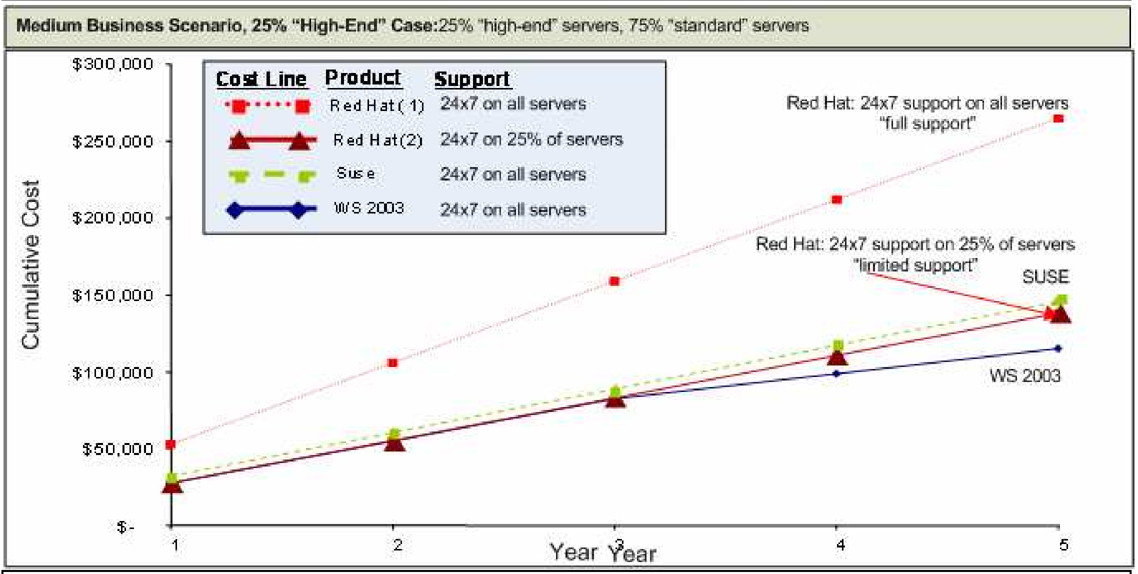
Figure 3.5 Challenges faced with open source



Source: Schadler 2003:6

In a study done by BearingPoint [2004:Internet], it was found that open source support costs could be even more than support costs for commercial software, as indicated in figure 3.6:

Figure 3.6 Support costs for medium businesses

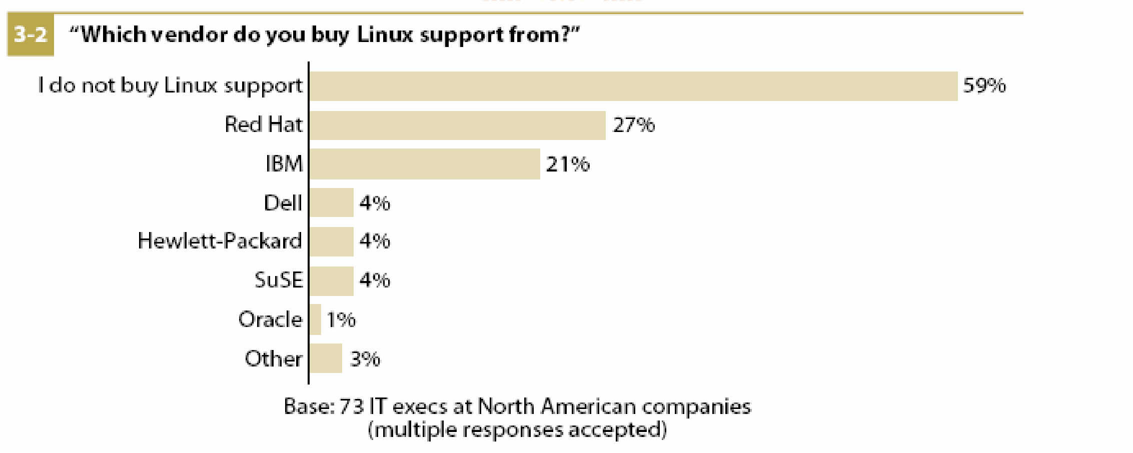


Source: BearingPoint 2004:Internet

In figure 3.6, it is clear that the support cost of Red Hat is considerably higher than that of Microsoft or of other Linux distributions such as SUSE. However, of all the server software tested, Microsoft Windows Server 2003 support was the least expensive. This is to be expected as open source companies provide the software for a low price and the business model is based on charging for services and maintenance instead [OpenXource 2005:Internet].

However, not all companies making use of open source make use of commercial support. In a survey by Forrester (Schadler 2003:6) of 73 IT executives, 59 percent of respondents do not buy Linux support as the figure 3.7 illustrates:

Figure 3.7 Commercial Linux support purchases



Source: Schadler 2003:6

Of those IT executives that do purchase commercial Linux support on the other hand, Red Hat receives the highest amount of those support queries. According to figure 3.6 these executives would be paying a premium to other open source and commercial providers. However, open source support does not necessarily imply paying a large premium to commercial software providers, as figure 3.6 indicates with the SUSE Linux product support costs.

Open source software support can also come from in-house expertise. It is therefore important to consider the availability of the open source skill base compared to the commercial software skill base. The accessibility of this skill is a central consideration in the decision to make use of in-house expertise vs. outsourcing support requirements. In an article entitled “Linux vs. Microsoft [Total cost of ownership] debate rages” [Best 2005:Internet], it is stated that although people with Linux skills are in short supply, the numbers are growing.

3.5.1 Stability and reliability

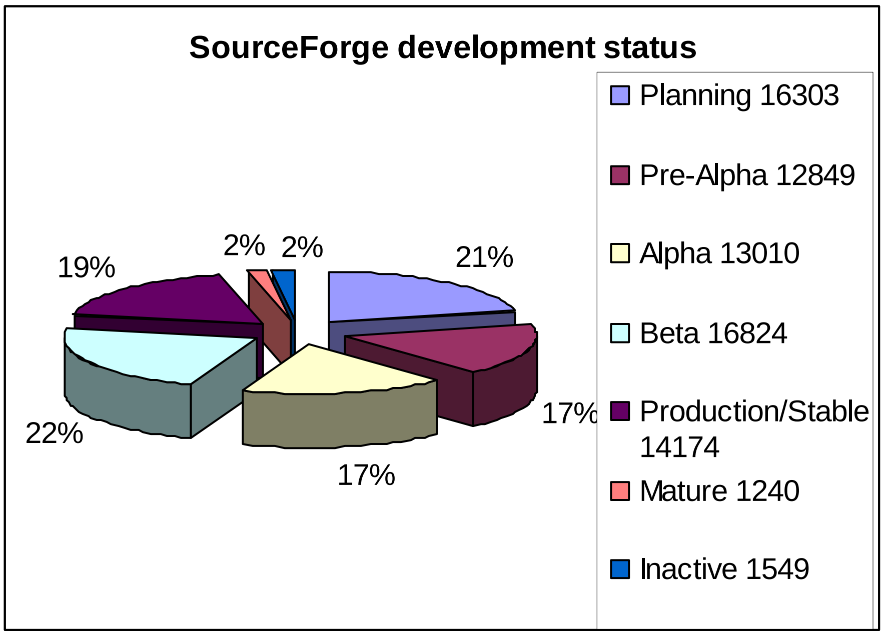
Support and maintenance go hand in hand with reliability and stability. The more reliable a software product, the less maintenance is required to keep it running. Open source products offer different levels of reliability in different arenas.

In the back-end arena, many open source products have proven to be very reliable. Apache [2005:Internet], an open source web server, has been the most popular web server on the Internet since April 1996. In fact it has been used more widely than all the other web servers from both open source and commercial vendors combined.

In the data centre arena, PostgreSQL and MySQL are both widely used open source databases, but do not provide the features of some commercial counterparts in the high-end enterprise environment yet, with Phillip Moore, an executive director at Morgan Stanley Dean Witter & Co in New York stating: “Commercial software have more to offer for a commercial venture like ours” [ComputerWorld 2001:Internet]. These open source applications runs on the Linux operating system that continues to prove its stability even in the high end server operating system arena [Linux.org 2005:Internet].

To assist with a product's readiness for prime-time, SourceForge [2005b:Internet], one of the popular open source development sites, also provides a detailed breakdown of a product's current development status. An overview of the current product statuses as available on this site is provided figure 3.8. Each status is provided with the amount of applications currently falling within that status:

Figure 3.8 SourceForge development statuses



Source: SOURCEFORGE 2005:Internet

As figure 3.8 indicates, only 21% of the current projects in development are in the production/stable or mature categories. The rest (77%) of the projects currently in active development are not yet considered stable or mature and 2% is currently inactive.

After addressing the above support and maintenance issues to consider, the following aspects should be evaluated with regard to total cost of ownership:

- The cost of Linux support compared to that of commercial software. In the above comparisons, certain distributions are in line with commercial support whereas others are significantly more expensive.
- The level of maturity of each open source product should be taken into consideration. Infrastructure products tend to be more mature currently than office productivity or business applications. Theoretically the more mature a product the less the support and maintenance costs.
- The current level of in-house expertise for supporting the open source install base. Low levels of expertise can equate to lower productivity. The degree of investment to increase this level must also be taken into consideration.

When looking at the cost of support and maintenance, the availability of open source support must be considered in the locations where an organisation does business. Currently Linux support is available from third parties, but can prove to be more expensive than commercial support. Increasing the internal skill base with regard to support and maintenance of open source is another alternative, but availability of reputable training institutions should be investigated.

Some applications such as in the back-end and data centre arenas are more mature and should therefore prove to require less maintenance in terms of ensuring optimal uptime. It is useful to look at a product's development status in order to gauge potential support requirements.

3.6 THE COST OF TRAINING

Training users to become proficient in their day to day IT tasks is crucial to ensure increased productivity which finally affects the bottom-line. In responses from a group of American companies [Giera 2004:Internet] it was found that training for Linux was more robust and more costly than related training on Microsoft Windows. This was attributed to two main reasons:

1. Training material was less available for Linux than for Windows
2. Customers felt that they would need to schedule more training to overcompensate for the lack of internal Linux knowledge compared to the knowledge built up over several years in the Windows environment.

This existing knowledge in a product is an important consideration when deciding to move to a new software application. The trade-off between expenditures to get up to speed on the new product should be weighed up against the value of existing knowledge of employees on a current product.

Kenwood [2001:Internet] states that open source tends to have a weaker graphical user interface than commercial software and could make it less practical to use currently for desktop applications. This also translates into a less user friendly environment which increases the costs of training on open source applications. However, for certain software applications which require technically skilled employees, such as in the backend or server arena, the graphical user interface is less of an issue, and therefore not such a drawback. Graphical user interfaces are also continually improving in the open source arena, with some desktops mimicking the look and feel of the more familiar Microsoft Windows [GTK Wimp 2005:Internet].

It is also advantageous to identify potential training centres geographically nearby to the organisation. Although online training is becoming more of a reality, many users may still find face-to-face training more effective. LinTraining [2005:Internet] provides information about Linux training centres around the globe and currently provides listings of 668 training centres, with 15 centres presently listed in South Africa.

Certain country-wide open source training programs are also appearing. Malaysia has embarked on a low cost PC initiative, which revolves around open source. In this regard it is offering "training to trainers" to assist in the rollout of open source endeavours. The Malaysian National Computer Confederation (MNCC) believe that the training will assist the industry adoption of open source and will help the retailers to support the low cost PC initiative. The programme is also open to all groups including training centres, institutions and individuals. [Smith 2004:Internet].

In South Africa, the Shuttleworth Foundation [2005:Internet] is also funding various open source initiatives. One of these initiatives is the Learning Linux Material programme. This programme provides Linux training material online and make it available free for download to promote open source awareness and skills growth.

When looking at the abovementioned issues, the following aspects can be included when looking at total cost of ownership from a training perspective:

1. The availability of training on a potential product is an important consideration. Since some employees might still prefer face to face training over correspondence or online training, therefore training centres in close geographical proximity to an organisation can be advantageous. Lack of available training is detrimental to productivity and therefore the bottom line.
2. The current knowledge that employees have of a specific software product or software environment should be evaluated against the cost to train them up in a new environment.
3. Training costs differ between open source and commercial software products and this should form part of the total cost of ownership.

4. Even though recognised training institutions from both open source and commercial software can be more expensive, the cost of this should be compared against the potential value gained from recognised establishments versus potentially cheaper, but not well known institutions.

Ultimately, the cost of training is a vital step in evaluating the total cost of ownership of a product. Poorly trained employees lead to poor performance and ultimately affect the bottom line.

3.7 CLOSURE

By using Total Cost of Ownership (TCO) it is possible to identify various aspects to consider in the "during" and "post" phases that all contribute to the software value or detract thereof. The initial realisation is that software cost is more than simply the purchase price and that issues such as implementation, migration, support, maintenance and training all form part of the cost calculation. Open source and commercial software offer different views on the total cost of ownership and it is up to the informed to understand the issues to evaluate.

Mature open source offerings are available in the infrastructure arena, with desktop and business applications not yet matching the level of maturity of various commercial alternatives. However, many open source offerings are "good enough" with some proponents believing that commercial vendors are "over selling" and most users do not need all the features on offer. Armed with the knowledge on what contributes to value an organisation can make the most appropriate decision as the situation dictates.

CHAPTER 4

4 RESEARCH METHODOLOGY AND RESEARCH FINDINGS

		Page
SYNOPSIS		4-58
4.1	INTRODUCTION	4-59
4.2	RESEARCH METHODOLOGY	4-59
4.2.1	Data gathering	4-60
4.2.2	Data analysis	4-64
4.3	RESEARCH FINDINGS	4-65
4.3.1	Characteristics of respondents	4-65
4.3.2	Perceived value, interoperability and risks	4-67
4.3.3	Software life-cycle considerations	4-71
4.3.4	Pros and cons with regard to open source	4-73
4.3.5	Current product usage	4-74
4.4	CLOSURE	4-76

SYNOPSIS

Chapter 4 deals with the research methodology used in the empirical research as well as the research findings. The exploratory research design was chosen as the most appropriate, partly because of the pioneering nature of the study. The research findings were divided into sections pertaining to the organisational and technical roles of respondents. Furthermore the respondents' view on open source value, interoperability and risks were requested. Respondents also had to prioritise the aspects of total cost of ownership identified in **Chapter 3** as it pertains to open source and commercial software starting with the cost of purchase and ending with the cost of training. Finally respondents were asked to list the positive and negative aspects they would currently associate with open source as well as to list the current applications they make use of for day-to-day operations.

4.1 INTRODUCTION

This chapter discusses the research methodology of the empirical research as briefly indicated in **Chapter 1**. The procedures employed in this research will be discussed and the methodology and findings will be set out in this chapter. As recent introductions and adoptions of various open source applications are fairly new and research into the broad spectrum of perceptions regarding open source are not yet very common, this research is of a pioneering nature.

4.2 RESEARCH METHODOLOGY

To understand the most appropriate method of data gathering, various research design alternatives were investigated. Research design can be defined as "A master plan specifying the methods and procedures for collection and analyzing the needed information" (Zikmund 2003:740). There are mainly three types of research design (Zikmund 2003:58):

1. Descriptive research: This form of research is used to describe certain characteristics of a population or phenomenon. It mainly seeks to answer questions related to *what, who, where, when and how* regarding a problem.
2. Causal research: The goal of this type of research is to identify cause-and-effect relationships among variables. In other words, to understand the impact of changing one variable has on another variable.
3. Exploratory research: This research approach should be applied when the problem is not well defined or ambiguous. Exploratory research is normally conducted with the view that subsequent research will be required to provide conclusive evidence. Exploratory research therefore

aids in "crystallising" a problem. Mouton & Marais (1996:43) defines the following objectives of exploratory research:

- Gathering information and insights.
- Undertaking a preliminary study before more structured research can be followed.
- Defining central concepts and constructs.
- To understand or determine items of importance for future research.
- Aids in the development of hypotheses.

There are four basic exploratory research techniques (Zikmund 2003:61): secondary research, pilot studies, focus groups and case studies. Since this study is relatively new and of a pioneering nature, the exploratory research technique was selected as the most appropriate type of research design.

4.2.1 Data gathering

The objectives of the data gathering and analysis phase of this study are as follows:

1. Gathering data on the opinions of different roles in the business environment.
2. Gathering data on the current perceptions regarding open source including the perceived value, risks and issues concerning total cost of ownership.
3. Comparing the opinions between the different roles in the business environment regarding open source.

A survey of individuals in the business environment was conducted through the use of a questionnaire. This questionnaire was made available online via the World Wide Web and was sent out to individuals in the business environment in the United States, United Kingdom, the Netherlands, Australia and South Africa. The survey could only be filled in once online by each respondent.

The respondents were selected based on their industry knowledge either with regard to commercial software or with regard to open source or both. Respondents were also encouraged to invite others to take part in the survey. The initial respondents were selected based on their interaction with the author.

Respondents in the business environment were selected specifically to understand the views regarding open source from the different roles within a business context. The aim of the research was to understand the perceptions regarding open source as a value alternative to commercial software firstly from a business role perspective (managerial, commercial, technical) and secondly from a technology role perspective (user, implementer and developer/programmer).

The survey was sent out to three initial respondents to test the clarity and layout of questionnaire items. After suggestions by these respondents have been implemented the questionnaire was made available to all respondents [**Appendix A**]. An invitation to the survey was sent out via email, including a link to the online survey. Statistical processing was done by the online survey provider in conjunction with Microsoft Excel 2003 (Excel:2003).

4.2.1.1 Benefits of the survey as a data gathering method

According to Zikmund (2003:175) surveys provide quick, inexpensive, efficient and accurate means of assessing information about a population. Surveys are quite flexible and when properly conducted can be extremely valuable to managers.

Furthermore, by using an online survey provider, respondents have the convenience of answering the survey in their own time. The knowledge and application abilities of the online survey provider also offer quick statistical processing and reduce the probability of errors during processing. The online survey provider can also increase survey validity by such measures as preventing a respondent from completing the questionnaire more than once as well as ensuring respondent anonymity.

However, there are also disadvantages to the survey method (Cooper & Emory 1995:287):

- *A low response rate*: More responses were completed online than the number of invitations sent out as respondents invited colleagues to complete the survey. As most respondents were quite interested in the contents of the survey they were quite eager to complete the survey.
- *No interaction between researcher and respondents*: Although interaction was minimal in this survey, each question was well explained and elaborated upon. In this case a low level of interaction was preferable as the researcher did not want to influence the responses and some respondents preferred anonymity.
- *Questionnaires can be too long and complex*: The questionnaire was purposefully kept short by asking only ten questions so that respondents did not feel overwhelmed by the survey.

4.2.1.2 Survey content and format

The survey consisted of a total of ten questions and contains 3 open and 7 close-type questions. This number of questions was chosen specifically to ensure a high response rate (as respondents tend to shy away from lengthy questionnaires), secondly each question was specifically chosen to identify an aspect of value with regard to open source and commercial software and to link these perceptions to the business and technical roles of a respondent or groups of respondents.

The first two questions were asked to ascertain the specific business and technical roles of the respondent. Closed questions were used (where a respondent is forced to choose between alternatives). Benefits of this type of question is that the respondent is forced to choose between the alternatives and furthermore that the analysis could be done on well defined roles.

Question 3 was regarding the environment in which each respondent felt open source contributed the most value, with question 5 regarding the most prominent risk currently inherent in the use of open source. Both questions provided a set of pre-defined options as well as the ability to specify a new alternative. Most respondents chose to make use of the pre-defined alternatives however.

Question 4, 6 and 7 used category rating scales. Question 4 asked each respondent their opinion of the level of standards implementation in both open source and commercial software with a 5-scale rating of *poor, average, good, very good and excellent.* Question 6 and 7 aimed to rate each respondent's opinion on the importance of each activity of total cost of ownership as discussed in **Chapter 3** for both open source and commercial software. The five-point Likert rating scale consisted of *not important, slightly important, moderately important, very important and crucial.*

Question 8 and 9 were open ended questions and aimed at understanding what each respondent viewed as the pros and cons of open source. Question 8 was entitled: *"In your opinion, what is WRONG with open source"*, with Question 9 asking: *"In your opinion, what is RIGHT with open source".*

Question 10 acted as a control question. Each respondent was asked what software they employed for various tasks, such as word processing, Internet browsing and their spreadsheet of choice. This question had no pre-defined alternatives and each respondent had to type in the name of each software application. This was done to ensure that the respondent do not simply make a tick mark next to a pre-defined software application and to ensure that if a respondent made use of open source or commercial alternative, they would be able to specify the names of the software applications they employ. This question was also aimed at understanding what the current popularity of open source and commercial software applications are for the pre-defined categories.

4.2.2 Data analysis

In most part descriptive statistics were employed to indicate frequencies (percentages) and comparisons between various aspects as identified by the survey.

For question 4 the ANOVA statistical test was also used to test for reliability regarding respondents' ratings with respect to interoperability for open source and commercial software.

For question 6 and question 7 the t-test was employed to test the correlation between respondents with a technical role and respondents with a commercial or managerial role regarding the assignment of priorities to the aspects of total cost of ownership as it pertains to commercial software and open source software.

The data was analysed by an online survey provider and by using Microsoft Excel 2003 (Excel:2003) .

4.3 RESEARCH FINDINGS

Meaningful statistical relationships and significant statistical differences are discussed in this section. Various categories of results are addressed in the following sections.

4.3.1 Characteristics of respondents

Individuals performing various roles in the business environment took part in the survey. The roles were divided into organisational and technical roles. The organisational roles relate to the business function of an individual and the technical roles relate to the level of software involvement for the respondent. Each respondent had to choose only one role. If the respondent performed more than one role, he/she had to choose the most dominant role. The organisational roles consisted of the following four categories:

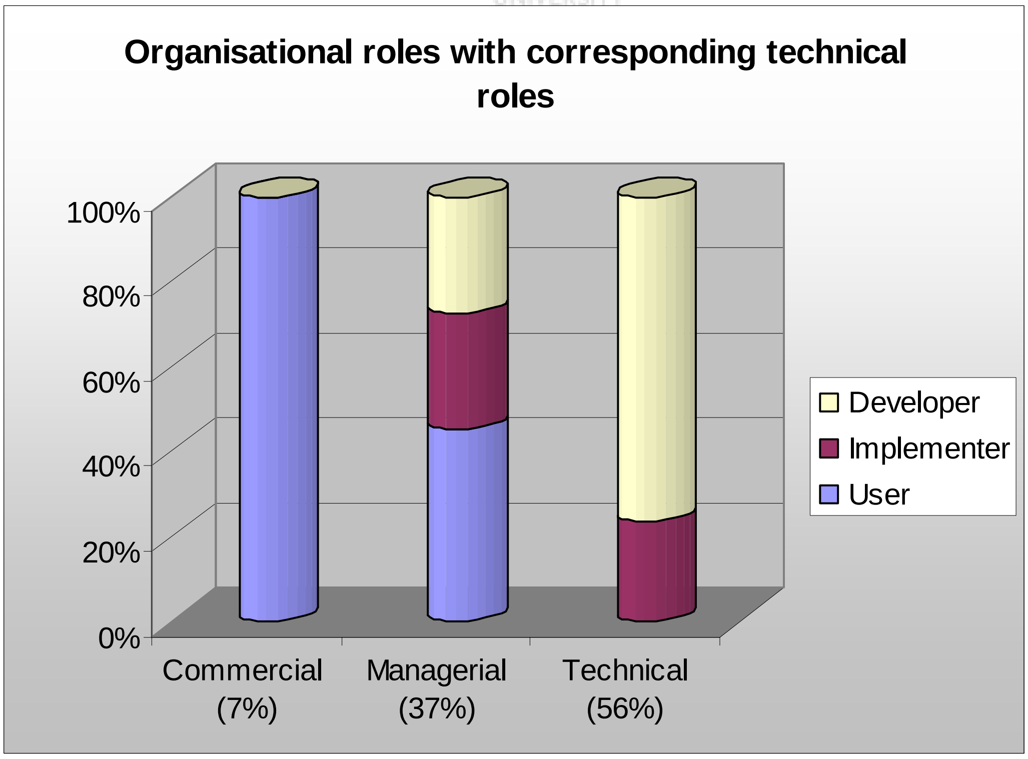
1. *Managerial* – performs mainly management oriented tasks within the organisation.
2. *Commercial* – performs mainly commercial oriented tasks within the organisation. This category includes the Sales and Marketing departments of an organisation.
3. *Technical* – performs mainly technically oriented tasks within the organisation. This category could include individuals from the systems administration department as well as software developers.
4. *Administrative* – performs mainly administrative tasks within the organisation.

The technical roles examined each respondent's level of technical proficiency as well as the main function they perform with regard to software. The technical roles consisted of:

1. *User* – An individual spending most of his/her time actually using a set of products, who is not mainly concerned about the installation or development of these products.
2. *Implementer* - An individual spending most of his/her time implementing and maintaining a set of products for other users.
3. *Developer* - An individual spending most of his/her time developing (programming) software applications for others to make use of.

Figure 4.1 represents the organisational as well as technical roles as a percentage of total respondents:

Figure 4.1 Organisational roles with corresponding technical roles of respondents



Source: Result of research

As the figure 4.1 indicates, no respondent selected the *administrative* category, with 37% selecting the *managerial*, 56% the *technical* and 7% the *commercial* categories.

The technical roles of the abovementioned categories are also shown in relation to the organisational roles. 54% of the *managerial* category classified themselves either as *implementers* or as *developers*, with the other 46% classifying themselves as *users*. 100% of the *commercial* respondents classified themselves as *users*. In the *technical* category 23% classified themselves as *implementers* with 77% classifying themselves as *developers*

4.3.2 Perceived value, interoperability and risks

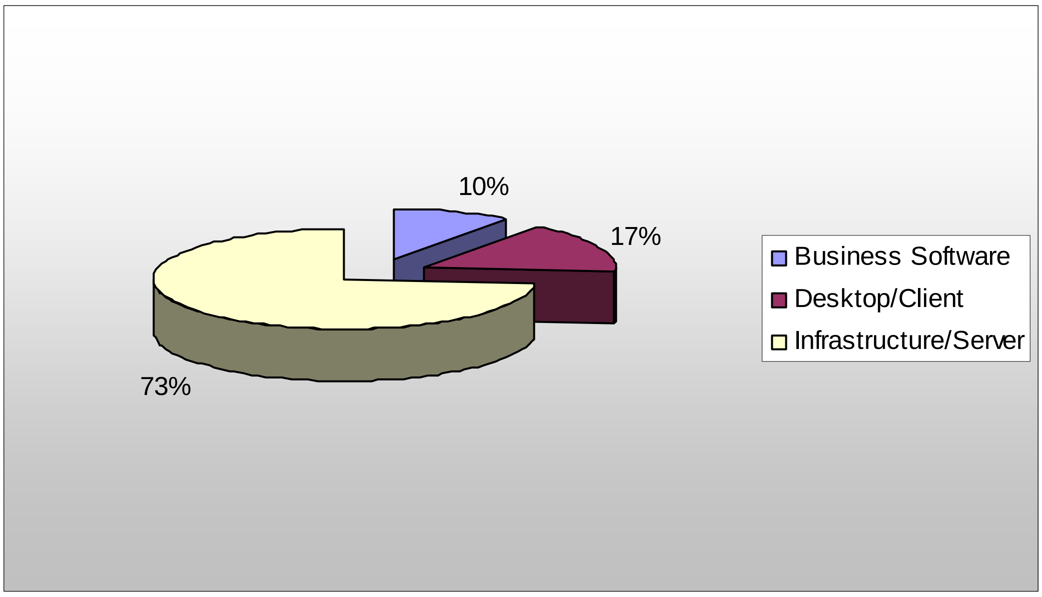
4.3.2.1 Perceived value

Perceived value refers to the software environment in which respondents felt open source contributed the most value. Question 3 asked respondents in which category they perceived open source to contribute the most. The pre-defined categories were:

- *Business software* – Software that exist in both the back-end and front-end of an organisation, such as ERP (Enterprise resource planning) software, accounting software and other administrative software applications.
- *Desktop/Client software* – The front-end environment where a user would spend time performing daily tasks. Software in this category includes office productivity software such as Microsoft Office and OpenOffice and mail client software such as Microsoft Outlook and Mozilla Thunderbird.
- *Infrastructure/Server software* – The back-end or server environment running web-servers, mail-servers and file-servers amongst other things.

The results of perceived value in this environment can be seen in figure 4.2:

Figure 4.2 Environment in which value is contributed by open source



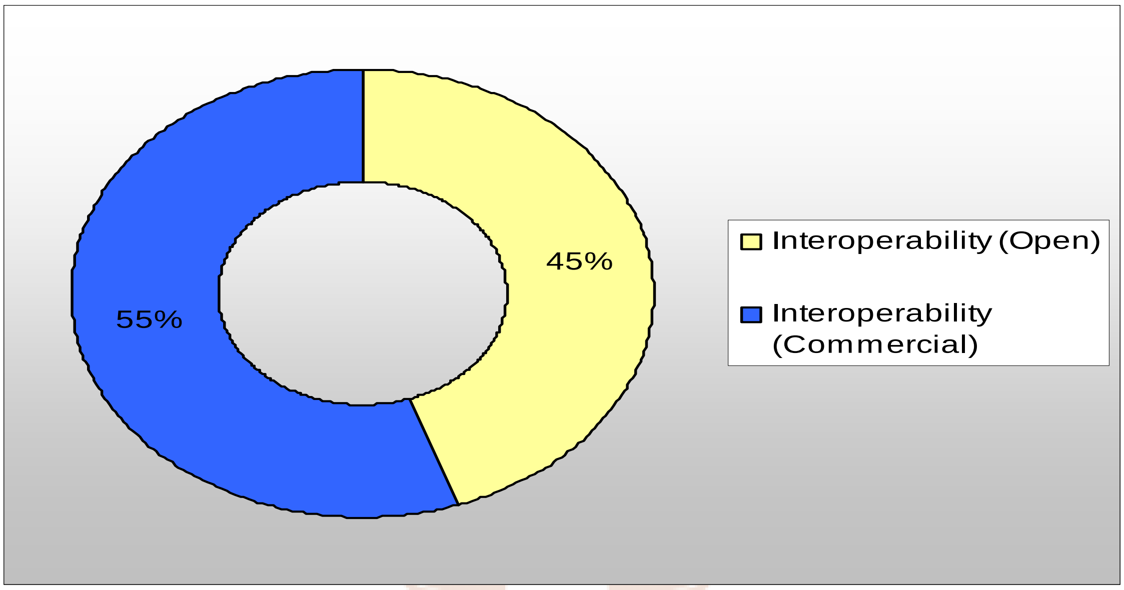
Source: Result of research

An overwhelming majority of the respondents (73%) were of the opinion that open source currently contributes the most value in the infrastructure/server environment rather than the desktop (17%) or business application (10%) environment.

4.3.2.2 Interoperability

Interoperability refers to the implementation of standards within a specific software environment in order to share data and interact with other software applications. It therefore also refers to the level of integration of a specific software application within the larger software environment of an organisation. Question 4 asked respondents their opinion regarding the interoperability of open source applications compared to that of commercial software. As businesses are increasingly working together and the Internet is bringing more and more organisations together, it becomes imperative that the corresponding software systems can interact. This can only be achieved when the software applications support standards which allows for interoperability. Results on the view on interoperability for open source compared to commercial software can be seen in figure 4.3:

Figure 4.3 Interoperability: Open source and commercial software



Source: Result of research

After the total score for interoperability have been tallied for the respondents with regard to both open source and commercial software, the interoperability score for commercial software was 10% higher than the corresponding interoperability score for open source software. This indicates that respondents were of the opinion that commercial software exhibits a higher level of integration within the larger software environment of an organisation and a higher level of potential information sharing with other applications is possible compared to current open source alternatives.

An ANOVA test (two-factor without replication) was performed on the data from respondents with respect to their interoperability ratings for both open source and commercial software, with a critical value set at 0.05. This test indicated a p-value of 0.002 which is a significant value indicating that the lower interoperability ratings assigned to open source by respondents cannot be attributed to chance alone.

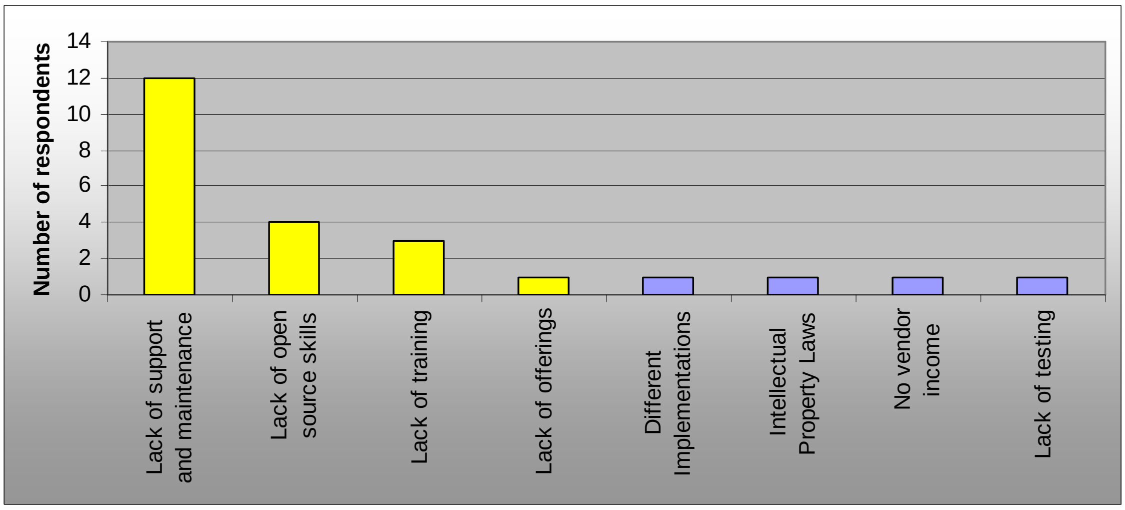
4.3.2.3 Risks

Question 5 was concerned with the perceived risks of open source. The question listed four pre-defined options:

- Lack of training
- Lack of open source skills
- Lack of application offerings
- Lack of support and maintenance

The question also allowed the respondent to provide any additional risk if it was not listed. Figure 4.4 illustrates the perceived risks with regard to open source usage, with the pre-defined risks provided in the survey indicated in yellow:

Figure 4.4 Perceived risks when using open source



Source: Result of research

Figure 4.4 indicates that respondents were of the opinion that lack of support and maintenance was the largest associated risk (50%) when using open source, with lack of open source skills (17%) and lack of training (13%) the second and third largest associated risk respectively. Other risks specified with one respondent per risk are different implementations, intellectual property laws, no vendor income and lack of testing.

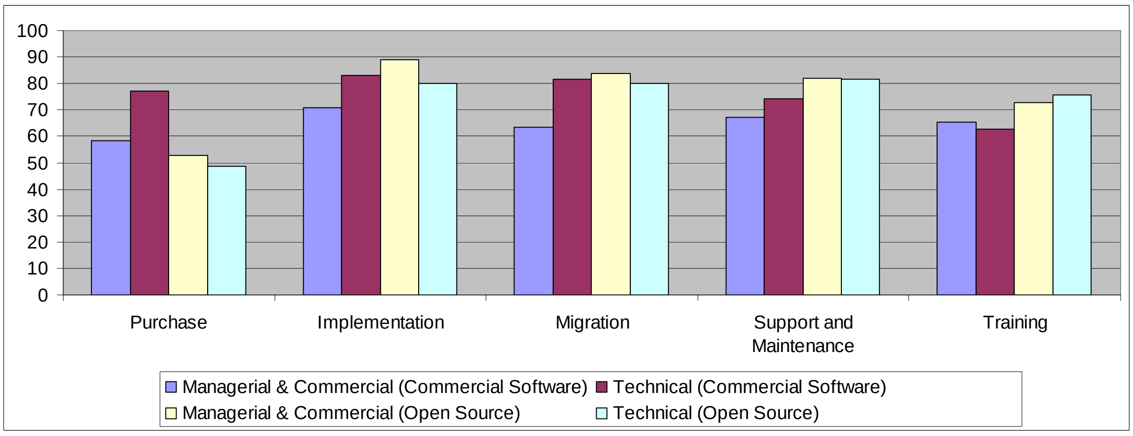
4.3.3 Software life-cycle considerations

Questions 6 and 7 asked respondents the priorities they would assign to activities in the software life-cycle that make up total cost of ownership. The categories assigned in the software life-cycle were:

- The cost of purchase
- The cost of implementation
- The cost of migration
- The cost of support and maintenance
- The cost of training

Figure 4.5 depicts the priorities as they are assigned by those performing a managerial and commercial role, compared to those performing a technical role within the organisation. A further comparison is the priorities as they are assigned to open source compared to commercial software:

Figure 4.5 Ratings assigned to aspects of total cost of ownership



Source: Result of research

As figure 4.5 indicates, apart from the purchase price, open source rates higher in all the other categories pertaining to total cost of ownership. Looking at the rating between the technical role compared to the commercial and managerial role pertaining to commercial software it is apparent that the technical role considered all the aspects of a higher influence apart from the training aspect, where the commercial and managerial roles rated higher. Considering the same aspect with regard to open source it is essentially the other way around with the commercial and managerial roles rating all aspects higher apart from the training aspect which the technical role rated of bigger importance.

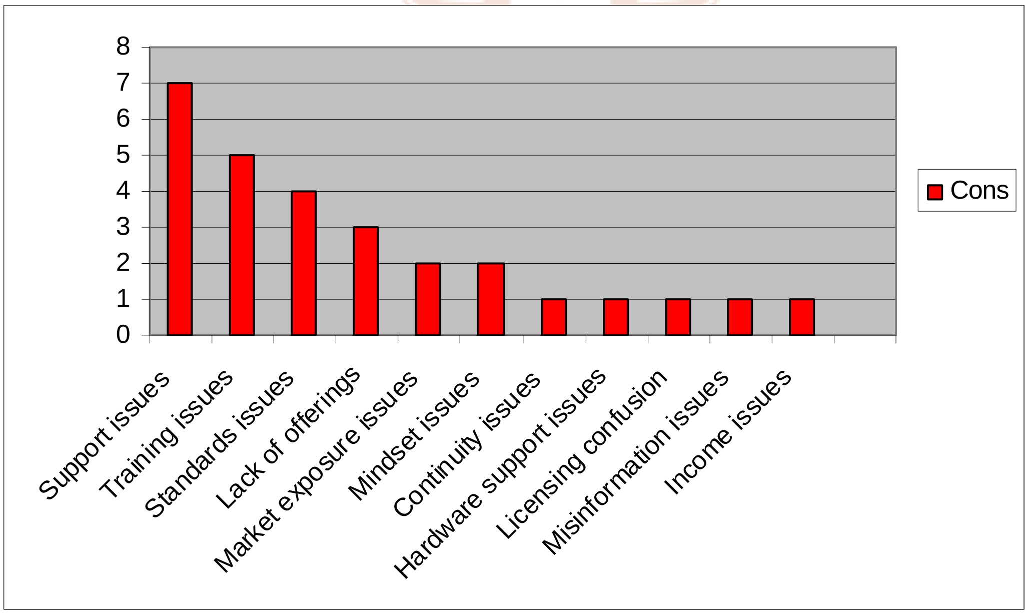
Performing a t-test (paired two-sample for means) based on respondents' answers with respect to open source software, the Pearson correlation value was 0.95. This is a very strong correlation indicating that respondents with a commercial or managerial role concur with respondents with a technical role on the priorities assigned to each of the aspects of total cost of ownership as it pertains to open source software.

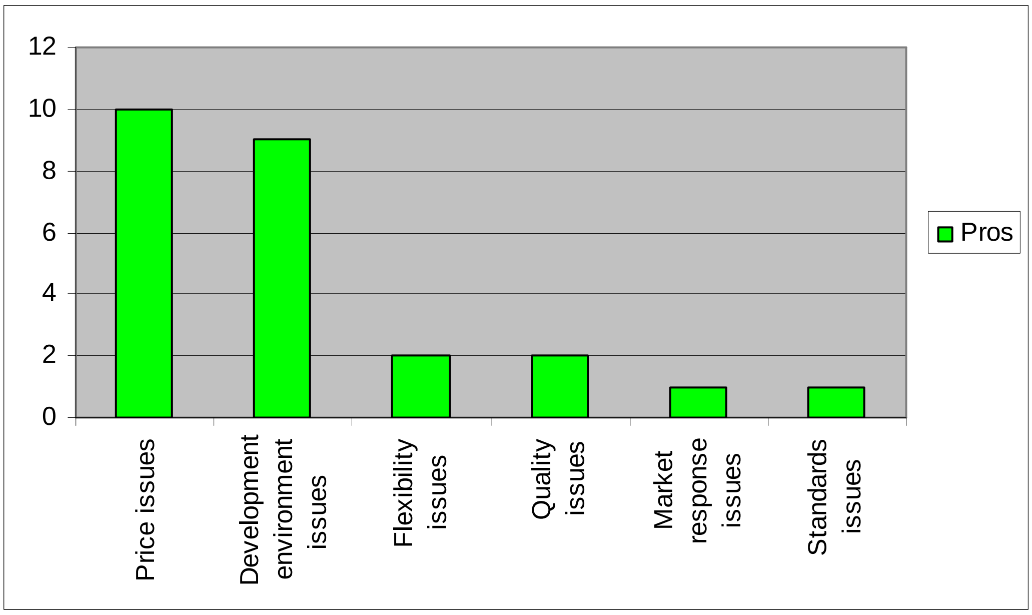
Conversely, the t-test based on respondent's answers with respect to commercial software indicates a Pearson correlation value of 0.11. This point towards a very weak correlation between the priorities assigned to aspects of total cost of ownership between respondents with a commercial or managerial role compared to priorities assigned by respondents with a technical role.

4.3.4 Pros and cons with regard to open source

Two open ended questions (Question 8 and 9) were used to ask respondents to state, in their opinion, what is *right* and *wrong* with open source as it stands today. After the responses have been coded, the resultant opinions are as follows:

Figure 4.6 Open source pros and cons





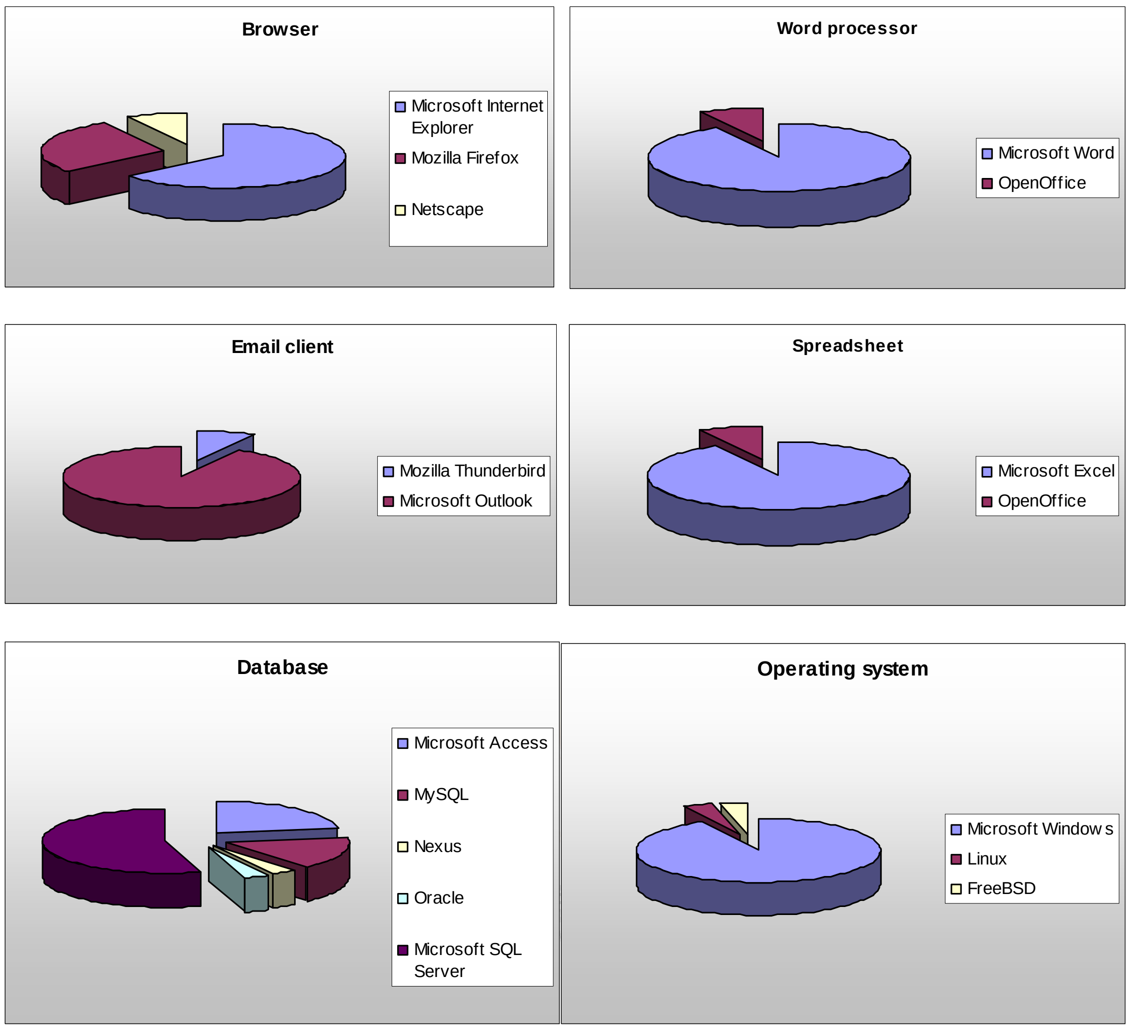
Source: Result of research

Respondents were much more varied on the negatives of open source than on the positives, with the highest negatives surrounding training and support issues. On the positive side, the open source development environment as well as the attractive price rated the highest.

4.3.5 Current product usage

The last question asked respondents their current product usage, mainly with regard to day-to-day software use. The question did not provide pre-defined options and required each respondent to enter the products themselves. The question also acted as a control question to understand from which environment the respondents come from or are used to. Some respondents listed more than one software application in a given category and in such cases both software applications were used to tabulate the results. After coding has taken place, the product usage can be summarised as follows:

Figure 4.7 Current product usage



Source: Result of research

The above figure shows that by and large most respondents still make use of commercial applications for day to day tasks. Interesting to note that respondents using open source applications do not necessarily run the applications on an open source operating system and that most respondents still run their applications on top of Microsoft Windows. However, the product landscape are already looking more varied with users opting for different browser software than Microsoft Internet Explorer and Open Office and Mozilla are also featuring on the product usage list.

4.4 CLOSURE

The survey results have been discussed in this section. Question 1 and 2 provided information regarding the organisational and technical roles of respondents. The organisational roles consisted of options for *managerial, commercial, technical* and *administrative.* The technical roles were regarding software application usage and consisted of *user*, *implementer* and *developer.*

Questions 3, 4 and 5 pertained to value, interoperability and risks. It was found that respondents are of the opinion that open source delivers the most value in the *infrastructure/server* arena. Furthermore, respondents felt that commercial software have a higher level of interoperability or support of standards than open source. Lastly, it was found that the largest perceived risk when using open source was lack of support and maintenance.

Questions 6 and 7 was regarding the priorities respondents would assign to elements of total cost of ownership in the software environment and the different priorities they would assign when considering open source or commercial software. It was found that apart from the *price* aspect, respondents rated all the other aspects higher when considering open source than commercial software.

Questions 8 and 9 were open ended questions and asked respondents to provide their opinion on the positives and negatives of current open source offerings. The negative aspects varied much more widely than the positive aspects, with *support issues* being the most widely mentioned negative aspect and *price issues* being the most widely mentioned positive aspect.

Finally question 10 asked respondents their current application usage for common day to day tasks. Commercial software is still more widely used, but open source alternatives are also becoming more prevalent.

Recommendations to the industry will be made in the **Chapter 5** as well as recommendations for future research.

CHAPTER 5

5 INTERPRETATION AND RECOMMENDATIONS

		Page
SYNOPSIS		5-79
5.1	INTRODUCTION	5-80
5.2	INTERPRETATION OF STUDY	5-81
5.2.1	Software value using the software life cycle	5-81
5.2.2	Total cost of ownership of software	5-82
5.2.3	Factors identified by the empirical research	5-84
5.3	RECOMMENDATIONS	5-84
5.3.1	Recommendation to the industry	5-84
5.3.2	Recommendation for further research	5-87
5.4	CLOSURE	5-88

SYNOPSIS

Chapter 5 aims to provide some recommendations to the software selection process when considering commercial software and open source alternatives. Software selection should be based on the value contributions that a software product brings to the organisation. Furthermore, the complete software life cycle should be used when investigating alternatives, starting at the organisational value chain and ending with the concept of total cost of ownership through which an organisation can assess the total running costs of a software application before making a decision to purchase.

5.1 INTRODUCTION

The free price tag of open source software makes for a very compelling argument when managers need to decide on the next software system to employ within an organisation. Unfortunately there is also a lot of confusing publicity surrounding open source offerings.

Organisations need to identify value-adding contributions throughout the software life cycle that will contribute towards or reduce the demands made on the bottom line.

In this regard managers are faced with the challenge of making fact-based decisions regarding the software systems to employ, whether it is commercial or open source software. Managers need to evaluate the software value contributions throughout the software life cycle and start before the decision to purchase a specific software package is made. Initial considerations should include the area within the organisational value chain in which software would contribute the most value.

Organisations should also investigate the interoperability requirements of its software systems with that of its trading partners. Lack of interoperability can hamper the flow of necessary information between partners and increase inefficiencies. Furthermore, open source and commercial software include inherent risks that need to be evaluated before a decision to purchase is made.

A software purchase decision should also include looking at all the cost aspects involved and not only the initial purchase price. Aspects such as the implementation costs as well as the costs with regard to support and maintenance can be much higher than the price of purchase.

Ultimately the decision to purchase software should be based on the total cost of ownership of the software package as well as expected contributions with regard to the organisational value chain, value delivery network and perceived risks involved.

The aim of the empirical research was to test the respondent's opinions regarding various aspects of value with regard to open source and commercial software as identified in **Chapter 2** and **Chapter 3**, as well as prioritise the aspects identified as part of total cost of ownership.

5.2 INTERPRETATION OF STUDY

5.2.1 Software value using the software life cycle

Chapter 2 started by briefly discussing the hype cycle, a term coined by Gartner to describe the stages through which new technology moves. This served as a starting point in order to understand the current phases of various open source software applications today, as well as to emphasise the need to make software decisions based on value add throughout the software life cycle instead of on "hype".

The software life cycle was developed at the hand of the customer activity cycle. This cycle identifies three phases for any customer activity – *pre, during* and *post.* In each of these phases, value-adding aspects can be identified. These phases can also be applied to the software selection activity, with the organisation being the customer.

Chapter 2 then continued to discuss the *pre* phase and started by looking at the organisational value chain. In this regard software selection should be based on the activities in the value chain which will benefit the most from a software application implementation. Furthermore, the value delivery network was discussed and aspects of standard implementation become important in order to facilitate the communication between an organisation and its trading partners.

Lastly the issue of risk was discussed. Open source and commercial software each have inherent associated risks that need to be analysed before the decision to purchase is made.

5.2.2 Total cost of ownership of software

Chapter 3 addressed the *during* and *post* phases as identified by the customer activity cycle. In this regard it used the concept of **total cost of ownership** to identify aspects that can either contribute towards or detract from the software value-add.

The first aspect that was considered was the *purchase price* of the software. In this regard commercial software offers various payment mechanisms to make the purchase less "painful". Organisations also need to consider the various licensing options inherent in the use of open source - certain licensing types require an organisation to make all derivative software available under the same license as the original open source product. In other words, if an organisation makes use of the software and modifies it to suit their needs, the source modifications must also be made available under the same licensing conditions as the original product – something an organisation might be against in order to guard over their "competitive edge".

Secondly the *cost of implementation* was discussed, where some opinions were that the implementation phase could cost as much as four times the original purchase price. Organisations making use of a structured change management plan can reduce this cost. Furthermore, making use of consulting firms that specialise in software implementations can also reduce this cost.

The *cost of migration* was analysed next. In this regard it was found that capable open source business applications are not always readily available for commercial counterparts, but that various software applications in the software infrastructure arena are quite stable and mature, with the majority of current web servers running on an open source product. Furthermore it was found that migration from a commercial UNIX based environment to a Linux based environment was much less involved than from a Windows based environment.

Fourthly, the aspect of the *cost of support and maintenance* was investigated. Research indicated that support and maintenance was a major concern for many IT managers with regard to open source software. The cost of support for open source products can also be more expensive than for commercial software. However, as the in-house open source skills of an organisation increase, the cost of support and maintenance should decrease.

Lastly the aspect of the *cost of training* was discussed. The level of proficiency in a software application, and therefore the efficiency and effectiveness in the organisation, can be linked to the training activity. In this regard certain training institutions are endeavouring to provide training in open source products as has been provided for commercial software. Furthermore, some governments are also embarking on open source training programmes.

5.2.3 Factors identified by the empirical research

The empirical research, as discussed in **Chapter 5**, aims to provide some ideas on the current perceptions of individuals towards commercial software and open source alternatives based on their organisational and technical roles within an organisation.

Secondly, the research looked at respondents' views on the software environment in which open source software currently contribute the most value. Thereafter respondents' were asked to identify the most prominent risk they would associate with open source as well as whether they are of the opinion that commercial software or open software offer the best level of interoperability currently.

The research also aimed to shed some light on current pros and cons associated with open source, as well as the importance attached to each of the aspects of total cost of ownership of software. Finally respondents were asked to list their current software application usage. This acted as a control question and gauged the average respondent's current software environment.

5.3 RECOMMENDATIONS

The findings of the study form the basis for the following recommendations to the industry and for further research.

5.3.1 Recommendations to the industry

Open source are increasingly offering alternative choices for various software applications commercially available. However, decision-making on the most appropriate software package between open source and commercial software alternatives must not be based on market build-up or the latest buzzwords. Rather it must be fact-based and consider aspects which either add or detract from value throughout the software life cycle. By using the customer activity cycle as a guideline for life cycle activities, an organisation can start by addressing issues in the *pre* phase.

Firstly, organisations should start by addressing the question of which software application, whether commercial or open source, will contribute the most value within the organisational *value chain*. This decision must be based on evaluating efficiencies such as time-savings, cost-savings or the reduction of human error within the value chain in order to identify which value chain activity would benefit most from the introduction of a software system.

After the value chain activity has been identified, available software systems can be researched. These systems might be required to interact with trading partner's systems throughout the *value delivery network* and it is therefore important to understand the level of interoperability possible with each identified software application. Furthermore, the implementation of industry standards, as applicable to the software systems in question, provides a level of protection for the software investment as it caters for future interoperability requirements.

Aspects that detract from software value should also be considered before evaluating the purchase price. In this regard there are various *risks* inherent in commercial and open source software applications which should be identified and addressed.

After these issues are dealt with, activities in the *during* and *post* phases should be investigated with regard to open source and commercial software applications. The first consideration would be the *cost of software purchase*. With regard to commercial software applications, an organisation can negotiate various payment options apart from the established once off purchase payment. With regard to open source applications, an organisation needs to be aware of the limitations or imposed restrictions of the licensing type under which the open source software has been released.

The next issue to consider is the *cost of software implementation*. A structured change management plan can assist in reducing the "implementation stress" as well as increasing the probability of a successful implementation. If an organisation does not have the necessary in-house skills for the software application, the use of consulting firms in that specific software environment can add the required credibility as well as implementation experience. Organisations can also increase the in-house skills by working closely together with these firms.

If software needs to be migrated to a new software system, the *cost of migration* should be considered. Organisations will do well to understand that a move to open source software from a commercial Windows based environment is more difficult than moving from a commercial UNIX based environment to a Linux based environment. Although open source software offerings are already quite broad with various alternative applications to commercial offerings, the user interface environment can be very different and may require additional training.

The *cost of support and maintenance* should also form part of the value assessment. Open source support and maintenance can be more expensive than corresponding commercial software support and maintenance, based on the support provider. If an organisation is not used to the software application, for example moving to an open source alternative, organisations should be aware of potential increased support requirements and should budget correspondingly.

Lastly the *cost of training* should also be part of the value-evaluation. Commercial software training are currently more mature than the open source counterparts and the level of training and costs involved should be assessed for each of the identified software packages. The advantages of having existing in-house expertise on a range of software products should also be considered when evaluating new alternatives. Finally it can be advantageous to identify training centres for the software packages in question that are geographically nearby to an organisation.

All the abovementioned aspects should form part of a complete value assessment for a software product. Only when these issues are considered is a fact-based decision possible and the decision to make use of an open source alternative can be put in context with the corresponding value that the product brings to the table.

5.3.2 Recommendations for further research

The research design employed within this paper is exploratory. The result of this is that various other research possibilities have emerged:

- Research on the quantifiable aspects that can be identified when investigating the organisational value chain as well as the value delivery network.
- In depth research on the various risks inherent in the use of open source compared to commercial software.

- An industry analysis on the use of open source software based on the software environment – infrastructure, desktop or business software.
- Research on quantifiable measures when considering aspects of total cost of ownership of software as identified in this document.
- Research on providing a value-assessment framework for value-based software selection for an organisation.

These research recommendations will provide decision-makers with a more in-depth, quantifiable approach to value decision making as it pertains to software systems.

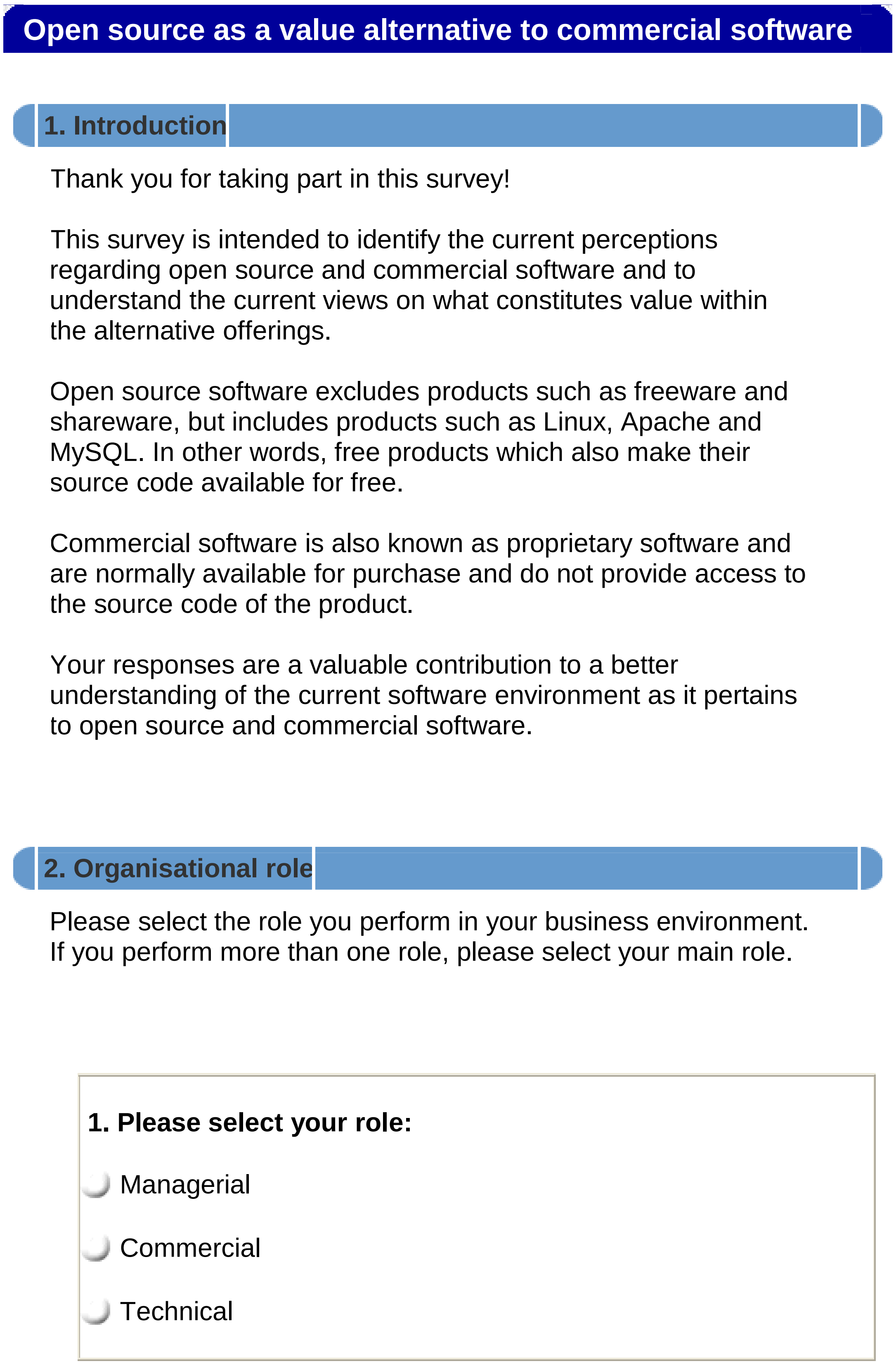
5.4 CLOSURE

The research aims to identify issues that need to be considered when evaluating open source alternatives to commercial software. It is important to look at the software purchase decision from a software life cycle perspective in order to understand the value contributions a software product would make compared to the cost of the software throughout its lifetime. Ultimately, open source software simply offers an organisation a new alternative that should still be evaluated on the value it would contribute to the organisation:

"Value should be the deciding factor."

APPENDIX A

Online survey made available to respondents



Open source as a value alternative to commercial software

1. Introduction

Thank you for taking part in this survey!

This survey is intended to identify the current perceptions regarding open source and commercial software and to understand the current views on what constitutes value within the alternative offerings.

Open source software excludes products such as freeware and shareware, but includes products such as Linux, Apache and MySQL. In other words, free products which also make their source code available for free.

Commercial software is also known as proprietary software and are normally available for purchase and do not provide access to the source code of the product.

Your responses are a valuable contribution to a better understanding of the current software environment as it pertains to open source and commercial software.

2. Organisational role

Please select the role you perform in your business environment. If you perform more than one role, please select your main role.

1. Please select your role:

○ Managerial

○ Commercial

○ Technical

○ Administrative

3. Category of software involvement

Please select the category that mainly applies to you:

User - An individual spending most of his/her time actually using a set of products, who is not mainly concerned about the installation or development of these products.

Implementer - An individual spending most of his/her time implementing and maintaining a set of products for other users.

Developer - An individual spending most of his/her time developing (programming) software for others to make use of.

2. Are you mostly a USER, IMPLEMENTER or DEVELOPER of software applications?

○ User

○ Implementer

○ Developer

4. Open source usage

In your opinion, in which software environment does open source currently contribute the most value. Please only select the most important option. If the option is not listed, type in your response in the "other" category.

3. Please select the environment in which open source currently contributes the most value, in your opinion:

- ○ Infrastructure/Server software
- ○ Desktop/Client software
- ○ Business software
- ○ Other (please specify)

5. Interoperability

Interoperability refers to the implementation of standards within a specific software environment in order to share data and interact with other software applications.

With regard to the implementation of standards and interoperability, does open source software or commercial software currently lead the way?

Please select the level to which commercial software and open source implement standards, in your opinion.

4. Please select the level of interoperability for each:

	poor	average	good	very good	excellent
Open source software	○	○	○	○	○
Commercial software	○	○	○	○	○

6. Risks

Various risks are identified below with regard to the current use of open source software.

Please select the major risk you would currently associate with the use of open source software. If a particular risk is not listed, type in your response in the "other" category.

5. Please select the most prominent risk currently inherent in the use of open source software, in your opinion:

○ Lack of support and maintenance

○ Lack of training

○ Lack of open source skills

○ Lack of application offerings

○ Other (please specify)

7. Software life cycle considerations

The next two questions concerns the various activities that can form part of the total cost of ownership of using a software product.

Five activities are listed:

The purchase activity - This activity encompasses the process of deciding which product to purchase, including the actual purchase process.

The implementation activity - This activity involves the implementation/installation of a software product.

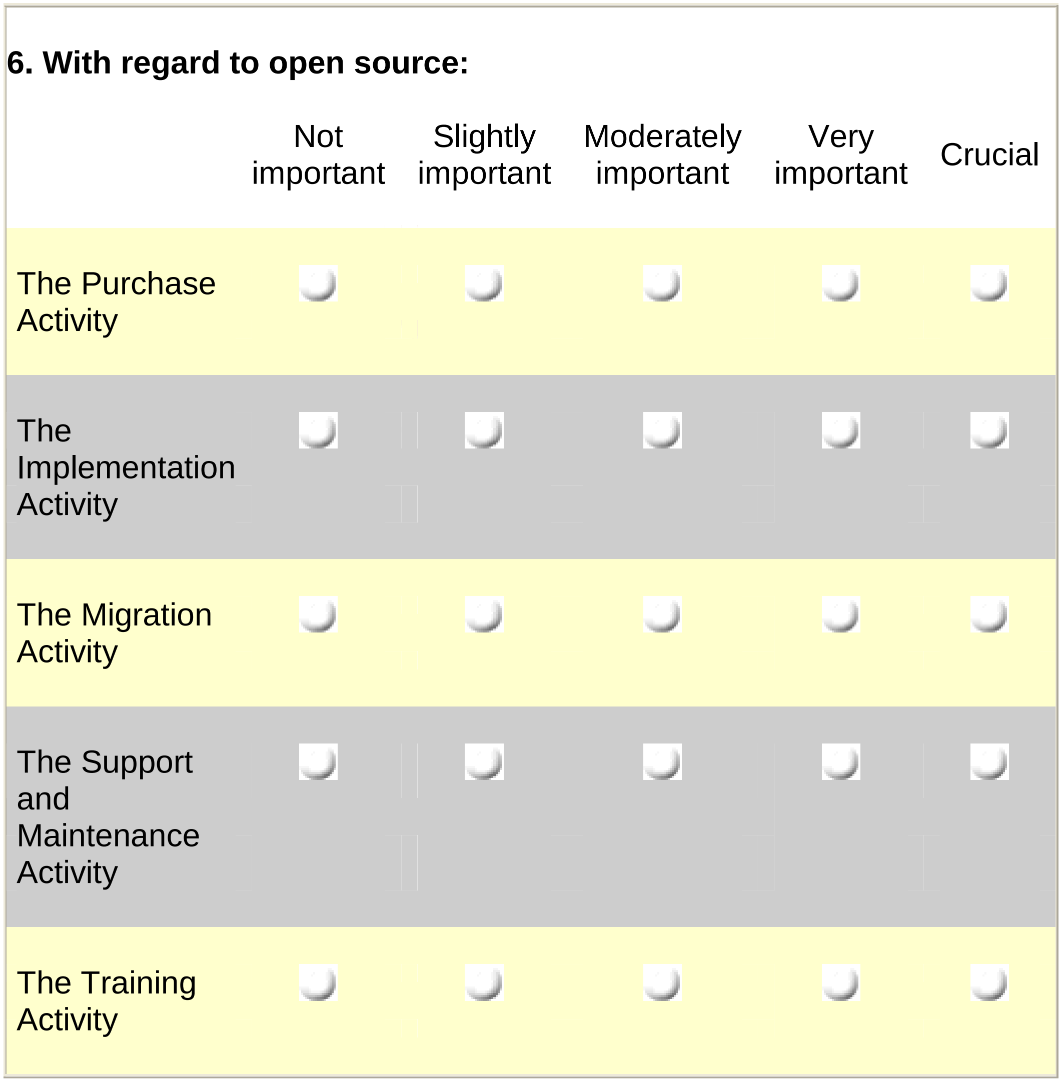
The migration activity - All the required actions to migrate from an existing install base to a new software product.

The support and maintenance activity - Tasks required to ensure that the software product is up and running and to ensure proper maintenance of the product.

The training activity - Activities required to ensure that users and maintainers of the product are skilled in order to be both efficient and effective on the product.

Please rate the level of importance of each activity in the software life cycle as it pertains to open source and commercial software.

The first question is regarding the importance of each activity within an open source software product, and the second question regarding commercial software.



6. With regard to open source:

	Not important	Slightly important	Moderately important	Very important	Crucial
The Purchase Activity	○	○	○	○	○
The Implementation Activity	○	○	○	○	○
The Migration Activity	○	○	○	○	○
The Support and Maintenance Activity	○	○	○	○	○
The Training Activity	○	○	○	○	○

7. With regard to commercial software:

	Not important	Slightly important	Moderately important	Very important	Crucial
The Purchase Activity	○	○	○	○	○
The Implementation Activity	○	○	○	○	○
The Migration Activity	○	○	○	○	○
The Support and Maintenance Activity	○	○	○	○	○
The Training Activity	○	○	○	○	○

8. Pros and cons

Please discuss, in your opinion, the pros and cons of open source.

8. In your opinion, what is currently WRONG with open source?

9. In your opinion, what is currently RIGHT with open source?

9. Current product usage

Please provide the names of the applications that you are currently using in the given categories. If you do not know the name, or are not using an application of that type, simply leave the field blank.

10. Please provide the names of the product you are using in

the given categories. Categories can be left blank.

Web browser	
Email client application	
Word processor	
Spread sheet	
Database application	
Operating system	

10. Thanks!

Thank you for completing this survey!

If you have any queries regarding this survey, please contact Frits Kok at fritsk@adroit.co.za.

BIBLIOGRAPHY

APACHE. 2005. HTTP Server Project. [Internet: http://httpd.apache.org , 2005-08-16].

ASAP Software. 2005. ASAP Leasing. [Internet: http://www.asap.com , 2005-08-12].

AVST. 2004. Investing in Unified Communications: Evaluating the Total Cost of Ownership for AVST CallXpress. [Internet: http://whitepaper.networkmagazine.com/cmpnetworkmagazine/search/viewabstract/70858/index.jsp , 2005-06-11].

BEARINGPOINT. 2004. Server Operating System Licensing and Support Cost Comparison. [Internet: http://www.microsoft.com/windowsserversystem/facts/analyses/comparable.mspx , 2005-04-13].

BEST J. 2005. Linux vs. Microsoft TCO debate rages. [Internet: http://news.zdnet.com/2100-9593_22-5794560.html , 2005-07-23].

BRADSHAW T. 2005. IT for free. [Internet: http://www.infoconomy.com/pages/linux-technology/group100524.adp, 2005-09-18].

BRICKLIN D. 1999. Visi-Calc. [Internet: http://www.bricklin.com/default.htm , 2005-07-20].

BROERSMA M. 2002. Sun's StarOffice no longer free. [Internet: http://news.zdnet.com/2100-3513_22-923039.html , 2005-06-10]

CAREY R. 2002. Planning for success: the importance of a sound change management plan. [Internet: http://www.cmswatch.com/Feature/74-Change-Management , 2005-07-20].

CHOU T. 2003. The hidden costs of software. [Internet: http://www.aspnews.com/strategies/asp_basics/article.php/11299_2214031_1 , 2005-05-12].

CIMNET. 2004. Manufacturing Execution System leasing options. [Internet: http://www.cimnetinc.com , 2004-11-20].

CLAPP J. 2001. The EDGE perspective. [Internet: www.mitre.org/news /edge_perspectives/march_01/ep_cots.pdf, 2005-06-13]

COLEMAN A. 1998. Lowering the Total Cost of Ownership (TCO) for the Desktop. [Internet: http://www.educause.edu/ir/library/html/cnc9807 /cnc9807.html#_Toc437604864 , 2005-08-01].

COMPUTER HISTORY MUSEUM. 2004. Timeline. [Internet: http://www.computerhistory.org/timeline/timeline.php?timeline_category=cm ptr , 2005-05-21].

COMPUTER WORLD. 2001. Open source databases bloom. [Internet: http://www.computerworld.com/cwi/story/ 0,1199,NAV47_STO63629,00.html , 2005-07-03].

CONRAD T. 2005. PostgreSQL vs. MySQL vs. Commercial Databases: It's all about what you need [Internet: http://www.devx.com/dbzone/Article/20743 ,2005-7-20]

COOPER DR & EMORY CW. 1995. Business research methods. 5th edition. Chicago: McGraw-Hill.

COYLE J, BARDI E & LANGLEY C. 2003. The Management of Business Logistics. 7th edition. Stamford: Thomson South-Western.

CRM SYSTEMS. 2005. CRM Software Comparisons. [Internet: http://www.crm-software-comparisons.com , 2005-08-12].

DEGIGLIO M. 2004. The software quality debate rages on [Internet: http://www2.cio.com/analyst/report2252.html ,2005-08-10]

DIDIO L. 2004. Indemnification is a "must-have" for Brown University. Boston: Yankee Group.

EBXML.ORG. 2005. Enabling a global electronic marketplace. [Internet: http://www.ebxml.org , 2005-07-12].

EXCEL. 2003. Microsoft Excel 2003 (Software application used to analyse the research data). Redmond:Microsoft.

GARTNER. 2003. The hype-cycle for open source technologies, 2003. [Internet: https://mailman2.grnet.gr/pipermail/open-source/attachments/20030803/81dfee65/Hype_Cycle_for_Open-Source.pdf , 2005-08-10].

GIERA J. 2004. The costs and risks of open source – debunking the myths. [Internet: http://www.microsoft.com/windowsserversystem/facts/analyses/opencost.mspx , 2005-04-21].

GONZALEZ-BARAHONA J. 2000. A brief history of open source. [Internet: http://eu.conecta.it/paper/ brief_history_open_source.html, 2004-10-04].

GRAEME S & BOARIU A. 2004. Microsoft Augments Intellectual Property Indemnification. Volume 1. IDC #32467. IDC.

GTK WIMP. 2005. Open source Windows impersonation. [Internet: http://gtk-wimp.sourceforge.net/ , 2005-08-28].

IDABC. 2005. German Bundestag completed Linux migration. [Internet: http://europa.eu.int/idabc/en/document/4596 , 2005-09-10].

INGRAM MICRO. 2005. Interested in financing your Microsoft software purchases? [Internet: http://www.microsoft.com/canada/licensing/softwarefinancing.mspx , 2005-04-21].

KENWOOD C. 2001. A business case study of open source software. [Internet: http://www.mitre.org/work/tech_papers/tech_papers_01/kenwood_software/kenwood_software.pdf , 2005-04-16].

KOTLER P. 2003. Marketing management. 11th edition. New Jersey: Prentice Hall.

LINTRAINING. 2005. Linux Professional Institute Certification. [Internet: http://www.lintraining.com , 2005-08-11].

LINUX.ORG. 2005. The Linux Home Page. [Internet: http://www.linux.org/ , 2005-05-10].

MCBRYDE J. 2004. Culture of innovation follow up: value and promise. [Internet: http://www.dmem.strath.ac.uk/compscot/PastEvents/Sep04INN/valueWorkshop15oct.ppt , 2005-08-10].

MOUTON J & MARAIS HC. 1996. Basic concepts in the methodology of the social sciences. 5th impression. Pretoria: HSRC.

MUNDIE C. 2001. Commercial software, sustainable innovation. [Internet: http://news.com.com/2010-1071-281466.html?legacy=cnet, 2005-04-06].

MYSQL. 2005. MYSQL Database. [Internet: http://www.mysql.com , 2005-05-22].

NETC.ORG. 2005. Glossary. [Internet: http://www.netc.org/openoptions/ appendices/glossary.html , 2005-08-10].

NOVELL. 2005. Novell Consulting: Linux & Open Source. [Internet: www.novell.com/consulting/expertise/open_source.html , 2005-08-10].

ODELLION RESEARCH. 2005. Total cost of ownership. [Internet: www.odellion.com/pages/financial%2520models/TCO/financialmodels_tco_de finition.htm , 2005-08-02].

OLD COMPUTERS. 2005. IBM PC. [Internet: http://www.old-computers.com/museum/computer.asp?c=274 , 2005-06-11].

OPENOPTIONS. 2005. Total cost of ownership. [Internet: http://www.netc.org/openoptions/pros_cons/tco.html , 2005-09-19].

OPENXOURCE. 2005. Services and Support Business Model. [Internet: http://www.openxource.com/crossings/2005/01/12/services_and_support _business_model/brief , 2005-04-12].

PORTER M. 1985. Competitive Advantage. Creating and Sustaining Superior Performance. New York: Free Press.

POSTGRESQL. 2005. PostgreSQL Database. [Internet: http://www.postgresql.org , 2005-04-23].

POSTNOTE. 2005. Open source software. [Internet: www.parliament.uk/parliamentary_offices/post/pubs2005.cfm , 2005-07-12].

REDHAT. 2005. The Red Hat Linux Distribution. [Internet: http://www.redhat.com , 2005-05-15].

RFG. 2005. Business advisors to IT executives. [Internet: http://www.rfgonline.com , 2005-06-10].

ROSEN L. 2001. Which open source license should I use for my software? [Internet: http://www.rosenlaw.com/html/GL5.pdf , 2005-7-28]

SCHADLER T. 2003. Your open source strategy. September. Wholeview TechStrategy Research. Boston: Forrester Research Inc.

SHUTTLEWORTH FOUNDATION. 2005. Learning Linux Material. [Internet: http://www.shuttleworthfoundation.org/index.php?option=content&task=view&id=285&Itemid=41 , 2005-08-02].

SILVER M. 2005. Management update: Enterprisewide open-source office adoption will be difficult. [Internet: http://mediaproducts.gartner.com/gc/webletter/microsoft4_enterprise/2005/article9/article9.html , 2005-08-12].

SMITH W. 2004. Asian Open Source Centre. [Internet: http://www.asiaosc.org/article_202.html , 2005-06-28].

SOURCEFORGE. 2005a. License usage summary. [Internet: http://www.jboss.com/company/licensing , 2005-06-19].

SOURCEFORGE. 2005b. Development status. [Internet: http://sourceforge.net/ softwaremap/ trove_list.php?form_cat=6 , 2005-08-19].

STAROFFICE. 2005. Sun Microsystems' StarOffice. [Internet: http://www.sun.com/staroffice , 2005-03-22].

SUSE. 2005. The SUSE Linux Distribution. [Internet: http://www.suse.com , 2005-06-02].

TAUB A. 2001. The EDGE perspective. [Internet: www.mitre.org/news/ edge_perspectives/march_01/ep_cots.pdf, 2005-06-13]

THE FREE ONLINE DICTIONARY. 2005. Value definition. [Internet: http://www.thefreedictionary.com/value , 2005-07-12].

UK OFFICE OF GOVERNMENT COMMERCE. 2004. Open Source Software Trials in Government. Final Report. [Internet: http://www.ogc.gov.uk/ embedded_object.asp?docid=1003914 , 2005-08-11].

UK OFFICE OF GOVERNMENT COMMERCE. 2002. Guidance on implementing UK Government Policy on Open Source Software. July Report.

VANDERMERWE S. 1999. Customer capitalism: increasing returns in new market spaces. London & New York: Nicholas Brealy.

WARRENE B. 2005. Navigating open source licensing. [Internet: http://www.sitepoint.com/article/open-source-licensing , 2005-08-01]

WORDREFERENCE. 2005. Definition of indemnification. [Internet: http://www.wordreference.com/definition/indemnification , 2005-08-12]

ZIKMUND W. 2003. Business Research Methods. 7th edition. Stamford: Thomson South-Western.