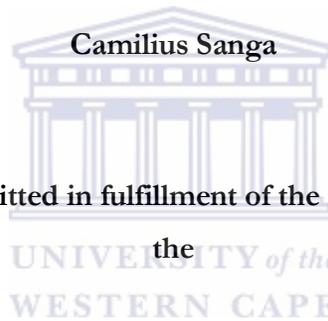


**A TECHNIQUE FOR THE EVALUATION OF FREE
AND OPEN SOURCE E-LEARNING SYSTEMS**

By



**A Thesis submitted in fulfillment of the requirements for
the**

Degree of

Doctor of Philosophy in Computer Science

University of the Western Cape

2010

Date: November 20, 2010

UNIVERSITY OF THE WESTERN CAPE

Abstract

**A TECHNIQUE FOR THE EVALUATION OF FREE
AND OPEN SOURCE E-LEARNING SYSTEMS**

By Camilius Sanga

Chairperson of the Supervisory Committee: Professor I. M. Venter

Department of Computer Science

Evaluating software is a universal and complex problem. The question is: how should software be selected and adopted, or rather, which of the software packages is the most suitable for a specific environment? Extensive research on the evaluation of software has been done, but only a few researchers have considered evaluation of e-learning systems based on three software quality characteristics (i.e. usability, maintainability and deployability) for implementation in third world countries. In this thesis, it will be considered how to use a mixed research methods for the evaluation of free and open source e-learning systems in a developing country. The scope of this investigation is the evaluation of two free and open source e-learning systems at the Open University of Tanzania using 33 stakeholders (some with more and others with less computer expertise).

Both quantitative and qualitative research methods were used in the case study. The qualitative methods comprised questionnaires and interviews (with the stakeholders) whereas the group fuzzy analytical hierarchy process, a multiple-criteria decision-making algorithm, represented the quantitative method. In particular, a quantitative method called Fuzzy Free Open Source Software Evaluation Technique (FOSSET) is proposed. This technique combines fuzzy approximation reasoning with the classical analytic hierarchy process algorithm. The proposed technique was validated in the context of the evaluation of free and open source e-learning system at the Open University of Tanzania, a distance learning university.

The outcome shows that the technique is efficient and effective. It reduces the number of pair wise comparison required and is therefore useful even when a large number of attributes are to be compared. Stakeholders participated in the evaluation process and in choosing the preferred software product. This technique focused on the system's usability, maintainability

and deployability. The technique identified the quality attributes, which might have problems when the system is deployed in a specific environment. This boosted user satisfaction and acceptance of the system.

Some of the limitation of the proposed technique is that it requires more computations in finding the composite priorities than conventional AHP.

Keywords: *multi-criteria evaluation tool, Free and Open Source Software, e-learning systems, software quality, Analytical hierarchy process (AHP), Fuzzy AHP*



TABLE OF CONTENTS

Abstract	i
Table of Contents	iii
List of Figures	viii
List of Tables	ix
Declaration	xi
Acknowledgments	xii
Glossary	xiii
<i>Chapter 1</i>	<i>1</i>
Statement and analysis of the problem	1
Sketching the background	1
Motivation for the research	4
The research problem	6
Research approach	8
Conclusion	9
<i>Chapter 2</i>	<i>10</i>
Literature review	10
Introduction	10
Free and Open Source E-learning systems in developing countries	10
What is an e-learning system?	10
What is free and open source software?	11
How is a developing country defined?	12
What is software quality and how is it determined?	13
Quality	13
Software quality	13
FOSS in Tanzania	14
What e-learning systems are currently being used in Tanzania?	14
Problems with selection and evaluation of FOSS in Tanzania	15
Research questions explained	16
How are the characteristics and attributes that determine software quality, identified?	16
How attributes are measured or quantified?	19

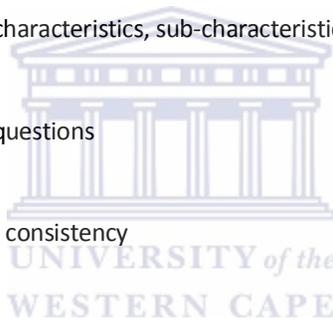


How can the specific software quality characteristics of interest to this study be quantified?	25
Framework formulation for the evaluation of the quality of software	32
Methods for the evaluation of software	33
The advantages and disadvantages of the frameworks for the evaluation of software	34
Can Fuzzy AHP address the subjectivity of judgement?	37
Conclusion	41
<i>Chapter 3</i>	42
Research design and methodology	42
Introduction	42
Research Approach	42
Epistemology	44
Theoretical perspective	44
Methodology	45
Survey	46
Ethnography	46
SSM – Soft Systems Methodology	46
Methods	50
Why combine research methods?	50
Actual study	54
Instrument design	54
Pilot testing	55
Data collection	55
Instruments used to collect data	55
Data preparation	56
How data was analysed	56
First cycle of SSM: Replication experiment	58
Second cycle of SSM: Implementation of AHP	58
Third cycle of SSM: Implementation of Fuzzy AHP (with extent analysis)	58
Fourth cycle of SSM: Implementation of Group Fuzzy AHP	59
Further cycles: Proposed future research	59
Limitation of the study's methods and methodologies	60
Conclusion	61
<i>Chapter 4</i>	62
Research results	62
Introduction	62
The problem revisited	62
First cycle of SSM: Replication experiment	62



Outcome of the QWS replication in the first cycle of SSM	64
Validation of the results	64
Second cycle of SSM: Implementation of AHP	65
Data modeling and analysis	68
Outcome of AHP: the second cycle of SSM	68
Third cycle of SSM: Implementation of Fuzzy AHP (with extent analysis)	69
Algorithm for the evaluation of software when judgement is not well defined	69
Fuzzy AHP	70
Outcome of the Fuzzy AHP (with extent analysis) from the third cycle of SSM	74
Fourth cycle of SSM: Group Fuzzy AHP	75
Outcome of the combined Group Fuzzy AHP	80
Comparison of the three algorithms outcomes	81
Conclusion	83
<i>Chapter 5</i>	<i>84</i>
The interpretation of the results	84
Introduction	84
Findings of the replication experiment –testing the feasibility of AHP	84
Findings of the implementation of AHP	85
Findings of the implementation of Fuzzy AHP (with extent analysis) and Group Fuzzy AHP	87
Research questions revisited	88
Motivation for using mixed research methods	93
Conclusion	94
<i>Chapter 6</i>	<i>95</i>
Discussion and concluding remarks	95
Introduction	95
Why this thesis is a contribution	95
What is knowledge?	95
What theories were considered?	96
What counts as knowledge?	97
Contextualising knowledge and its contribution in Computer Science	98
Assessment of the knowledge contribution in theory	101
Limitation of the research study	108
Suggesting directions for future research	108
Concluding remarks	109
Bibliography	110
Appendices	122

Appendix A	123
AHP	123
Appendix B	131
Evaluation results of e-learning systems	131
Replication experiment	132
The experiment explained	132
Appendix C	136
Ethical consideration	136
Consent Form	137
Appendix D	138
Proposed software quality attributes	138
Appendix E	140
Explanation of: software quality characteristics, sub-characteristics and attributes	140
Appendix F	143
Questionnaires and open-ended questions	143
Appendix G	157
Stepwise procedures for checking consistency	157
Appendix H	159
Fuzzy ahp and proposed algorithm development	159
a. Fuzzy problem structuring	160
b. Deriving priorities from Fuzzy Comparison matrices	160
c. Computing Fuzzy Prioritization	164
d. Computing global priorities	164
Proposed algorithm	165
Why propose a new algorithm?	166
Appendix I	168
Group Fuzzy AHP	168
Steps of the proposed Fuzzy AHP (with extent analysis) algorithm	169
Appendix J	177
The checking of consistency ssm cycle 1.	177
Appendix K	178
Application of AHP to data collected from OUT	178
Appendix L	181



Principles of evaluating interpretive field studies	181
Appendix M	183
Priority weight obtained after using Fuzzy AHP (with extent analysis) and Group Fuzzy AHP	183
Group Fuzzy AHP	186
Appendix N	196
CASE STUDY SUMMARY	196
Index	206



LIST OF FIGURES

<i>Number</i>	<i>Page</i>
FIGURE 1: CURRENT AND ENVISAGED AFRICAN UNDERSEA CABLES	5
FIGURE 2: RELATIONSHIP OF PROCESS QUALITY, PRODUCT QUALITY AND QUALITY-IN-USE.....	20
FIGURE 3: HIGH LEVEL MEASUREMENT CONTEXT.....	25
FIGURE 4: FRAMEWORK.....	40
FIGURE 5: FOUR ELEMENTS OF THE RESEARCH PROCESS	43
FIGURE 6: THE FOUR ELEMENTS OF RESEARCH AS WAS USED IN THIS THESIS	43
FIGURE 7: BASIC SHAPE OF SSM.....	48
FIGURE 8: ATTRIBUTES MEASUREMENT.....	55
FIGURE 9: HOW THE SSM WAS USED TO MANAGE THE RESEARCH.....	57
FIGURE 10: THE FIRST CYCLE OF SSM	63
FIGURE 11: FLOWCHART OF AHP.....	66
FIGURE 12: THE SECOND CYCLE OF SSM.....	67
FIGURE 13: HIERARCHICAL STRUCTURE OF USABILITY CHARACTERISTIC.....	70
FIGURE 14: FLOWCHART OF THE FUZZY AHP (WITH EXTENT ANALYSIS) AND GROUP FUZZY AHP	72
FIGURE 15: THE THIRD CYCLE OF SSM.....	73
FIGURE 16: FLOWCHART OF THE GROUP FUZZY AHP.....	76
FIGURE 17: THE FOURTH CYCLE OF SSM.....	77
FIGURE 18: PROPOSED DATABASE RELATIONSHIP	90
FIGURE 19: CONCEPTUAL FRAMEWORK.....	91
FIGURE 20: FINAL FRAMEWORK	92
FIGURE 21: INTERPRETATIONS OF KNOWLEDGE IN COMPUTER SCIENCE.....	98
FIGURE 22: DIVERSITY DISCIPLINES IN COMPUTER SCIENCE	99
FIGURE 23: SIMPLE HIERARCHICAL STRUCTURE.....	125
FIGURE 24: FUZZY TRIANGULAR MEMBERSHIP FUNCTION.....	161
FIGURE 25: MEMBERSHIP FUNCTION PLOT FOR TABLE 28	162
FIGURE 26: MEMBERSHIP FUNCTION PLOT FOR TABLE 29	163
FIGURE 27: FUZZY AHP.....	165
FIGURE 28: FUZZY AHP (WITH EXTENT ANALYSIS) AND GROUP FUZZY AHP	166
FIGURE 29: GROUP FUZZY AHP.....	168
FIGURE 30: MEMBERSHIP FUNCTION PLOT FOR TABLE 33	170

LIST OF TABLES

<i>Number</i>	<i>Page</i>
TABLE 1:	ADVANTAGES AND DISADVANTAGES OF EXISTING FRAMEWORK FOR EVALUATION OF SOFTWARE... 34
TABLE 2:	ADVANTAGES AND DISADVANTAGES OF EXISTING FRAMEWORK FOR EVALUATION OF SOFTWARE... 35
TABLE 3:	ADVANTAGES AND DISADVANTAGES OF EXISTING FRAMEWORK FOR EVALUATION OF SOFTWARE... 36
TABLE 4:	THE CATWOE MNEMONIC (CHECKLAND & SCHOLES, 1990) 49
TABLE 5:	THE CATWOE MNEMONIC FOR THE RESEARCH STUDY..... 49
TABLE 6:	PROS AND CONS OF PROPOSED RESEARCH METHODS..... 52
TABLE 7:	OUTCOME FROM FIRST CYCLE OF SSM..... 64
TABLE 8:	OUTCOME FROM THE SECOND CYCLE OF SSM 68
TABLE 9:	OUTCOME FROM FUZZY AHP (WITH EXTENT ANALYSIS)..... 74
TABLE 10:	OUTCOME WHEN USING GROUP FUZZY AHP FOR THE LOWER ELEMENTS 78
TABLE 11:	OUTCOME WHEN USING GROUP FUZZY AHP FOR THE MIDDLE ELEMENTS 79
TABLE 12:	OUTCOME WHEN USING GROUP FUZZY AHP FOR THE UPPER ELEMENTS 80
TABLE 13:	OUTCOME OF THE COMBINED GROUP FUZZY AHP 81
TABLE 14:	COMPARISON OF OUTCOMES..... 82
TABLE 15:	TAXONOMY OF THEORY - ADAPTED FROM (GREGOR, 2006)..... 96
TABLE 16:	DIVERSITY DISCIPLINES IN COMPUTER SCIENCE 99
TABLE 17:	BODY OF KNOWLEDGE FOR SOFTWARE QUALITY MEASUREMENT 101
TABLE 18:	SAATY SCALE 126
TABLE 19:	QWS DATASETS..... 131
TABLE 20:	SYMBOLS WITH THEIR GIVEN WEIGHTS..... 132
TABLE 21:	PAIRWISE COMPARISON MATRIX FOR ADAPTATION CATEGORY..... 133
TABLE 22:	NORMALIZED MATRIX FOR ADAPTATION CATEGORY 134
TABLE 23:	PRIORITY VECTOR FOR EACH OF THE CATEGORIES OF THE EVALAUTED E-LEARNING SYSTEM 134
TABLE 24:	OVERALL EVALUATION RESULTS 135
TABLE 25:	THE HIERARCHICAL STRUCTURE OF THE QUALITY ATTRIBUTES FOR THE PROPOSED EVALUATION ... 139
TABLE 26:	RANDOM INDEX (ADOPTED FROM SAATY, 1980) 158
TABLE 27:	IDENTIFICATION OF MEMBERSHIP FUNCTION 162
TABLE 28:	EXAMPLE OF THE MEMBERSHIP FUNCTION..... 162
TABLE 29:	ANOTHER EXAMPLE OF MEMBERSHIP FUNCTION..... 163
TABLE 30:	MEMBERSHIP FUNCTION PLOT FOR CONVERSION OF CRISP TO FUZZY PAIRWISE COMPARISON..... 164
TABLE 31:	CRISP SCALE 169

TABLE 32:	FUZZY SCALE	169
TABLE 33:	MEMBERSHIP FUNCTION FOR CONVERSION OF CRISP TO FUZZY SCALE	170
TABLE 34:	PRIORITY WEIGHT	180
TABLE 35:	PRINCIPLES FOR CONDUCTING AND EVALUATING INTERPRETIVE FIELDS STUDIES	182
TABLE 36:	RESULTS FROM CONVENTIONAL AHP	185
TABLE 37:	RESULTS FROM GROUP FUZZY AHP - FOR LOWER ELEMENTS	188
TABLE 38:	RESULTS FROM GROUP FUZZY AHP - FOR MIDDLE ELEMENTS	189
TABLE 39:	RESULTS FROM GROUP FUZZY AHP - FOR UPPER ELEMENTS	190
TABLE 40:	RESULTS OF THE COMBINED GROUP FUZZY AHP	191
TABLE 41:	RESULTS FROM FUZZY AHP (WITH EXTENT ANALYSIS)	194
TABLE 42:	COMPARISON OF OUTCOME	195



DECLARATION

I hereby declare that all information in this thesis has been presented in accordance with academic rules. I also declare that, as required by these rules, all the sources I have used or quoted have been indicated and acknowledged by complete references.

Signature *C. Sanga*



ACKNOWLEDGMENTS

Thanks, honour, glory, praise and acknowledgement belong to our Almighty God who made it possible for me to endure all the difficulties during my studies.

I would like to give special thanks to my supervisor, Prof. Isabella. M. Venter for supervision, support, guidance, inspiration, encouragement, challenges and advice. I will always cherish the time spent at UWC with my Prof. I learnt a lot- not only about Computer Science but also about life in general. Thanks also to Prof. D. Keats, Prof. Renette Blignaut, Prof. H. K. O. Nyongesa, Prof. J. Agbinya, Prof. Madsen, Dr. B. Turker and Prof. Blackledge for their support during my study.

In addition, I would like to thank DAAD for financial support.

Moreover, thanks to the Sokoine University of Agriculture who allowed me study leave to pursue my studies.

Furthermore, thanks to the following for their support: my colleagues (who were also doing a PhD), UWC Computer Science department staff members, my co-workers at Sokoine University of Agriculture – Computer Centre and all my friends.

Last but not least, I would like to give thanks to my wonderful wife, Elizabeth M. John, for being a source of encouragement, support and motivation. Also thanks to my family members and my relatives (Lucy, Emma, Christian, Christopher, Patricia and Dafroza) for their prayers. I wish my parents, Aloyce and Stephania, and my son, Emmanuel, could be alive to share my joy.

GLOSSARY

ACM: Association of Computing Machinery

AHP: Analytic Hierarchy Process

ANP: Analytic Network Process

AVOIR: African Virtual Open Initiatives and Resources

CATWOE: Customers, Actors, Transformation process, Worldview, Owners and Environmental constraints

CD: Compact Disk

CI: Consistency Index

COTS: Commercial of the Shelf Software

CR: Consistency Ratio

CVS: Control Version Systems

Distance education: is the provision of learning and teaching where the instructor (s) and learner(s) are separated mostly by time and space / location.

DVD: Digital Video Disc: is an optical disc storage media format used to store data. Data can be in form of text, graphics, audio and video.

EFL: English as a Foreign Language

E-learning system: Software package that automate learning and teaching processes. The functions which can be automated ranges from: administration, assessment, and also either course or content management and authoring system.

E-learning: (Electronic learning): is an advanced teaching mode using electronic technology with a wide variety of learning strategies and technologies. (http://www.elearningeuropa.info/extras/pdf/mid_term_en.pdf)

ELECTRE: ELimination Et Choice Translation Reality

ESL: English as a Second Language

ESSE: Experts System for Software Evaluation

FOSS: Free and Open Source Software. Software which is made available with its source code and free available to use, copy, modify and redistribution either the original or modified software (http://www.dwheeler.com/oss_fs_why.html)

FOSSET: Fuzzy Free and Open Source Software Evaluation Technique

GDP: Gross Domestic Product

GQM: Goal-Question-Metric

ICT: Information and Communication Technology: It is the term used to describe a range of technologies for gathering, storing, retrieving, processing, analysing and transmitting information. (http://www.smartstate.qld.gov.au/strategy/strategy05_15/glossary.shtm).

ICT4D: ICT for Development

IEC: International Electrotechnical Commission

IEEE: Institute of Electrical and Electronics Engineers

IMF: International Monetary Fund

INCAMI: Information, Need, Concept model, Attribute, Metric and Indicator

ISO: International Standards Organization for Standardization

IT: Information Technology

KEWL.NextGen: Knowledge Environment for Web Based Learning Next Generation. It is an example of Free & Open Source e-learning systems, developed by University of the Western Cape Free Software Innovation Unit in collaboration with other African universities under African Virtual Open Initiatives and Resources (AVOIR).

MCDA: Multi-criteria Decision Aid

MCDM: Multi-criteria Decision-Making

OTSO: Off-The-Shelf-Option

OUT: Open University of Tanzania.

PCM: Pairwise Comparison Matrices

Pedagogy: The field of study that deals with learning and teaching methods including what is taught how teaching occurs and how learning occurs. (www.futurelab.org.uk/download/pdfs/research/lit_reviews/futurelab_review_10.pdf)

PORE: Procurement Oriented Requirements Engineering

QWS: Qualitative Weight and Sum

SABS: South Africa Board of Standards

SAW: Simple Additive Weighting

SSEF: Software System Evaluation Framework

SSM: Soft Systems Methodology

STACE: Socio Technical framework for COTS software Evaluation

Student: Any learner who takes part in either campus (resident) or Open and Distance Learning (ODL) universities.

SUA: Sokoine University of Agriculture

SUE: Systematic Usability Evaluation

SUMI: Software Usability Measurement Inventory

TOPSIS: Technique for Order Preference by Similarity to Ideal Solution

UDSM: University of Dar es Salaam

UWC: University of the Western Cape

WebQEM: Web Quality Evaluation Methodology

WTO: World Trade Organisation



CHAPTER 1

STATEMENT AND ANALYSIS OF THE PROBLEM

Doubt, the essential preliminary of all improvement and discovery, must accompany the stages of man's onward progress. The faculty of doubting and questioning, without which those of comparison and judgment would be useless, is itself a divine prerogative of the reason.
Albert Pike [1809-1891]

Sketching the background

It is often problematic to find suitable software for a specific purpose. To do so, available software applications need to be considered and a method found to determine which application is most suitable for both the purpose of the application and the needs of the user. In developing countries, evaluating software for a specific purpose is in most cases left (by the management of the organisation) to the Information Technology (IT) experts and other stakeholders' views are seldom used. This approach is not always feasible as IT experts are not necessarily available (Tedre et al., 2009). It is therefore important to incorporate and consider the views of stakeholders when adapting new software in a developing country. In general, when referring to a developing country, the understanding is that it is a country with limited infrastructure, poor standards of living, a high level of illiteracy and a low gross domestic product (GDP) (Kunda, 2001). The Information and Communication Technology (ICT) infrastructure of a developing country is thus (in most cases) not comparable to the ICT infrastructure available to developed countries. This is often referred to as the "digital divide". Mehra defines it as:

"the troubling gap between those who use computers and the Internet and those who do not". (Mehra et al., 2004, p. 782)

It is thus necessary to consider ICT solutions within the environment or context in which it will be deployed (Tedre et al., 2009).

In order for developing countries to leapfrog into the knowledge era there must be contextualised development and implementation of quality and low cost ICTs. Will free and open source software (FOSS) provide a low cost solution for the deployment of ICTs in developing countries?

Although FOSS may offer such a solution, it also introduces new challenges for its implementation. Even though FOSS is considered to be “free”, software cost is only part of the cost associated with the implementation of software. Maintaining software, providing the infrastructure and training users is also associated with its implementation cost. Quality of software is often associated with the cost of it, thus users’ acceptance may be compromised as they may assume that FOSS is of lesser quality. Furthermore, if users (stakeholders) are not involved in the development of the software, as is typical in the FOSS environment (Nichols & Twidale, 2003) there may be further problems with the acceptance of it.

According to Wesson (2003), the successful adoption of FOSS systems in a developing country depends on the following factors:

- user acceptance (keeping in mind that users are from varied cultural and educational backgrounds with differing computer expertise).
- availability of resources (since resources are scarce)
- quality of the software

User acceptance encompasses the user’s ability to interact with the software; the knowledge to maintain the software and the resources available to deploy the system in a particular environment.

The quality of software is important when considering which software package to adopt. This is corroborated by Fenton and Neil (2000), Khosravi and Guéhéneuc (2005), and Côté et al. (2007). The quality of software also reduces the cost of software maintenance (Tan & Mookerjee, 2005).

Although several characteristics determine software quality (ISO, 2001), maintainability, deployability and usability are the characteristics that, in the context of a developing country, are the most critical for sustainability. Sustainability of the software implemented in an organisation requires competent IT personnel who have the task of regular maintenance of the system (Pscheidt, 2008). Maintenance of the software comprises many tasks such as: correcting errors/defects, modifying the software system

according to evolving user requirements, as well as servicing, sustaining or improving the software system over time¹.

- *Maintainability* activities depend on the infrastructure available in a particular environment as well as the IT expertise available. The maintainability of software in a developing country is problematic because of the high level of computer illiteracy, shortage of IT personnel, as well as limited infrastructure available.
- Infrastructure plays an important role in providing the enabling environment for unsophisticated software deployment. The *deployability* of software encompasses a large range of activities—from the software development (in the computer laboratory) to the installation in a production environment. The production environment is hampered by low bandwidth and scarce resources in most developing countries.
- Another key desirable criterion for the deployed software is its *usability*. Usability can be defined as the ease at which the user interacts with the software system to accomplish certain goals. Thus, the usability deals with effectiveness (fit for the purpose) and efficiency of the software system (its usefulness). Usability covers a large range of sub-characteristics but in this thesis, only the aspects that deal with the user interface will be explored. In most cases, the user interface of software systems deployed in developing countries will be in a language foreign to the user. The usability of a user interface plays a major role in promoting user satisfaction and user acceptance of software.

The Open University of Tanzania (OUT), a traditional distance learning institution (McHarazo & Olden, 2000), is currently in the process of implementing electronic learning (e-learning) as an initiative to improve learning and teaching at the university. The e-learning initiative of OUT was considered to explore different evaluation frameworks as well as the use of novice IT users in the evaluation of software with uncertain judgement. According to Colace et al., the quality of software can be categorised as either, technological or pedagogical (Colace et al., 2003). E-learning systems were considered as a case study in this thesis, however the focus is on software quality and therefore, the pedagogical aspect of e-learning software will not be considered in this study although its importance is

¹ Reasons for only considering three software quality characteristics – which are deemed the most important for adapting software in a developing country

acknowledged. This has been done to be able to meet the objective of the study as well as specifying the boundary of the research

Motivation for the research

The recent launch of the undersea optical fibre cable (see Figure 1) promises to link Sub-Saharan countries to Europe and Asia and allow wider implementation of e-learning systems in these countries (Cables, 2009). It is anticipated that growth of Internet coverage and usage will be accelerated by the higher bandwidth and low cost of Internet connections the undersea optical fibre will provide. According to Heeks (2008), many developing countries recognise and advocate the importance of ICT for development (ICT4D). He also states that some of these countries have already (i) formulated ICT policies for e-learning; (ii) procured the infrastructure for Internet connectivity and ICT usage, have (iii) rolled out mobile connectivity and (iv) are adopting FOSS.

One motivation of the thesis is to improve the selection of Free and Open Source Software (in particular – Free and Open Source e-learning system) in developing countries. Selection of FOSS according to software quality is endemic problem that is why few organizations adopt (Padayachee et al., 2010). Bruggink (2003) identified the problems which hinder adoption of FOSS in developing countries (especially those countries in Africa) being: (i) limited availability of the Internet makes it difficult to download software even though they are free (ii) lack of technical expertise (iii) top management are reluctant to shift from commercial software (iv) there is perception that FOSS are not user friendly (v) lack of experience, support and information on how best to switch from one type of software to another makes the process to be costly and complex. However, the adoption of FOSS—has costs implications. Besides cost, the FOSS can be evaluated in a number of factors: - maintainability, usability, functionality, reliability, efficiency and portability (ISO, 2001). Choosing the correct application for a given situation is important. This concur to the concluding remark of Bruggink (2003) who calls for a need to develop decision-making tools specifically geared towards the requirements of developing countries.

This prompted the researcher to consider developing a framework, which would assist users with the evaluation of FOSS e-learning systems. Such a framework promises to increase the uptake, adoption and implementation of quality and cost effective FOSS e-learning systems in developing countries.

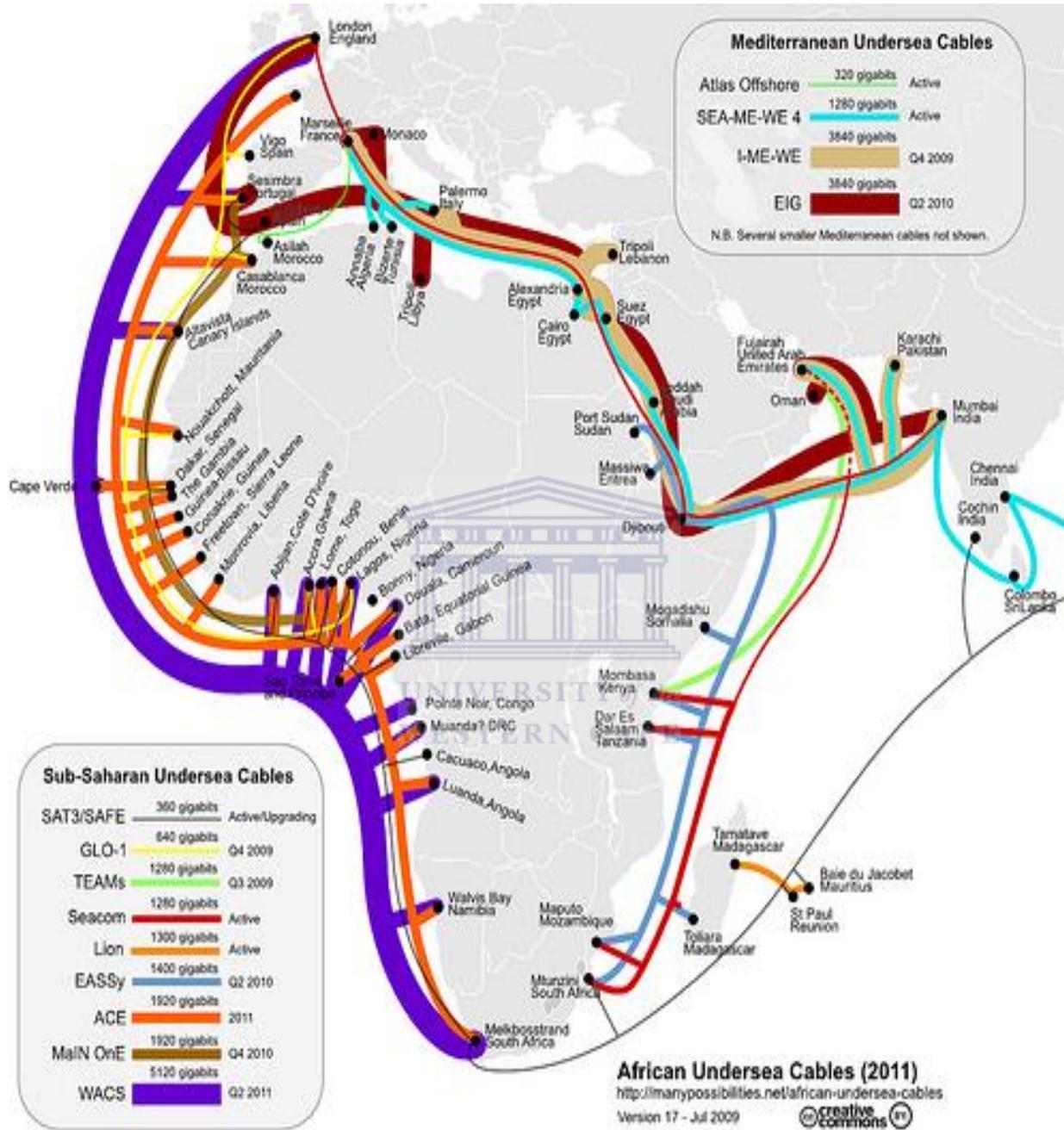


Figure 1: Current and envisaged African Undersea Cables

The research problem

The selection of FOSS is often done by trial and error. Software packages are introduced and if they do not “live up to expectations” another software package is found and introduced until most of the needs of the users are satisfied. This is not an appropriate evaluation method because of its cost, in terms of time and money, and its subjectivity. Furthermore, this method does not counter check “uncertainty” of judgements, that is: judgements which a user makes without the necessary information or without the necessary experience to be able to make such a judgement.

There are known methods for the evaluation of software, for example the qualitative weight and sum (QWS) method (Graf & List, 2005), the analytic hierarchy process (AHP) algorithm (Saaty, 1980), etc. These methods, each with its pros and cons, are employed by software specialists to evaluate software. If only software specialists are used during evaluation, they have the disadvantage that input from the end-users is omitted. When users are used for software evaluation, they may have different skills and experiences, and their environment or contexts of evaluation, may also be different (Bevan, 1999). This has an impact on their ability to judge the quality of the software. The evaluation of software quality is thus generally subjective.

Several researchers have used different methods to evaluate the quality of software according to certain characteristics. Methods used are for example the QWS method, the Goal-Question-Metric (GQM) method and AHP (Finnie et al., 1993; Basili et al., 1994; Vila & Beccue, 1995; Triantaphyllou et al., 1997; Karlsson et al., 1998; Ossadnik & Lange, 1999; Graf & List, 2005; Shee & Wang, 2008). These methods have disadvantages, which include the following, mentioned by Seffah et al. (2001):

- (i) it is difficult to visualize the relationship between characteristics, sub-characteristics and attributes
- (ii) the models are not user-friendly for all stakeholders in the sense that the models do not provide a unified framework for software evaluation by different stakeholders
- (iii) the different software quality models are not well incorporated into the software development cycle.

In addition, they have the disadvantage of not being able to handle the subjectivity of user judgements (Cheng et al., 1999; Wang et al., 2008).

The literature suggests that soft-computing methods should be used to design algorithms to address the subjectivity of user judgement. Some of these methods are fuzzy logic, neurocomputing, evolutionary computing and probabilistic computing (Zadeh , 1994; Chang & Dillon, 2006; Chang et al., 2008). The soft-computing methods (i.e. Bayesian belief network, artificial neural networks and evolutionary algorithms) can be trained to predict the results obtained from other evaluation algorithms, which deal with imprecision, uncertainty, partial truth and approximation during software selection.

The main problem is that user in developing countries find it cumbersome to evaluate software when they are required to make a decision which software to adopt as per specific software quality characteristics, sub – characteristics and attributes. Thus, this study seeks to develop a framework which can assist software users in evaluation of software. The aim of the developed framework is to assist software users in a developing country to select and evaluate appropriate, quality as well as cost effective software. Here the term “appropriate” emphasis the importance of framework being able to improve adoption of FOSS as per Wesson’s three criteria (Wesson, 2003). That is it guides user to select quality software among many software products and the one selected will obvious be accepted by the user and which can be used on available scarce resources. Thus, a framework, which incorporates the opinions of both technical and non-technical users, will be useful and will definitely make a contribution to the larger ICT community. The framework will not only guide the users to use the software but it will also, assist the user to rank the different software products according to software quality characteristics.

According to Chang and Dillon (2006), it is difficult to determine software quality when characteristics are subjective, fuzzy, and not well defined. The subjectivity of evaluators’ judgements further contributes to evaluation uncertainty or fuzziness.

In general, although there are methods to guide users in the evaluation and selection of different software products, many of these methods cannot handle subjectivity (Czogala & Pedrycz, 1981; Chang & Dillon, 2006). This calls for the design of a framework that incorporates algorithms that can manage uncertainty, ambiguity and fuzziness in a decision support model.

The main objective of this thesis is thus to develop a framework for the selection and evaluation of software products in terms of software quality when the evaluators’ judgements are uncertain.

The main objective can be considered in terms of two sub objectives, namely:

- (i) To examine and analyse different algorithms for decision making
- (ii) To design a new or modified algorithm for supporting decision making in the context of a developing country

Thus, in line with these sub objectives, the following research questions can be asked:

- What are the key attributes to consider (in terms of usability, deployability and maintainability) in order to evaluate the e-learning system options for OUT?
- How should the attributes be measured?
- How can the measured attributes be used to create a framework for the evaluation of e-learning systems?

Research approach

The evaluation of two e-learning systems currently being used/implemented at OUT was used as a case study. According to Olivier (2004), a case study is an appropriate method for investigating phenomena when there is a shortage of participants. Even though case studies are usually used when developing theories and mostly in the social sciences (Yin, 1994), a case study approach was considered because this research depends on human evaluation judgement which is prone to subjectivity. According to Yin (2003), a case study design should be considered when: (a) the focus of the study is to answer “how” and “why” questions; (b) you cannot manipulate the behaviour of those involved in the study; (c) you want to cover contextual conditions because you believe they are relevant to the phenomenon under study; and (d) the boundaries are not clear between the phenomenon and context. Case studies have been used before in Computer Science and software engineering to study human centred factors (Runeson & Höst, 2009). For example, in a study to find defects in using user requirements in the design of the software as well as the use of code, Shull found that a case study method (with a mix of qualitative and quantitative approach) is appropriate, effective and useful (Shull, 1998). The research approach of this thesis uses both qualitative and quantitative research methodologies. A combination of methodologies was used to counter a naturalistic or qualitative approach with an objective (or quantitative) approach. It was found that by using an intertwined methodology the results of would be more credible. Credibility comes from the fact that

use of more than one methodology (i.e. triangulation), increases the precision of the research results (Runeson & Höst, 2009). It also allows the results, in certain circumstances, to be generalised. Furthermore, combining methodologies allows the problem to be considered from several viewpoints (Venter, 2000) and thus results in a more holistic approach. Specifically for this study, multiple research methodology was undertaken. Baxter and Jack (2008) strategies for establishing credibility, transferability, dependability, and confirmability for the case study was followed. Baxter and Jack (2008) mentioned the strategies as follows: (a) research question must be clearly written, and the question is substantiated; (b) appropriateness of the research question and purposeful sampling strategies is appropriate for the study and must be applied; (d) data are collected and managed systematically; and (e) the data are analyzed correctly.

Several multi-criteria evaluation algorithms were considered in order to evaluate the feasibility of the proposed framework. The algorithms considered were AHP, Fuzzy AHP (with extent analysis) and Group Fuzzy AHP. The data set used, was collected at OUT in 2007 and 2008. The outcome of the analysis revealed the potential benefits of using all users' judgement in the evaluation process.

Conclusion

This Chapter contextualised the research study. It briefly introduced the background information regarding the problem of evaluating FOSS e-learning systems in a developing country. The rationale for the research was presented and the problem statement, namely, the evaluation of software when judgement is uncertain, was explained. Furthermore, the research objective – how to develop an evaluation framework for a multi-criteria evaluation problem – was outlined. In order to allow a systematic investigation of the study, several research questions were defined.

The rest of the thesis is organised as follows: In Chapter 2 the literature that informs each research question is presented; In Chapter 3 the research design and the methodologies implemented are discussed; In Chapter 4 the results are presented; In Chapter 5 the results are discussed in terms of the research questions asked, and finally in Chapter 6 conclusions are drawn. Unfinished and future works in the research field are briefly mentioned.

CHAPTER 2

LITERATURE REVIEW

The whole structure of science gradually grows, but only as it is built upon a firm foundation of past research.

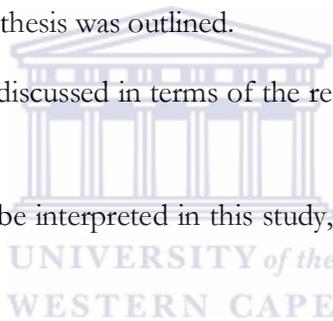
Owen Chamberlain [1920-2006]

Introduction

In the previous chapter, the research study was contextualized. The background information regarding the problem of evaluating FOSS e-learning systems in a developing country, was presented and the rationale for the research approach and the research questions were briefly discussed. Furthermore, the roadmap for the development of the thesis was outlined.

In this chapter, the literature will be discussed in terms of the research questions identified in Chapter 1.

To clarify how certain concepts will be interpreted in this study, some terms of reference will now be explained.



Free and Open Source E-learning systems in developing countries

What is an e-learning system?

E-learning is defined in the literature as an advanced learning mode which uses ICTs (Sambrook, 2003). It integrates various learning approaches and electronic technologies. These technologies include compact disks (CD-ROMs), video disks (DVD), mobile devices, video conferencing, web based learning, satellite-delivered learning and virtual educational networks (Kinuthia & Dagada, 2008). Thus, the definition used in this thesis is that an e-learning system is a system that supports some or all of these kinds of electronic learning. An e-learning system is characterized by being learner-centered, system interactive and allowing asynchronous and synchronous learning and remote teaching (Zhang et al., 2004; Graham, 2006). The quality of the features of an e-learning system determines its usefulness for a specific application and environment. As Zhang et al., points out a lesser quality e-learning system can lead to frustration, confusion and inhibit or hinder learning (Zhang et al., 2004).

What is free and open source software?

FOSS can be defined as software which is made available to all who wish to use it. The software as well as its source code is freely available to use, to copy, to modify and to redistribute. It can be redistributed in its original form or in a modified form or some components of the software can be used (O'Reilly, 1999). In this study, the definition adapted by Rothfuss (2002) will be used: He defined FOSS as software where source codes are freely available for use, modification and redistribution.

To summarize: FOSS grants a user the several types of freedoms. The freedom to: (i) run the program, (ii) study how the program works, (iii) redistribute the software (provided the software is attached to the first terms of license) and (iv) improve the programs for public release. These types of freedom characterize FOSS design, development, implementation and adoption.

FOSS development is characterised by its distributed parallel development methodology where by software developer(s) start a software development project as a hobby or personal interest. They then invite other users to participate or contribute (O'Reilly, 1999). According to O'Reilly (1999), this development methodology has advantages of involving the user as co-developers and this in turn leads to the enhancement of the debugging process. This process can result in a quality product, which is available to the users without having to pay for it (Feller & Fitzgerald, 2000). Even though errors in the software can arise, they are quickly fixed because many software developers are working on the same problem (ibid.).

However, some researchers question the FOSS development process and the quality of its product (Fuggetta, 2003; Samoladas et al., 2004). Manieri et al. (2008) argue that most FOSS applications are poorly documented in terms of requirement specification, system architecture use and system prototypes tested. Several releases of the FOSS products complicate the evaluation in cases where users choose a software product which has not reached maturity. Furthermore, user or stakeholder involvement in FOSS development is scant in developing countries (Kunda, 2001). In the commercial world, closed source software are developed for specific usages (i.e. for particular customers) and the opinion of stakeholders' are valued and incorporated throughout the software development cycle. This is not always the case with FOSS. IT Experts are able to adapt FOSS for their requirements, however it is difficult for non-technical IT users to evaluate and choose an appropriate software

application (from the many that are freely available on the Internet) and adapt it for their specific needs.

The cost of the licenses of closed source software, however is in some case prohibitive for developing countries and thus the fact that FOSS is free, is very attractive to developing countries.

How is a developing country defined?

Kunda (2001) argues that the term “*developing country*” has no universal meaning. The variation in terms of size of country, availability of resources, per capital income of citizens, gross domestic product, standard of living and level of industrial development makes it difficult to classify these countries in a uniform way (ibid.).

The World Trade Organisation (WTO) does not have a definition for a developing country, but rather allows members to classify themselves either as a developed, emerging or developing country (Correa, 2000). The classification given by any country can be questioned by other member countries since it must be based on facts and figures of the country in terms of the economic, industrial and technological capabilities.

The International Monetary Fund classifies the countries as either developed, emerging or developed according to three criteria (IMF, 2009). These are (i) per capital income level (ii) exports capability (iii) the extent at which it is integrated into the global financial system. Most so-called developing countries have small budgets for research and development and thus have economic constraints which results in systems infrastructure deficiencies (Bandalaria, 2007).

Other researchers have used the terms “*third world country*”, “*developing country*”, “*least developed economies*” and “*least developed country*” interchangeably, to define the same term (Ahluwalia et al., 1976).

As can be seen from these definitions a “*developing*” country in general, is a country with high: illiteracy; unemployment; and population growth rate. These mentioned constraints (of the so-called developing countries) affect the design, development, implementation and adoption of ICTs.

What is software quality and how is it determined?

Quality

Quality of the product is the term, which is not tangible. Garvin (1984) outlined five approaches which can be used in defining the quality of the product. These are: (1) the transcendent approach of philosophy; (2) the product-based approach; (3) the user-based approach and (4) the manufacturing-based and (5) value based approaches. Transcendent approach of philosophy view quality is viewed as something which can be measured since the intention of the producer always is to develop an ideal best product. The product-based approach quality is taken as the measure of the internal characteristics of the product in terms of the weight according to the level of importance. The user-based approach defines quality of the product depending on the user satisfaction. The manufacturing-based approach defines quality as the conformance to design requirements identified during the initial stage of the development of the product. Value based approach view quality by relating its value (or degree of excellence) and price of the product. Using value based approach expensive product is considered to have good quality while poor quality is related to cheap product.

The general definition of the quality of a product

is that, it is an abstract concept which can be viewed as the capability of a product to meet/comply to criteria or standards set by an organization or professional board. So for instance in South Africa, the South African Board of Standards (SABS) ensures that “*a product, service or management system meets specified requirements*” (South African Boards of Standards, 2008).

Software quality

In terms of a software product, what is considered to be *quality*? There are two schools of thought for the definition of software quality (Kan, 2003). First, software quality can be defined in terms of a set of characteristics exhibited by software that meet the user’s needs and their expectations. Second, software quality can be defined as the capability of software to conformance to requirements and specifications set during software development. Examples of characteristics that are commonly considered, when evaluating the quality of a software product, are: maintainability, usability, functionality etc. The International Organization for Standardisation (ISO) (ISO, 2001), defines

software quality in terms of maintainability, usability, functionality, reliability, efficiency and portability characteristics.

What *determines* the quality of a software product? According to Fenton and Pfleeger (1997), the quality of software is determined by: (i) the quality of the development process; (ii) the quality of the product, as well as; (iii) the quality of the product-in-use.

The quality of the development process refers to the different phases of the software development process. While the product quality deals with the quality facets exhibited by the software product in the testing environment. On the other hand, the quality of the product-in-use consists of quality features exhibited by the running software in the production environment. This thesis deals with the quality of the product-in-use.

FOSS in Tanzania

Any software application designed for a “*developing country*” need to take into account the prevailing problems of that country. From Kunda’s definition and that of the WTO, as well as that of IMF, Tanzania is listed as a developing country (Kunda, 2001).

FOSS has the potential to address some of the ICT problems that developing countries face in terms of ICT implementation, adoption and use. A framework to assist users in the evaluation and selection of quality software packages could simplify its use in a developing country.

What e-learning systems are currently being used in Tanzania?

At Tanzania’s universities the implementation of e-learning systems are still in its infancy. Some FOSS e-learning systems have been implemented at the University of Dar es Salaam (UDSM), Sokoine University of Agriculture (SUA), the Open University of Tanzania (OUI) and Ardhi University. These universities have recently developed policies to incorporate ICTs into their core functions namely: teaching, learning, research and consultancy to improve the quality of their services. A challenge which all universities face when implementing e-learning systems, is the cost of the licensing software (Mutagwahywa et al., 2003). To address this licensing problem the universities in Tanzania have considered adopting and using FOSS e-learning systems instead of proprietary software.

- UDSM has recently implemented e-learning for a number of courses. It established an independent section which deals with the development and provision of e-learning services to the university community. WEBCT and Blackboard are the proprietary software e-learning systems currently being used at UDSM however they are currently in the process of considering FOSS e-learning systems. One that they are exploring is the FOSS e-learning system called KEWL.NextGen developed at the University of the Western Cape (UWC).
- SUA is currently running a pilot implementation of a FOSS e-learning system called Moodle.
- Ardhi University has already implemented a FOSS e-learning system called Dokeos.
- OUT is in process of adopting and implementing a FOSS e-learning system. This has been considered as the case study for this thesis.

Problems with selection and evaluation of FOSS in Tanzania

Out of all the Tanzanian universities, only UDSM has a unit for FOSS. It deals with all matters related to FOSS from design, development, prototype testing, implementation and adoption. Some individuals at the other universities participate in FOSS related activities out of personal interest, but in general, (at the other universities) there is limited collective knowledge about FOSS. This limited knowledge about FOSS is a problem which many ICT users in developing countries face when having to adopt FOSS software products (Câmara & Fonseca, 2007).

Some of the problems users experience when considering to adopt FOSS (Kunda, 2001) are due to the fact that users lack:

- (i) a well defined process for the evaluation and selection of software;
- (ii) the know-how to understand the source code. It sometimes necessitates that small changes are made to the source code of FOSS by user. Unfortunately, it is something that few user cannot make.
- (iii) the necessary knowledge to make changes, as user needs change or technology evolves and
- (iv) an effective method to consolidate data during the analysis of the evaluation.

Shull (1998) is of the opinion that in many cases software practitioners are in a dilemma when it comes to the selection of software tools or techniques. He argued that selection is normally based upon

assumption, gut feeling, anecdotes, IT expert opinions and flawed research. He suggested that instead of using ad-hoc selection methods, there is need to use well-researched empirical approaches.

Research questions explained

From this discussion, it is clear that there is a need for the development of a framework which can guide stakeholders in the evaluation of FOSS before deciding which product to adopt. The advantage of such a framework would be that it would help stakeholders (both technical and non-technical users) to identify the correct FOSS application for deployment in a specific environment.

User knowledge and experience are the key components which enables the user to evaluate quality FOSS. User knowledge and experience can be attained if the users are involved in decision making of software quality (i.e. design, development, implementation, adoption and use). If user opinions are incorporated in the choice of a software system, the deployment of the software will boost software acceptance, user satisfaction and it will also enhance the user's knowledge about the technology itself (Câmara & Fonseca, 2007). It is important to implement quality software as it limits the incidence of software implementation failure. But, how is the quality of software determined?

As mentioned before “*software quality can be defined in terms of a set of characteristics that reflect the user's needs*”. To explain a characteristics a set of sub-characteristic are defined which are in turn described by a set of attributes.

How are the characteristics and attributes that determine software quality, identified?

This task of “*identifying*” software quality characteristics, sub characteristics and attributes is difficult because by nature software is intangible or “*fuzzy*”.

IEEE, Boehm, MCall and ISO 9126 software quality models provide the basis for the identification of characteristics, sub-characteristics and attributes (Seffah et al., 2001). By definition, software quality models can be viewed as blueprints used in the evaluation of software according to some predefined characteristics, sub-characteristics and attributes (of software quality). The software models can be either hierarchical or non-hierarchical. The hierarchical models present different characteristics at the top level of hierarchy followed by different sub-characteristics and attributes at the lowest level. Such a

hierarchical depiction of a software quality model provides the foundation from which measurements for software evaluation can be taken.

Examples of hierarchical models are IEEE, Boehm, McCall and ISO 9126 (Seffah et al., 2001). Seffah et al. have described software quality models as follows:

- i. Boehm's model is a multilevel hierarchy (or tree) of characteristics. Boehm's model includes hardware characteristics. It further categories attributes by viewing a product as a provider of an utility to its different stakeholders (Boehmet al., 1976).
- ii. McCall's model is based on product revision, software operations and software transitions. This model defines the different factors as they are perceived by the users (external views) and software developers (internal views). In summary, it is a model for identifying and decomposing software quality attributes using the user's view of the final product (Cavano & McCall, 1978).
- iii. IEEE 1061 depicts a hierarchical model of quality characteristics, sub-characteristics and metrics (the value of an attribute). It applies this to each phase of software development life cycle.
- iv. The ISO / IEC 9126 model is a standard model which has a hierarchy of six characteristics; these characteristics are divided into sub-characteristics which are also subdivided into measurable attributes. ISO 9126 classifies the characteristics, sub-characteristics and attributes as either developer oriented or user oriented. Developer oriented views involves portability and maintainability characteristics, while the usability, functionality; efficiency and reliability characteristics belong to user oriented views (i.e. quality-in use). The sub-characteristics provide a detailed viewpoint of a characteristic. For example under ISO 9126, the usability characteristic contains the following sub-characteristics: understandability, learnability and operability. Sub-characteristics can be divided into attributes so that they can be quantified. Attributes are the measurable properties of characteristics or sub-characteristics. They can be identified, assessed and measured differently according to specific software systems, users and environments (Koscianski & Costa, 1999).

The weakness of using software quality models are mentioned in studies by Seffah et al. (2001), Blin and Tsoukiàs (2001) and Khosravi and Guéhéneuc (2005), and are:

- i. several evaluators are involved in the evaluation which may complicate quantification of software quality attributes
- ii. the different evaluators may have different objectives
- iii. each evaluator may propose a specific quality model
- iv. in most cases the evaluators choose their own characteristics, sub-characteristics and attributes and not necessarily those from standard software models; and
- v. sometimes all the sub-characteristics are treated as if they have an equal impact on the software.

Osterweil (1996) argues that there is a need for innovative research in “*software quality*”. For example, it is easy to evaluate the quality of hardware because there are established standardized specifications which can be used to measure it. Since software quality is viewed as an abstraction (without physical presence), it is difficult to identify and quantify its quality attributes (Cavano & McCall, 1978). This concurs with Osterweil’s (1996, p. 1) statement:

“there is no single monolithic measure of software quality and, no general agreement about how to quantify definitely any of the key quality concerns”

Furthermore, the recent study by Mebrate (2010) found that using the software quality models which are meant for evaluating generic software in evaluating web based software (this include e-learning software) is not fair. The reasons mentioned by Mebrate are: (i) the model lack justification which characteristics to determine for evaluating a particular web based software in a particular environment (ii) there is no general principle how someone can relate the attributes, sub characteristics to characteristics and (iii) no clear method to measure and compose the attributes, sub characteristics and characteristics to obtain the overall assessment.

Franc and Carvallo (2003) proposed the development of domain specific quality models for attributes selection. Similarly, Covella and Olsina (2006) argue for the need of taking into consideration the users opinions while identifying attributes for the quality-in-use of the software. This aspect is important because user opinions with regard to attribute identification differ. Khosravi and Guéhéneuc (2005) proposed a technique for identifying characteristics and attributes; they suggest looking at the relationship and impact of each characteristic, sub-characteristics and attributes of software in terms of

the quality requirement of the users. This technique facilitates the identification of quality attributes by balancing the quality requirement from the users and the capabilities exhibited by software products (Alves & Finkelstein, 2002).

For closed source (proprietary) software, this technique can be used throughout the development cycle. The distributed nature of the FOSS development methodology complicates the identification of quality attributes. This is because users who participate in FOSS development are often from different geographical locations. This makes it difficult to ensure that quality aspects are considered throughout the development cycle. It is therefore difficult to check characteristics, sub characteristics and attributes for the following:

- the quality of the requirements;
- the quality of the software product developed; and
- the quality of the software product-in-use.

Even though the software quality models (such as IEEE, Boehm, Mccall and ISO 9126) can be used to identify characteristics, many of these characteristics are important but not necessary in the developing world. The characteristics that are likely to influence quality of software (in a developing country), were identified as being: maintainability, usability and deployability (see Chapter 1, page 3). The importance of these characteristics are not the same in all developing countries (Kunda, 2001, p. 35) since countries differ in terms of the availability of resources, experience of users (to maintain the systems), language of the user interface and the environment in which the software systems must be deployed.

How attributes are measured or quantified?

When measuring software for its quality it is necessary to consider both the quality of the software process and the software product-in-use (Polancic & Horvat, 2000). This is what is called the principle of duality of the software measurement.

Measuring the quality of the software process of FOSS is not simple. Most FOSS products are poorly documented (in terms of requirement specification, software architecture and software prototype), thus it is difficult to determine the quality of the FOSS development process and hence Manieri et al.

(2008) argue that the best way to evaluate FOSS according to software quality is to consider the software product-in-use.

Quantifying the quality of the software product-in-use can only imply the quality of the software process and thus the quality of the software product (see Figure 2, adapted from Polancic & Horvat (2000)).

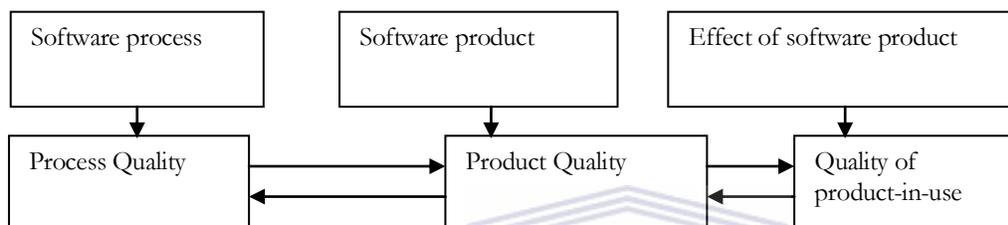


Figure 2: Relationship of process quality, product quality and quality-in-use

↓ This line denote software process implies the measures of process quality, similarly: software product implies the measures of product quality and effect of software product implies the measure of quality of product-in-use.

→ This line denotes “influence”. For example, process quality influences product quality.

← This line denotes “depends”. For example, product quality depends on process quality

How do we define: product quality; process quality, and the quality of a product-in-use in terms of quantification?

Product quality can be defined as the quality of the software product during testing.

Process quality can be defined as the quality of the software development lifecycle processes (Polancic & Horvat, 2000).

Quality of product-in-use is how the product meets the user requirements in the production environment (ibid.). ISO (2001) defines quality-in-use as the quality attributes of the software product exhibited when the final product is being used in real environment. Mebrate (2010) defines quality-in-use as

user's view in using the final software product. Furthermore, quality-in-use has been defined by Bevan as the degree to which the software product used by specified user in their context to meet their specific needs in an actual context of usage to fulfill specific goals with effectiveness, productivity and satisfaction (Bevan, 1999). From these definitions, it can be found that quality-in-use depends on the context of which the software product is being used, types of users and the environment.

The cumbersome attained by user in defining the term quality can lead to the difficulty in defining and hence, measuring the quality of product in use. As discussed before, that there are three different approaches in defining the term quality (Garvin, 1984) and these difference contributes to the subjectivity in measuring the quality-in-use of software product (Basili, 1985; Chang & Dillon, 2006; Chang et al., 2008). This is a problem hindering the adoption of e-learning system in developing countries (Padayachee et al., 2010). Since acceptance of a product is related to its quality, the quality of a product should be defined and measured to improve its acceptance and use of product (Kan, 2003). In addition, the quality of the software product involves the quality measured by considering internal and external attributes (ISO, 2001). But, what is an attribute?

Bertoa and Vallecillo (2002) define an attribute as a quality property to which a metric can be assigned – either a symbol or a number. The goal is to measure the extent or degree to which the software product satisfies a certain quality characteristics, sub-characteristics or attribute (Basili, 1985).

Basili (1985) classified metrics as objective or subjective. According to Basili objective metrics are absolute measures while subjective metrics are relative measures. Objective metrics can best be applied by software engineers who have knowledge of software metrics and the know-how to evaluate a software product (Basili, 1985; Basili et al., 1994). On the other hand, subjective metrics can be identified by all, even users with limited ICT knowledge (Dick, 1993).

Other ways of determining the metrics of the software are: direct (or actual or basic) measurement and derived (or indirect) measurement (Jørgensen, 1999).

- *Direct measurement* produces basic, direct or actual metrics. Direct metrics are obtained from internal attributes, which can be measured by an IT expert. Examples of direct metrics (of software products) are: lines of codes, execution speed, memory, etc. and that of the software process are: cost, effort estimation needed to develop a process, etc. Furthermore, basic

(direct or actual) metrics are those which do not need any further division (or decomposition), while derived metrics are those which depend on further decomposition.

Direct metrics are objective but have the following disadvantages, namely: (i) programming language dependent (ii) also require a detailed analysis to be done by a competent IT expert (iii) there is little agreement in terms of the relationship between source codes and the quality exhibited by the software product.

Dick (1993) listed the following advantages of direct metrics: (i) they evaluation process of the direct metrics can be automated (ii) the results from analysis of metrics can be reproduced or replicated in an experiment.

- *Indirect measurement* produces derived or indirect metrics. Indirect metrics are those obtained from external attributes. This can be viewed externally from the properties of the deployed software (Pressman, 2005). The indirect metrics of the software products can be obtained after decomposing its characteristics (functionality, usability, reliability, deployability, maintainability, etc (Dick, 1993)) into its respective attributes.

The disadvantage of indirect metrics is that they are subjective. It is difficult to automate the evaluation process of indirect metrics since they are subjective and dependent on the experience of the evaluators. Thus in order to obtain indirect metrics both qualitative and quantitative methods (by users in a particular environment) should be used (Fenton & Pfleeger, 1997).

The importance of qualitative and quantitative methods is on the collection and analysis of the collected data during either software process development or software product testing or software implementation. Basili (1985) argues that the foundation for quantitative method is on setting the goals which can be refined into questions (that are quantifiable). The value obtained after quantification is the software quality attributes of the software product. Experimental, quasi-experimental, correlational, and descriptive are examples of quantitative research (Waterfield, 2010). On the other hand, qualitative method is based on describing, interpreting and explaining the phenomena of software quality exhibited by a software product. Narrative, phenomenology, grounded theory, ethnography, and case study are examples of qualitative research (ibid.).

Different researchers have discussed the motivation why qualitative and quantitative methods can be used (Johnson and Onwuegbuzie, 2004; Waterfield, 2010). According to Johnson and Onwuegbuzie (2004), the following are advantages and disadvantages of quantitative and qualitative methods.

According to Johnson and Onwuegbuzie (2004), the following are the advantages of quantitative method:

- They can test and validate already constructed theories about the phenomena.
- It is possible to test hypotheses that are constructed before the data are collected.
- Generalization of research findings can be done if data are random sampled from sufficient size.
- Also, you can generalize the research finding when the research has been replicated on many different populations.
- Valuable for obtaining data that allows predictions.
- It allows the researcher to reduce variables and hence, cause-and-effect relationships can be assessed credibly.
- Provides data collection methods which are relatively fast.
- Data collected by it are precise, quantitative and numerical data.
- It provides data analysis which are relatively less time consuming.
- The research findings are relatively independent of the researcher (i.e. no bias).

According to Johnson and Onwuegbuzie (2004), the following are the disadvantages of quantitative methods:

- The categories and theories used by researcher may not reflect user understandings in particular environment.
- Since the researcher focus on theory or hypothesis, it is easy to miss out on phenomena occurring.
- The research results obtained may be too abstract and general for direct application to specific contexts and environment.

According to Johnson and Onwuegbuzie (2004), the following are the advantages of qualitative methods:

- The data collected are based on the respondents' own categories of meaning.
- It is useful for researching a limited number of cases in detailed.

- It is useful for explaining and describing phenomena which are complex.
- Allows provision of individual case information.
- It allows the researcher to do cross-case comparisons and analysis.
- It gives the viewpoint of participants
- It provides the description of the phenomena in detail as they are situated in particular contexts.
- It is possible for a researcher to contextualise and set factors with respect to the phenomenon of interest.
- Explanatory theory about a phenomenon theory can be induced.
- Using this method, data are usually collected in naturalistic settings
- Qualitative methods are responsive to particular environment, context, and stakeholders' needs.
- Data collected in the words and categories from the participants provides exploration of how and why phenomena occur.
- It provides the causes of a particular event.

According to Johnson and Onwuegbuzie (2004), the following are the disadvantages of qualitative methods:

- The research findings from it may not be generalized to other people or other settings.
- Quantitative predictions are difficult.
- Testing hypotheses and theories is also difficult.
- Sometimes, it results has lower credibility to different stakeholders.
- Time consuming in data collection and analysis when compared to quantitative research.
- The research results are influenced by the researcher's personal biases

Thus, qualitative and quantitative methods can be used to encapsulate the user evaluation judgement in terms of subjective attributes, into a value (Basili, 1985; Chang & Dillon, 2006; Chang et al., 2008, Padayachee et. al., 2010). This process is possible because mapping software quality phenomena to a numerical value, is accepted practice in measurement theory and software measurement (Fenton & Pfleeger, 1997). Fenton and Pfleeger illustrated the schematic representation of four aspects, which constitutes software measurement (see Figure 3): measurement, collection of measurement, collation and storage of measurement and analysis of the collected measurements (ibid.).

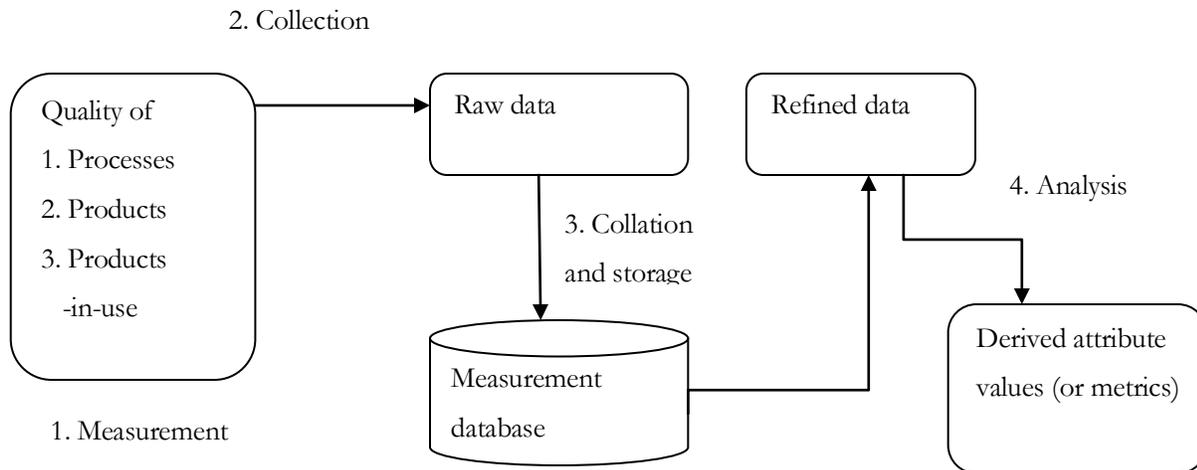


Figure 3: High Level Measurement Context

How can the specific software quality characteristics of interest to this study be quantified?

Deployability

The first software quality characteristic of interest in this study is *deployability*. Deployability refers to ability of software to be packaged for a usage in a certain environment. Bertoa & Vallecillo (2002) defined portability as the ability of software to be able to be transferred from one environment to another. Thus, the term portability is incorporated in the term deployability. Deployability is included in some software quality models and different software quality models (i.e. IEEE, Boehm, Mccall and ISO 9126) provide different sub characteristics of deployability characteristic (Seffahet al., 2001).

The sub characteristics of the deployability characteristic are portability, installability, adapatability, configurability and distributability.

The following are the sub characteristics of deployability with their respectively attributes were used in this study:

- a. Portability is the ability of the free and open source software to be transferred from one environment to another.

- i. software system independence: this refer to the extent at which the free and open source software is independent of operating system and other environment constraints (type of databases management system, programming languages)
- ii. machine independence: the ease at which the free and open source software can run in any machines with minimum consumption of resources (software, hardware, network)
- b. Installability: the ability to be installed to suite a particular environment
 - i. ease of installation: this refers to the effort required for the software packages to be installed in a production environment
- c. Adaptability: the ability of free and open source software to be installed in the different environment without requiring any additional actions from those specified in the user manual.
 - i. Suitability for personalization: this express the capability of the free and open source software to be able to provide a specific user functionalities according to experience.
 - ii. Adaptivity: this refers to the capability of the free and open source software to accommodate the needs/requirements of different user.
- d. Configurability: the capability of the software packages to be setup for running in production environments
 - i. Technical documentation: the ease at which the technicality of the free and open source software is explained in the manual
- e. Distributability: the ability of the software to run at different locations concurrent.
 - i. distributed system: the ease at which the system can be deployed as distributed system

Li et al. (2005) used a metric tool to identify *deployment* and usage metrics from mailing list archives, control version systems (CVS) and bug tracking systems. According to Li et al. (2005), little research has been conducted in the measuring of metrics for the deployment of software systems. Thus, the way forward is determine how to apply subjective software measurement, in order to quantify deployability.

Maintainability

The second software quality characteristic of interest in this study is *maintainability*. Maintainability is the ability of the software to be corrected, adapted and improved depending on the changing user requirements or functional specifications. Bertoa & Vallecillo (2002) defines maintainability as the ability of software to be modified.

Different software quality models (i.e. IEEE, Boehm, Mccall and ISO 9126) provide different sub characteristics of maintainability characteristic (Seffah et al., 2001). According to ISO (2001), Bertoa & Vallecillo (2002) and Khosravi & Gueheneuc (2004) the sub characteristics of the maintainability characteristic are complexity, stability, analyzability, changeability, testability, trackability, flexibility and upgradeability.

The following are the sub characteristics of maintainability with their respectively attributes (Bertoa & Vallecillo, 2002; Khosravi & Gueheneuc, 2004) were used in this study:

- f. Complexity: this refers to the complexity of packaging and using the system
 - i. Complexity of the provided interface: this refers to ability of user to operate the free and open source software without ambiguity. It reflects the number of the provided interfaces of the software.
- g. Stability: the ability of the software to withstand unexpected effects from the modification process
 - i. Occurrence of error: refers to ability of the free and open source software to recover from error. (i.e. self handling of error without crushing/hang)
- h. Analyzability: this refers to the ability of the free and open source software to be able to indicate the errors, deficiencies (weakness or causes of errors/failures) or artifacts which need to be modified
 - i. tracing error: the ease at which the errors can be pinpointed when they do occur in the production environment
- i. Changeability: refers to ability of easiness and effort required to enable modification of the free and open source software.

- i. Customizability: the ability of free and open source software to be able to be customized according to the evolving user needs.
 - ii. Extensibility: refers to ability of the free and open source software to be able to adapt new features / functions as the improvement is done. This is usually done to extend its functionality. Furthermore, the aspects of the compatibility of the new version with the previous versions are included in extensibility.
 - iii. Portability: the easy at which the modified software can be transferred to certain environment
- j. Testability: this refers to the attributes that facilitate the modification and testing of the functionality of the software.
- i. Observability: this indicates if the start up self-test have been provided to check the performance / functionality of the free and open source software during modification
 - ii. Controllability: this indicates if the free and open source software was extensive tested before release using test suite of its packages.
 - iii. Accessibility: this refers to the kind of environment under which the testing of the free and open source software was done. This is a good indicator of the type of the environment which the software will operate without problem. Are start up self-test and test suite accessible?
- k. Trackability
- i. Look and feel: tracing/ monitoring /reporting the software modification
- l. Flexibility: this refers to ease at which the software can be modified to adapt it to other environment or to different application which was not meant for.
- i. Scalability: the ease of free and open source software to support the incremental growth of data volumes from user as well as modify to expand its capabilities. Also regardless of the data growth the software must be efficiency (i.e. processing capacity)
- m. Upgradeability
- i. Easy to upgrade: refers to easiness of the software to be upgraded to a new version of the free and open source software.

Researchers argue that it is important to start evaluating maintainability of software in the early stages of the software development process because it is a means of reducing faults/errors of a product,

which in turn reduces maintenance costs (Garcia et al., 2005, Lee & Jefferson, 2005). Different researchers have used different methods to measure the maintainability characteristic but there is no agreed method to measure maintainability (Hayes & Zhao, 2005). Some of the proposed methods are: (i) regression analysis measure, (ii) linear model, (iii) correlation method, (iv) effort based metric, and (v) hierarchical tree (ibid.). The key method for all these methods is the hierarchical tree structure which identified and used 92 software quality attributes to measure the maintainability characteristic (Oman & Hagemester, 1992). The disadvantage of most of the many methods mentioned above is that they do not address the uncertain judgements of the evaluators during subjective measuring of this characteristic. Furthermore, most of the methods described are meant for technical IT users who understand the source codes. One would think that since the source codes of FOSS is available on the Internet and the several methods have been developed to measure maintainability, that it would provide the necessary and sufficient datasets for the researchers to measure this metric. However, according to Yu et al. (2005) this is not the case. They argue that even though data such as data from change logs, from source codes and from defect tracking, is accessible online, these do not necessarily measure *maintainability*. As indicated by Dick, direct software measurement of software quality attributes is complex (Dick, 1993).

Usability

The last software quality characteristic of interest is *usability*. The usability characteristic deals with the capability of software to provide the easy of use, easy to learn, easy to understand and configure under specific environment (Boloix, 1997). This definition is similar to that of Sharp et al. (2007) who defined usability as the degree at which the software products are easy to learn, effective to use and user friendly from the user's perspective.

Different software quality models (i.e. IEEE, Boehm, McCall and ISO 9126) provide different sub characteristics of usability characteristic (Seffah et al., 2001). For example, according to ISO (2001) and Seffah et al. (2001) the sub characteristics of the usability characteristic are understandability, learnability, operability and attractiveness.

The following are the sub characteristics of usability with their respectively attributes (Seffah et al., 2001) were used in this study:

- a. Learnability: it refers to the time required for a user to learn to use, configure and administer free and open source e-learning software.
 - i. Time to configure: refer to time for a user to learn how to configure the free and open source software.
 - ii. Time to expertise: refer to time required for a user to administer any task on the free and open source software
 - iii. Time to use: refer to time required for a user to learn how free and open source software can be used.
- b. Understandability refers to the capability of free and open source software to provide all necessary documentation for assisting user in using the software in the simplest manner.
 - i. Assistance / training: refers to the extent, depth and effectiveness of the guidance provided through different help systems
 - ii. User documentation: this refers to the completeness, readability, clarity, usefulness and understandability of the user manual, installation and administration manuals.
 - iii. Help system: refers to the completeness of the help files provided to the user to discover and understand its services/functions
 - iv. Demonstration coverage: this refers to the thoroughness of the provided demos (i.e. snapshots), tutorials, sample codes, online support (i.e. discussion groups, error tracking systems). These provide demos must be compared to all that are available to the interface of software.
- c. Operability: this refers to the ability of the free and open source software to indicate the level of effort needed to operate and administer.
 - i. Effort to operate: it refers to effort required to operate the free and open source software. E.g., task operated manually for software to execute. This determines the user satisfaction.
 - ii. Tailorability: refers to effort require to customize the free and open source software by configuring its packages/parameters
 - iii. Administrability: refer to effort required for administration tasks.
- d. Attractiveness: this covers the extent at which the free and open source software product attract user through layout, color and graphics designs.

- i. Confidence: this refers to whether user are at ease when using the free and open source software (i.e. simplicity of the interface)
 - ii. Error correction and prevention: the capability of the free and open source software interface to monitors different operation from user and identify error, in case of its occurrence, as well as its ease to rectify error by user and prevention repetition of error
 - iii. User control: it deals with whether the user are in control (i.e. mange) when performing different operation on the free and open source software
 - iv. Satisfaction: this is concerned with the ability of free and open source software to satisfy specific user in particular environment (i.e. user feelings when using the software)
 - v. General user support: ease at which the interface of the software is supportive to fulfill the intended aim/goal
 - vi. Informative feedback to user: This is concerned with the reaction/response of the free and open source software when user is using to perform certain tasks.
 - vii. Compatibility with user conventions and expectations: this deal with providing a proper meaning which matches the graphics of the interface. Also refers to ability of free and open source software to complete each task executed by user successful.
 - viii. Consistency of screen presentation: this refers to the uniformity of the user interface in conveying the intended meaning.
- e. Usability compliance (conformance): refers to capability of free and open source software to comply/adhere to standards, guidelines, convections related to usability set by /certified by external or internal organization or standardization body
- i. Standard: this refers adherence of the free and open source software to the usability standard set by ISO 9126

Nichols and Twidale (2003) argue that even though the open source community has successfully developed several software products (freely available on the Internet), many computer users still prefer to use proprietary applications (closed source software). Nichols and Twidale commented that poor *usability* of FOSS might be the causative reason for this trend. They feel that the tendency of developers (of FOSS) to sometimes develop the software without knowing real users (or involving the end-user), may contribute to this phenomenon. To address this problem of poor uptake of FOSS, developers should involve both technical and non-technical IT users in all phases of software

development. This has the advantage of pinpointing conflicting quality requirements because of the preferences of different users (Nodder et al., 1999; Nichols & Twidale, 2003).

Although the pedagogical aspect of e-learning software will not be considered in this study, its importance is acknowledged in terms of its usability. The usability of an e-learning system is different from other systems (Mayes & Fowler, 1999). Usability of e-learning system must have (i) user friendly interface (ii) interactive facilities to help communication between instructors and learners (iii) accessibility (iv) consistency (v) good organization of course contents (vi) understandability of course contents (vii) multiple language support (Mebrate, 2010). Ardito et al. (2004) and Zaharias (2004) argue that the interface of a poorly designed e-learning system can become a barrier for learning. Research by McCarty (2005) highlighted the importance of paying attention to language and its contexts when considering the interface of an e-learning system. The contexts can be English as a second language (ESL) or English as a foreign language (EFL). Therefore, when usability of software is measured, and compared, one solution will not fit all contexts, each should be treated separately.

Ardito et al. (2004) used a systematic usability evaluation (SUE) method to test the usability of e-learning systems. Ardito et al. found that the problems facing students were mostly in terms of: functionality, presentation and orientation. According to their literature review and pilot study, the dimensions to consider for usability evaluation are presentation, hypermediality, application proactivity and user's activity. They argue that it is important to evaluate the e-learning system (container) and the educational module (content) separately because the quality of the system determines the quality of educational module.

As can be seen from the above discussion, software quality attributes identification and attributes measurement for FOSS e-learning systems, is still problematic.

Framework formulation for the evaluation of the quality of software

The above discussion raises the question: how can quantified attributes be used to create a framework for the measurement of the quality (of the experience of users) of e-learning systems in a developing country? Several researchers have suggested methods for determining software quality. Some of these methods, as well as their advantages and disadvantages, will be discussed and summarised. Fuzzy AHP will be discussed as a method that could address the weakness of some of the existing frameworks.

Methods for the evaluation of software

Traditional methods for software evaluation involve IT expert judgements (Boehm, 1981). Usually an IT expert evaluates software independently; the results from different IT experts are then merged to find the most appropriate software for a specific purpose. According to Somerville (2004), this approach can be biased because it is subjective. A more objective approach was suggested by Basili et al. (1994) who used the GQM paradigm to evaluate software. They found that the advantage of GQM is that it is very flexible. They however indicated that a disadvantage of GQM is that it can only be used by experienced IT users who are able to define and answer questions relevant to the purpose of the evaluation (which differs depending on the environment of application). A further disadvantage of GQM is that it does not allow the comparison of software – that is, it does not provide a means of combining different attributes into a single value for the sake of comparison.

Unlike GQM, all IT users irrespective of their experience QWS can be used. Graf and List applied the QWS method (Graf & List, 2005) to evaluate several FOSS e-learning systems. They found that QWS is simple to use and thus even novice IT users can be used in the evaluation of software. However, its disadvantage is that it uses symbols for the evaluation and because it is difficult to combine the attributes into a single value for comparison, it is difficult to compare the different software packages.

Saaty showed that the AHP algorithm can be used to solve any multi-criteria decision making problem (Saaty, 1980). Blin and Tsoukiàs undertook an investigation into using multi-criteria methodologies for the evaluation of software quality (Blin & Tsoukiàs, 2001). They recommend using a multi-criteria methodology over using the weighted sum method. The multi criteria methods allow aggregating homogenous as well as non-homogenous information (characteristics, sub-characteristics and attributes) regarding software quality.

Aggregating weights from different levels of a hierarchy (tree) can be done using the AHP algorithm (Koscianski & Costa, 1999). A further advantage of AHP is that caters for a step-wise evaluation of software, which helps in visualizing the problem clearly. This step-wise evaluation makes it suitable for use in developing countries since technical expertise in most developing countries is limited. According to Koscianski and Costa (1999), AHP has the advantage of simplifying the process of choosing software. Like GQM and QWS, AHP uses subjective information as data.

The advantages and disadvantages of the frameworks for the evaluation of software

According to Carrey and Wallnau (1998), the principles of software evaluation are: (i) related to decision making (ii) inherently subjective and hence must accommodate the technique that address the element of uncertainty (iii) based on the theory that the abstracted software design must take the context and environment where it will be deployed, into account (iv) rooted in specificity, this means there is no framework fit for all environments. This derives the premise: the way software evaluation must be done for a specific software product in a specific system (or context) is by using a specific set of criteria identified by the user (or evaluators) (Carney & Wallnau, 1998).

Even though several frameworks exist for the evaluation of software each has its advantages and disadvantages (Koscianski & Costa, 1999; Kunda, 2001; Pruengkarn et al., 2005; Covella & Olsina, 2006). The following (see Table 1 – 3) is a summary of the properties, advantages and disadvantages of some of these frameworks.

Framework	Properties	Advantages	Disadvantages
Web quality characteristics tree (Pruengkarn, Praneetpolgrang, & Srivihok, 2005.)	Data for this framework were collected from students using questionnaires and they were ranked according to the weight values.	To some extent it is objective.	It does not handle the problem of uncertain judgements.
INCAMI (Information Need, Concept model, Attribute, Metric and Indicator) (Covella & Olsina, 2006)	It uses a Web Quality Evaluation Methodology to collect data for metrics and user satisfaction. It involves the use of questionnaires, screen sequences, videos and systematic usability evaluation method.	It is objective.	It does not address subjective evaluation judgements.
Combination of ISO / IEC 9126 and AHP (Koscianski & Costa, 1999)	Software is evaluated as whole and not constitutes parts only.	Both subjective and objective.	It does not show how subjective judgements are addressed.

Table 1: Advantages and disadvantages of existing frameworks for evaluation of software

Framework	Properties	Advantages	Disadvantages
Socio technical framework for COTS (commercial off-the-shelf) software evaluation (STACE)(Kunda, 2001)	<p>It uses a systematic approach for evaluation and selection of COTS.</p> <p>It recommends an evaluation strategy that first selects the underlying technology and then the COTS products.</p> <p>The use of social-technical techniques is proposed to improve the COTS software selection and evaluation process.</p> <p>It incorporates multi-criteria decision-making algorithm (i.e. AHP) to merge attributes.</p>	<p>It deals with the non-technical factors through the application of social technical techniques.</p> <p>Both COTS products and the underlying technology are evaluated and selected.</p> <p>It incorporates a database of evaluation results, which make the use of previous results possible.</p> <p>Consensus, transparency and consistency checking are enhanced through use of AHP.</p>	<p>It does not address effectively the evaluation and selection of software for smaller company and projects.</p> <p>The inclusion of non technical factors increases the number of attributes and hence the number of comparison.</p> <p>AHP does not handle bias judgements - it does not show how subjective judgements are addressed.</p>
Software system evaluation framework (SSEF) (Kunda, 2001)	<p>It proposes a top-down approach that needs a user to identify the important elements that must be included in evaluation of software.</p> <p>It uses different world's views (i.e., usage world, development world and system world).</p> <p>It applies multiple viewpoints to evaluate user satisfaction and economic returns.</p> <p>It reduces the evaluators' conflicting viewpoints by providing the definition for: dimensions, factors, and categories.</p> <p>It is grounded in three dimensions (i.e. software's producers, operators, and users).</p>	<p>It provides a foundation for establishing metrics.</p> <p>It allows the incorporation of different perspectives from end users, developers, and operators.</p> <p>It is flexibility.</p>	<p>How to define the evaluation criteria are not addressed.</p> <p>It provides little detailed insight about strengths and weaknesses of its technology in comparison with others.</p>
Off-the-shelf-option framework (OTSO) (Kunda, 2001)	<p>It uses incremental, hierarchical and detailed definition of evaluation criteria.</p> <p>It suggests a model for comparing the costs and value associated with each alternative.</p> <p>It uses an effective decision-making method for evaluation.</p>	<p>It handles the complexity of COTS software evaluation.</p> <p>Learning can be achieved from the experience of the use of systematic repeatable process. This can improve the COTS selection process.</p> <p>The incorporation of AHP help to provide evaluation consistency and structured information.</p>	<p>AHP is only suitable when there are few comparisons criteria and for independent criteria.</p> <p>It does not include the non-technical factors or "soft" factors.</p>

Table 2: Advantages and disadvantages of existing frameworks for evaluation of software

Framework	Properties	Advantages	Disadvantages
Delta technology framework (Kunda, 2001)	<p>It is used in evaluation of a software and technology by investigating its features in relation to others.</p> <p>It uses a systematic approach, which involve modelling and experiments.</p> <p>It evaluates how new technology's features "delta" differ from other technologies in addressing the needs of specific usage contexts.</p>	<p>It can be used to evaluate the underlying technology of the software product.</p> <p>It can also determine individual product evaluations by differentiating their characteristics in relation to their technology and product related to them.</p>	<p>It focuses on technology evaluation and neglect product and vendor evaluation.</p> <p>It does not deal with the political and economic factors that often contribute to a chosen technology from others under consideration.</p>
Procurement oriented requirements engineering (PORE) (Kunda, 2001)	<p>It combines the existing requirements engineering methods and others, for example: feature analysis and multi-criteria decision making.</p> <p>It provides guidelines for conducting evaluation through template-based approach.</p> <p>It promotes for a parallel and an iterative requirements elicitation and software product selection.</p>	<p>It provides procedure to model requirements for COTS software selection.</p> <p>Requirements elicitation informs COTS software selection and vice versa because of conducting those processes in parallel.</p>	<p>It is vulnerable to abandon social factors because of use of traditional approaches.</p> <p>It is laborious.</p>
Expert system for software evaluation (ESSE) (Vlahavas, Stamelos, Refanidis, & Tsoukias, 1999)	<p>It automates the software evaluation process.</p> <p>It provides suggestion of a software evaluation model depending on the type of the problem.</p> <p>It checks consistency of the evaluation model.</p> <p>It selects appropriate MCDA method according to the information available.</p> <p>Evaluators compare software with the assistance of the expert module.</p> <p>It makes use of the past evaluation results.</p>	<p>It is generic and hence offers flexibility in solving evaluation problems.</p> <p>It is objective.</p>	<p>MCDA is only suitable when there are few comparisons criteria and for independent criteria.</p> <p>It does not show how subjective judgements are addressed.</p>
Metrics based decision analysis (Paul, Kunii, Shinagawa, & Khan, 1999)	<p>It integrates database of metrics and analytic tools.</p> <p>Software metrics are stored throughout the software development and maintenance cycle.</p>	<p>It is generic.</p> <p>It shows how subjective metrics are addressed.</p>	<p>More work is needed to automate all its components.</p> <p>It cannot be used by novice user.</p>

Table 3: Advantages and disadvantages of existing frameworks for evaluation of software

The frameworks depicted above in Table 1, Table 2, and Table 3 confirms the need for incorporating the technique or algorithm, which can address or handle the subjective judgements from evaluators (or users).

The use of neural networks and fuzzy logic to deal with subjectivity was proposed by Covella and Olsina (Covella & Olsina, 2006) as well as Koscianski and Costa (Koscianski & Costa, 1999). These researchers pointed out that the framework of combining AHP and ISO / IEC 9126 is useful but suggested that soft computing techniques (such as neural networks and fuzzy logic) could be effectively used to deal with subjectivity and uncertain judgements.

Ross described the concept of *uncertain judgements* (Ross, 2004): What causes so-called *uncertain judgements* (during the evaluation of software systems)? According to Ross, it is caused by: (i) the complexity of the system, (ii) ignorance of evaluators / stakeholders, (iii) various classes of randomness (iv) inability to do measurements (v) lack of knowledge and (vi) vagueness and fuzziness inherent to the natural language with which users normally give their opinions or judgements.

The above uncertain judgements can be handled by fuzzy logic. Fuzzy logic is a logic that deals with continuous variables and could probably address the uncertainty of judgements (see Appendix N). According to Ahmad et al (2004), fuzzy logic is the best method compared to deterministic approaches, algorithmic approaches, probabilistic approaches and machine learning (Ahmad et al., 2004). The extension of AHP which address uncertain judgements will be called Fuzzy AHP.

Can Fuzzy AHP address the subjectivity of judgement?

Many researchers have considered the validity of using AHP to solve multi-criteria evaluation problems (Lai, 1995). Mikhailov (2003) identified that the weakness of AHP is that it can't be used when judgements are considered to be uncertain. AHP is applicable when a user can give crisp (exact) comparison values during the evaluation of software products. In practice, this is not always possible since human evaluation judgement can sometimes be vague. The factors that contribute to ambiguity/fuzzy/uncertainty judgements are: (i) lack of enough information about the domain problem, (ii) incomplete information, (iii) lack of method for data validation, (iv) changing nature of the problem (dynamic), (v) lack of appropriate measure or scale, or being uncertain about the level of preferences. Mikhailov argues that the best way to solve uncertain judgement is to express it in terms of fuzzy sets or fuzzy numbers (Mikhailov, 2003).

Zadeh suggests that any crisp theory can be fuzzified by applying fuzzy set theory within that theory (Zadeh, 1994). The extension of AHP with fuzzy sets is called Fuzzy AHP (Mikhailov, 2003). It comprises the steps of conventional AHP, with fuzzy logic, namely: (i) structuring the problem into

hierarchy; (ii) computing the pairwise comparison matrix to obtain the weight or priority vector and (iii) computing the global prioritization weight.

Different approaches have been proposed by researchers to compute the prioritization weight. The approaches differ according to whether they are applied to crisp preference values or fuzzy preference values. Examples of the approaches for computing crisp preference value (Saaty, 1980; Golany & Kress, 1993; Wang et al., 2008) are: (i) the Eigenvalue method (ii) distance functions (iii) least squares (iv) weighted least squares (v) logarithmic least squares (vi) logarithmic least squares with absolute values and (vii) the goal programming method.

Examples of the approaches used for computing fuzzy preference values (Wang & Fu, 1997) are: (i) extent analysis (ii) fuzzy preference programming (iii) fuzzy goal programming.

Wang and Chin (2008) used extent analysis to compute priorities for fuzzy judgements. Mikhailov (2003) applied fuzzy preference programming method for fuzzy judgements to address the weakness of other methods in applying fuzzy AHP, namely: (i) all methods derive priorities from fuzzy comparison matrices, (ii) fuzzy priorities obtained lead to the final fuzzy scores results, which are also fuzzy, (iii) ranking can be done by using different methods in the defuzzification of the final fuzzy scores, although this can result in giving different outcomes (Bortolan & Degani, 1985).

Srdjevic (2005) proposes a combined method for prioritization which combines methods from the traditional AHP and Fuzzy AHP. In the Srdjevic's method, the extent analysis method was not included.

Chang (1996) showed the method for finding priority weight using the basic theory of extent analysis in Fuzzy AHP. Mikhailov (2003) stated that the fuzzy extent analysis method has problems especially because of its use of the arithmetic mean method to compute fuzzy priorities. According to Saaty (1980), this technique has no problem if the evaluation judgements are consistent. Zhu et al. (1999) introduced an improved approach of Fuzzy AHP (with extent analysis). The fuzzy extent analysis method has been applied in certain research studies, for example, Bozdog et al. (2003) applied Fuzzy AHP (with extent analysis) in selecting computer integrated manufacturing systems. Kwog and Bai (2002) also used Fuzzy AHP (with extent analysis) and noted that it was effective in solving multi-criteria evaluation problem. However, its disadvantage is that it has the problem of eliminating some

software quality characteristics, sub-characteristics and attributes during decision process. Wang and Chin (2008) suggest that the use of extent analysis can lead to incorrect decisions. Also, the extent analysis method does not show how consistency can be checked. This calls for more studies to be done to rectify the problem of extent analysis and consistency checking in fuzzy AHP.

The work by Leung and Cao (2000) gives a good outline of how consistency checking and ranking can be done when using fuzzy AHP. They questioned all research done in with Fuzzy AHP but without testing consistency. On the other hand, Saaty and Tran (2007) question research done using a method that fuzzifies AHP. Their stance is that all judgements before being fuzzified are already fuzzy and they are worried that fuzzification might make the results even more inconsistent. However, the study by Leung and Cao (2000) that describes the procedure in computing the consistency of evaluators' judgements, may provide answers to their doubts on consistency. Mikhailov (2003) provides a method for deriving fuzzy priorities when using Fuzzy AHP without requiring calculation for aggregation and ranking procedures. This approach is best used in non-linear equations approach by computing priorities without calculating fuzzy comparisons, it addresses the doubts raised by Saaty and Tran.

This research concurs with Triantaphyllou et al. (1997) that multi-criteria decision methods are controversial and that there is no unique theory accepted by all in the field. In general, decision theories such as liner programming, queuing theory, dynamic programming, inventory and scheduling models seem to be more accepted. Fuzzy AHP (with extent analysis) needs to be improved to address the shortcoming stated in the literature reviewed. Having said that, little research has been done in this area and specifically none for the design and development of a fuzzy algorithm for the evaluation of FOSS in a developing country (Chang & Dillon, 2006; Chang et al., 2008). This thesis thus proposes a new algorithm, "Group Fuzzy AHP", an improved algorithm that could replace the Fuzzy AHP (with extent analysis) algorithm.

The open questions (identified by Paul et al., 1999) that need to be asked in order to evaluate software are:

- a) Of which metrics should data be collected? (*metric database*)
- b) How to get the information for the metrics data? (*query formulation*)
- c) What analysis to perform? (*metrics analysis*)

- d) What information is needed to make a decision? (*decision analysis*)
- e) What feedback should be provided, to whom, and at what levels of abstraction? (*feedback for process / product improvement*)
- f) How to use rule-based systems, truth – maintenance systems etc, for decision making and information extraction for use now and the future? (*decision querying, prediction or forecasting*)
- g) What information is needed for either process or product or product-in-use control to improve quality and productivity? (*process or product improvement*) (Paul et al., 1999)

Paul et al.'s questions summarises the problems experienced with the evaluation of software. These questions can be used as the foundation for the formulation of any framework for the evaluation of software according to software quality.

Paul et al.'s question “*what information is needed to make a decision?*” forms the core part of this thesis. The idea is to design a framework, which can be used in the development of a decision support system. Each of the frameworks discussed in the literature review, has some or other weakness. To address some of these weaknesses, an algorithm is proposed, it will address C and D in the Paul et al.'s framework (depicted in Figure 4).

This thesis will present work towards the design and implementation of an algorithm to enhance decision making in terms of the selection of FOSS e-learning systems.

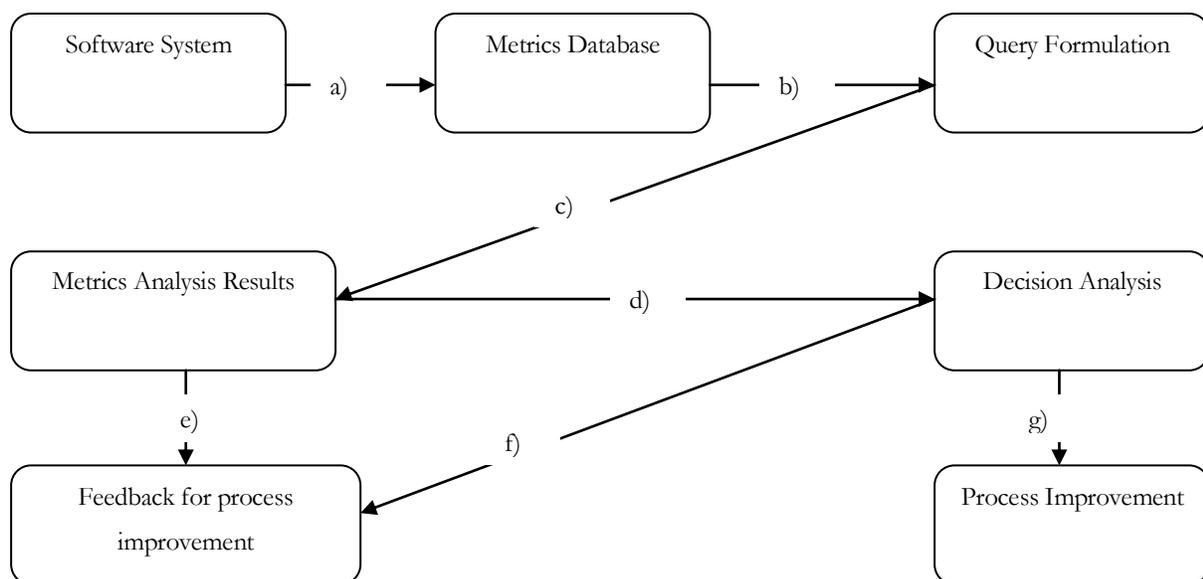


Figure 4: Framework

Conclusion

The key concepts, around which the study is built, were clarified. In this chapter the boundaries of the research was demarcated and the literature was discussed in terms of the research questions identified in chapter 1. The focus of the research was in a case study done at OUT in Tanzania. Finally, the shortcomings of current methods used in software quality attributes identification, software quality attributes measuring and framework formulation, were identified. In the next chapter the research approach, research methodology as well the research methods used, will be discussed.



CHAPTER 3

RESEARCH DESIGN AND METHODOLOGY

As far as the laws of mathematics refer to reality, they are not certain, and as far as they are certain, they do not refer to reality.

Albert Einstein [1879-1955]

Introduction

In the previous chapter, the literature was reviewed in terms of the research questions asked. The research approach, research methods as well as why a combination of research methodologies are proposed, will be discussed in this Chapter. Furthermore, how data was analysed, will be explained. At the end of the Chapter, the limitation of the study will be discussed.

Research Approach

According to Crotty, four questions need to be posed to explain the pivotal issues of the research (Crotty, 1998):

- (i) What *method* to use?

The researcher needs to identify the technique or procedures for collecting and analysing data according to the research questions.

- (ii) What *methodology* governs the choice and use of the proposed methods?

The methodology is the strategy, plan of action or design to obtain the desired results.

- (iii) What *theoretical perspective* (or approach) is suggested?

Thus, what is the philosophical stance behind the suggested methodology?

- (iv) And finally, what *epistemology* informs the suggested theoretical perspective?

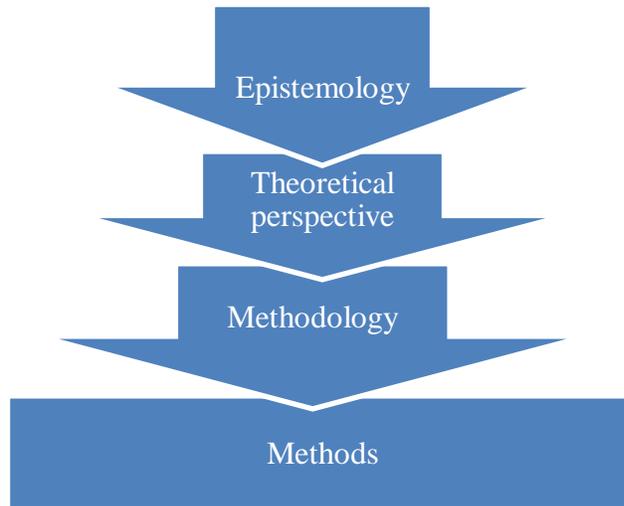


Figure 5: Four elements of the research process

The outline of the research for this thesis (using Crotty’s model) is represented below (see Figure 6):

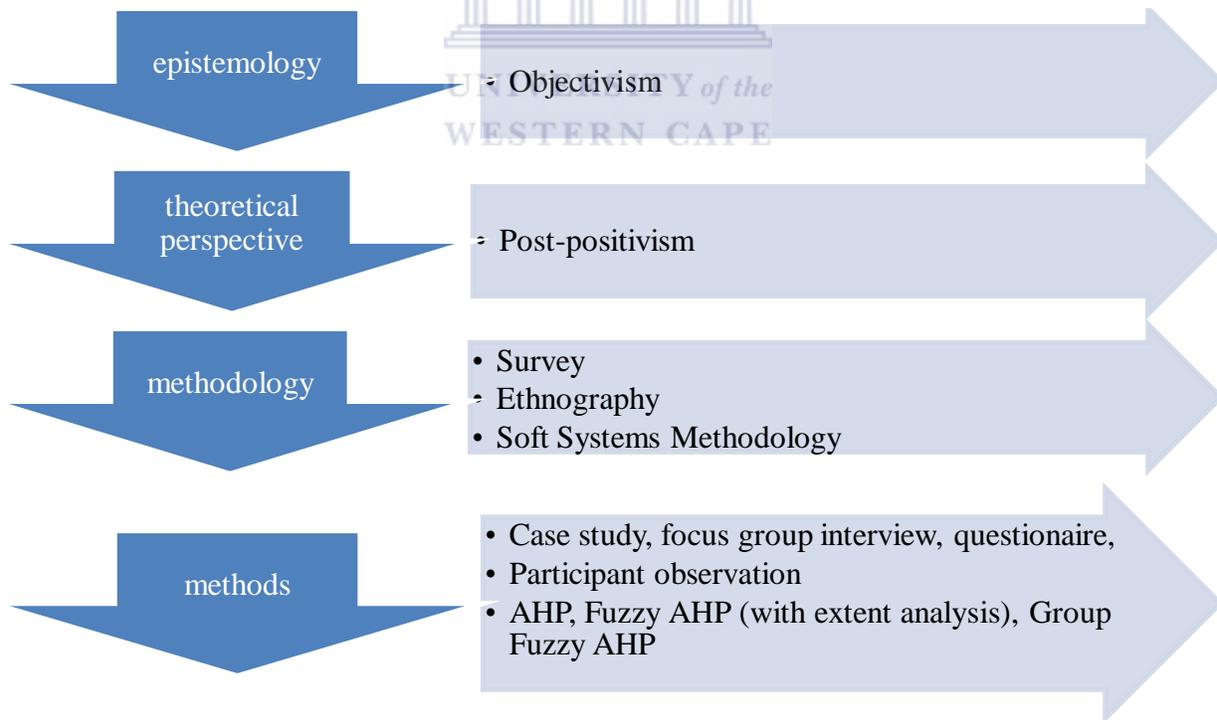


Figure 6: The four elements of research as was used in this thesis

Epistemology

Epistemology is the theory of the knowledge. It includes its origin, nature, methods, validity and limits (Rand, 1979).

According to Rand's philosophy, the task of human's consciousness is to reason about reality (the external world). This is the only means to acquire knowledge through perception of reality. Guba and Lincoln (1985) mentioned interpretivism, subjectivism and objectivism as the dichotomy of epistemology. Objectivism rejects the assertion that individuals or a group create or invert their own reality (Crotty, 1998). Interpretivism argues that there is no universal truth (Fitzgerald and Howcroft, 1998). Using *objectivism* as epistemological stance, the "*assumption*" is that the user's subjective evaluation judgement can be quantified. Crotty stressed that the injection of assumption in the research helps the researcher to unpack the "meaning of research question", "the aim of research methodologies" and "the interpretability of research results" (Crotty, 1998).

There has been a debate whether the subjectivism and objectivism dichotomies of epistemology can be combined (Fitzgerald and Howcroft, 1998). The researchers who do not support mixed epistemology argue that objectivity and subjectivity are regarded as opposites and thus, one can never include aspects of the other (Niehaves, 2005; Becker and Niehaves, 2007). However, if objectivity and subjectivity are regarded as a continuum where they appear on opposite sides then it is possible to be not 100% objective or subjective. Therefore each type of epistemology need another one to make it complete. This is the reason why they are referred as dichotomy of epistemology because of their contradictory but mutually exclusive property. This means they complement each other - given their individual weaknesses and strength.

Theoretical perspective

According to Hirschheim positivism, post-positivism, constructivism, pragmatism and advocacy are examples of theoretical perspectives (Hirschheim, 1992).

The research approach that acknowledges that objectivity does not exclude subjectivity, is called *post-positivism*. The post-positivist stance states that there is uncertainty in any claim of *truth*. This is called the "*uncertainty principle*" which was coined by Heissenberg (Crotty, 1998, p. 29). This is in contrast with the *positivist* perspective, which is based on the assumption that scientific research is certain

(Olivier, 2004). Thus, post-positivism offers its findings as an interpretation of the truth rather than an objective truth.

According to Hirschheim positivism postulates that the universe is comprised of objectively given objects and structures, which exist on their own as empirical entities and they are independent of the observer's appreciation (Hirschheim, 1992). Furthermore, Hirschheim argues that positivism differs from post-positivism in a sense that in positivism the reality is a subjective construction of the mind. Hirschheim (1992) states that constructivism refers to scientific knowledge as being socially constructed and socially sustained. Thus, the significance and meaning of constructivism theoretical perspective can be understood within its social context. Thus, constructivism follows a subjective epistemology.

Pragmatism is a paradigm emerged as philosophical movement located between the objective positivist epistemology and subjective constructivism (Johnson & Onwuegbuzie, 2004). This means through pragmatism the world is viewed between objective positivist epistemology and subjective constructivism.

Advocacy deals with participatory way of creating knowledge (Creswell, 2003).

The research orientation adopted in this study does subscribe to post-positivism epistemology because post-positivism as a paradigm offers a range of methodological choices and researchers could employ multiple or mixed methods. Even though mixed research methods are debatable (Becker and Niehaves, 2007) but according to Gable (1994) mixed research methods increases the robustness of results, research findings can be strengthened through the cross-validation achieved when different types and sources of data embed and are found to be congruent or when explanations are developed to account for divergence.

Methodology

The methodologies chosen for data collection for this research were: the survey and ethnography. Soft Systems Methodology (SSM) was used to manage the analysis of data in a systematic way.

Survey

A survey is a research methodology, which is used to show the correlation between characteristics and is not used to show the cause–effect of phenomena in a population (causality) (Gable, 1994). When using a survey methodology, a sample is chosen among a larger population, depending on the type of theory the researcher wants to prove for the entire population) (Olivier, 2004).

Ethnography

When using this methodology the research participants are studied in their natural setting (Olivier, 2004). It normally involves a field study where the researcher observes the research participants and writes up his/her findings (Myers, 1999; Yin, 2002). According to Yin (2002) the strength of ethnography are:

- addresses the events in real time (i.e. reality)
- covers content of events (i.e. contextual) and
- gives the insight of interpersonal and motives

The above are the reasons why ethnography was selected to be one of the research methodologies.

SSM – Soft Systems Methodology

In this research effort, SSM was used to manage the complex data analysis process. SSM was chosen because of the strength identified by Dorairaja (2008) and Yaghini et al. (2009). According to the Checkland and Scholes, SSM is well suited to solve unstructured, poorly defined and complex problems (Checkland & Scholes, 1990). It is worth noting that SSM allows the researcher to investigate ill-defined (i.e. fuzzy) problem holistically and cyclically. According to Dorairaja (2008), SSM is used when a researcher wants to:

- get holistic view of the situation under consideration;
- obtain the worldviews of various participants involved in the situation;
- identify the conflicting perspectives and issues within the context;
- involve the participants when looking for the solution of the problem under study; and

- engage all participants in cycle of action and learning.

Soft systems methodology differs from hard system methodology (Yaghini et. al., 2009; Dorairaja, 2008). According to Dorairaja the following are the advantages of SSM:

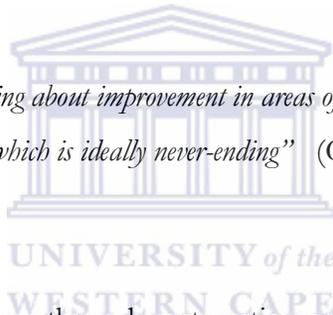
- deals with the dimension referring to people and their perceptions, values and interests
- takes into consideration multiple perceptions of reality
- works within a real world problem perceived from human activity systems
- solves issues pertaining to complex problems (i.e. unstructured problem) and
- allows worldview' to be questioned and debated

What is SSM?

“SSM is a methodology that aims to bring about improvement in areas of social concern by activating in the people involved in the situation a learning cycle which is ideally never-ending” (Checkland & Scholes, 1990, p. 30)

Basic shape of SSM

In the diagram in Figure 7 shows how through systematic operations, a real world problem can be addressed). Initially, the researcher must understand the real-world problem of concern. This helps the researcher to identify choices of purposeful activities that can improve the situation (Checkland & Scholes, 1990). The model of the actions that can be taken and the perceived real-world problem-situation are compared, evaluated and action is taken to improve the situation. This cycle is repeated as often as required. The diagram is a graphical representation of the basic shape of SSM (Checkland & Scholes, 1990, p. 7):



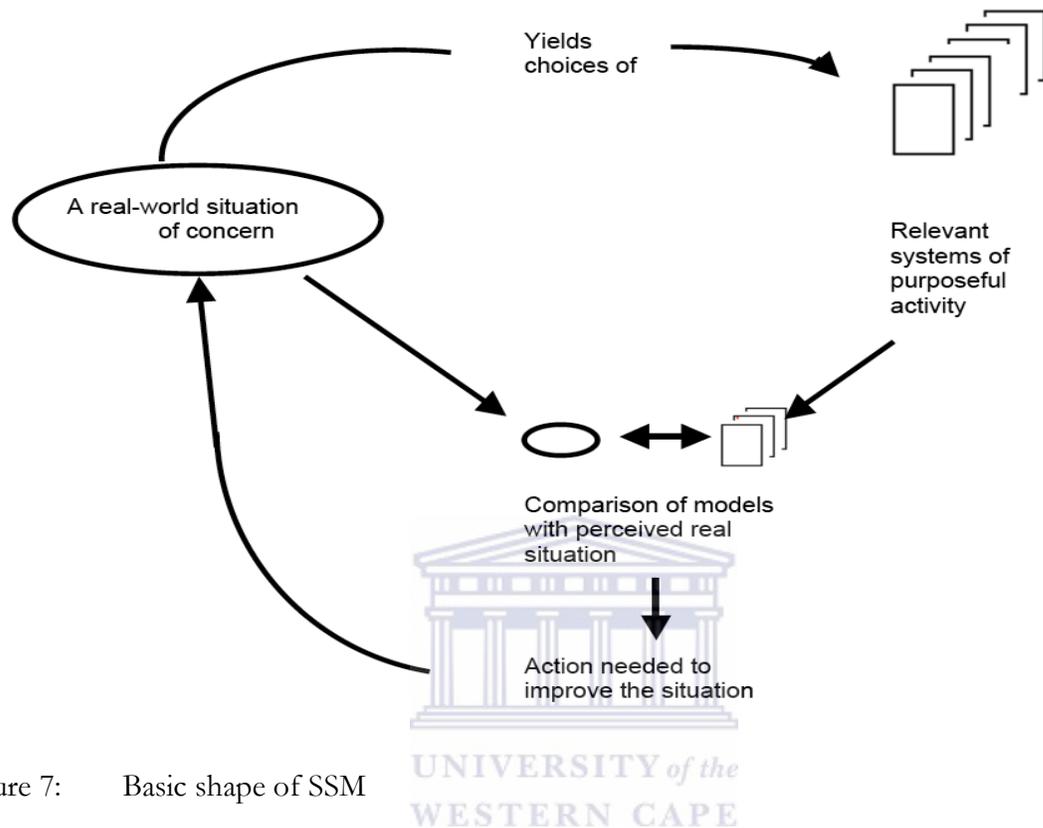


Figure 7: Basic shape of SSM

Depending on how the researcher and participants view the problem (i.e. what their worldview is), a number of relevant purposeful activities can be articulated. The formulation of a “*root definition*” is necessary to express the purposeful action of each relevant system. The aim of the root definition is to transform an input entity into an output entity depending on the model developed to solve the original problem. In order to express the root definition, the researcher must define the CATWOE mnemonic. CATWOE represent the following terms (in relation to the real world problem of concern): Customers, Actors, Transformation process, Worldview, Owners and Environmental constraints (see Table 4).

Customers	the victims or beneficiaries of transformation
Actors	those who would do the transformation
Transformation process	the conversion of input to output
Worldview	the worldview which makes this transformation meaningful in context
Owners	those who could stop transformation
Environmental constraints	elements outside the system which it takes as given

Table 4: The CATWOE mnemonic (Checkland & Scholes, 1990)

The task of researching “*how to develop an algorithm for improving the evaluation of software when judgement is uncertain*” is complex. Thus, the SSM methodology was used. Checkland and Scholes (1990) argue that SSM is: “*an organized way of tackling messy situations in the real world*”.

With reference to Figure 7, the “*real world situation of concern*” was to develop an algorithm to solve the problem of evaluating software with uncertain judgement. This concern yields “*relevant systems of purposeful activity*” namely applying an algorithm to solve the evaluation problem.

Since CATWOE is the foundation of SSM, Table 5 presents how CATWOE was formulated for the problem of evaluating software at OUT.

C	Customers	OUT community (i.e. students and staff)
A	Actors	OUT community members and researcher
T	Transformation process	Modeling the uncertain user evaluation judgement into an output – informed decision
W	Weltanschauung (Worldview)	It is possible to design and develop an algorithm, which reflects the real situation where input must be accepted from both technical and non-technical IT users.
O	Owner	OUT administration.
E	Environment constraints	ICT resources, maintenance and its deployability in developing countries and its impact on quality.

Table 5: The CATWOE mnemonic for the research study

After defining CATWOE, the next step was to express the “root definition” which gave the conceptual model of the “real world situation of concern”.

Thus, the root definition for this thesis is defined as:

Human judgement and reasoning during evaluation of software is subjective but can be quantified. The quality of the evaluation is dependant on the expertise of the evaluators, which makes the evaluation process to be fuzzy. An algorithm, which can handle fuzziness, would be useful to model the evaluation of software by novice and expert IT users.

This root definition represents the transformation process and includes the worldview (W) of researchers, namely, that it is possible to quantify subjective evaluation.

CATWOE was used to identify “*relevant systems of purposeful activity*”. These were formulated in a cyclical manner. During the first cycle of SSM, the AHP algorithm (see Appendix A) was applied as a replication experiment to find if AHP is feasible, applicable and useful for data analysis. This replication was done using Graf and List’s data (Graf & List, 2005) (see Appendix B). In further cycles of the SSM, different algorithms (AHP, Fuzzy AHP (with extent analysis) and group Fuzzy AHP) were developed and evaluated using the data collected from OUT.

Methods

Qualitative methods and quantitative methods were used in the data collection and data analysis respectively. The qualitative method entailed a case study where participant observation was used and focus group interviews were conducted. The quantitative methods entailed the use of questionnaires to collect data and the different algorithms such as AHP, Fuzzy AHP and Group Fuzzy AHP to analyse the data.

Why combine research methods?

According to Olivier, qualitative research methods provide useful insights when it is combined with quantitative research, but are seldom used by researchers in the field of information technology and Computer Science (Olivier, 2004). However, in a recent publication Blake (2009) argues that qualitative research should be considered in the field of Computer Science:

Computer Science is a synthetic discipline and has always been concerned with design: this would be our “Professional Doctorate” based on qualitative research, case studies, contextual enquiry, and the like. (Blake, 2009, p. 110)

Using a combined research approach, the disadvantages of the methods used can be minimised and their advantages maximised (see Table 6).

What is a combined research approach?

Dubé & Paré (2003) argue that a

“multi-method approach to research involves several data collection techniques, such as interviews and documentation, organized to provide multiple but dissimilar data sets regarding the same phenomena” (Dubé & Paré, 2003, p. 615)

Why to use a combined research approach?

A major strength of case study data collection is the opportunity to use many different sources of evidence to provide a richer picture of the events and/or issues than would any single method (Yin, 1994).

When a combined research approach should be used?

Furthermore, Dubé & Paré (2003) argue that mixed methods is used when researchers want to avoid

“from being carried away by vivid, but false, impressions in qualitative data, and it can bolster findings when it corroborates those findings from qualitative evidence” (Dubé & Paré, 2003, p. 615)

How a combined research approach should be undertaken?

Mixed methods can be used in sequentially, parallel, equivalent, dominant or multi-level (Leech & Onwuegbuzie, 2005).

Different from what others have used (Leech & Onwuegbuzie, 2005), in this thesis a combined research approach in parallel (i.e. simultaneous) and in multi-level (see Table 6) was used.

Methodology	Method applied	Advantages	Disadvantages
Survey	Case study	It allows specific cases to be examined in detail within a context (Yin, 1994) It allows both qualitative and quantitative data to be collected (Yin, 2003)	Case studies may shift the focus and thus make deductions unreliable (Yin, 2003) It is difficult to report all the information (Gable, 1994)
	Focus group interview	Interviews are conducted in an interactive group setting. It allows the researcher to study people in a more natural setting than in a one-to-one interview (Myers and Newman, 2007)	The researcher, who is a facilitator, can influence the outcome of the research. Time consuming (Johnson and Onwuegbuzie, 2004)
	Questionnaire	Provide data to a researcher from all users (Gable, 1994)	Not as flexible as an interview where researcher can discover new problems and thus derive new insights (Gable, 1994)
Ethnography	Participant observation	The researcher is part of the group which allows the researcher to collect useful data from the experience obtained while observing the participants (Johnson and Onwuegbuzie, 2004)	The researcher may influence the results Lack of time, lack of trust, and biasness (Myers, 1999)
SSM	Data analysis using methods such as statistical analysis or algorithms such as AHP, Fuzzy AHP, Group Fuzzy AHP	Some methods are considered to be more objective and verifiable (Yaghini et al., 2009) While some methods allow the use of both quantitative and qualitative data (Dorairaja, 2008) In addition, some methods have the capability of handling uncertainty of user evaluation judgement	In AHP, the subjectivity associated with the quantitative data is ignored. In cases where subjectivity can be handled by the algorithm, careful consideration must be given to the modelling of the algorithm because the change of scale, membership function and the inference method can result in a different outcome

Table 6: Pros and cons of proposed research methods

The selection of appropriate research methods for conducting a case study is normally based on the research questions, time constraint and resources (Olivier, 2004). A major strength of case study data collection is the opportunity to use many different sources of evidence to provide a richer picture of the events and/or issues than would any single method (Yin, 1994). Crotty (1998) categorise case study as a method under survey, which is a methodology. Gable (1994) argued that case study is a method under qualitative methodology while survey is a method under quantitative methodology.

In this study, qualitative evaluation methods were used during data collection because of its usefulness in providing detailed information and rich description of phenomena in a short time. The analysis of qualitative data was done by finding patterns in the collected data, as suggested by Seidel (Seidel, 1998). The collected data from the interviews notes, existing documents and field observations were coded. Thereafter the coded data helped to draw conclusions related to research questions. The quantitative

data were collected using questionnaires and were analysed with different algorithms to evaluate which algorithm's outcome is more useful and effective. This combined approach helped to harness the advantages of both qualitative and quantitative methods.

The advantages of mixed methods are well documented by Johnson and Onwuegbuzie (2004). According to Johnson and Onwuegbuzie (2004) the following are the advantages of mixed methods:

- Words, pictures, and narrative can be used to add meaning and precision to numbers and vice versa.
- Includes the strength of both quantitative and qualitative methods – as it is reviewed in Chapter 2.
- Does not confine a researcher to a single research method thus, allows a range of research questions and answers.
- since it allows multiple research methods therefore the strength of one method can be used to overcome the weakness of the other methods. The conclusion can be provided through convergence of findings from a chain of logical evidence.
- Reveals some insights and understanding of phenomena that might be either missed or difficult to find when only a single method is used.
- Increases the credibility and the generalizability of the research results and
- the use of mixed methods from qualitative and quantitative research produces comprehensive knowledge necessary to inform theory and practice.

An algorithm that can handle both qualitative and quantitative evaluation judgements, will allow the inclusion of all users' views in the evaluation process. In addition, a combination of methodologies will allow broader consultation when evaluating software in a particular environment.

When using a combination of methodologies it is important to document the research carefully to ensure validity of the data and the results. In this research, the following basic steps for software product evaluation were used: planning the evaluation process; identifying criteria for evaluation; collecting data; and finally analysing the data using the developed algorithms (Comella-Dorda et al., 2002).

Actual study

Twenty-three participants from OUT, who had experience in both ATutor and Moodle, were selected by means of purposive sampling (Van Vuuren & Maree, 1999). The purposive sampling procedure was used because of the limited number of respondents (or participants). Even though a large number of knowledgeable users are not mandatory criteria for providing data to the developed algorithm, the number of staff and students (i.e. population) who could be used as sample in this study were limited. The participants comprised of 20 Second year BSc Information Communication Technology students and 3 IT lecturers (who are, also, involved in systems administration and maintenance). The participants were given a consent form to complete before taking part in the research to adhere to the ethical standards set for this study (see Appendix C).

Instrument design

To determine what type of questions to ask in the questionnaire and how to ask it, a questionnaire (as instrument for data collection) was developed and refined in a pilot study.

For the three main characteristics (usability, maintainability, and deploy-ability), sub-characteristics were identified as proposed in the literature (Boehm et al., 1976, p. 595; Cavano and McCall, 1978, p. 136; IEEE, 1998; ISO, 2001) (see Appendix D and E). For each of these sub-characteristics, the researcher decided on attributes that best describe the selected sub-characteristics. This was done before the pilot testing. The selected attributes were combined with attributes identified in the literature (Coleman et al., 1994; ISO, 2001; Bertoa and Vallecillo, 2002, p. 65). The combined characteristics, sub-characteristics, and attributes were used in the formulation of a questionnaire; some of the questions were adopted from a Software Usability Measurement Inventory (SUMI) questionnaire (Ryu, 2005, p. 192). Attributes that were duplicated, were removed from the questionnaire. The aim was to come up with a list of representative attributes, agreed to by all participants (see Appendix D).

Open-ended questions or probes were designed for use with the focus group. These probes covered some aspects not included in the questionnaire. The questions in all the instruments used for data collection; were short, precise and to the point.

Pilot testing

The pilot study was conducted at UWC using test data. The applicability and validity of the questionnaire for the collection of quantitative data as well as the appropriateness of the open-ended questions for the collection of qualitative data were evaluated and discussed with a quantitative analysis expert from the United States.

The usefulness of AHP, as an analysis tool, was determined in this pilot study. The instruments for data collection and analysis were thus ready to be used in the actual case study.

The following diagram (Figure 9) represents a snapshot of the aim of the data collection.

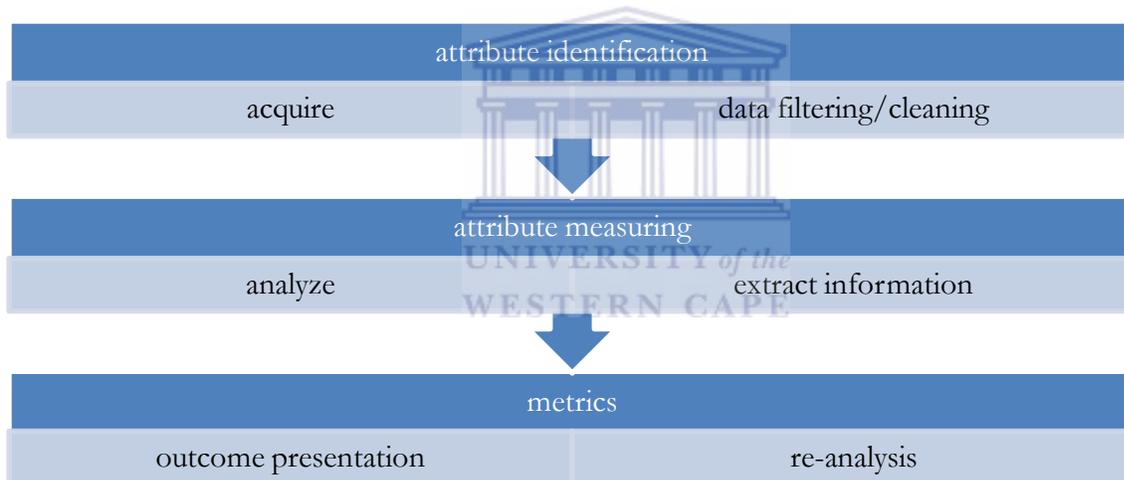


Figure 8: Attributes measurement

Data collection

Instruments used to collect data

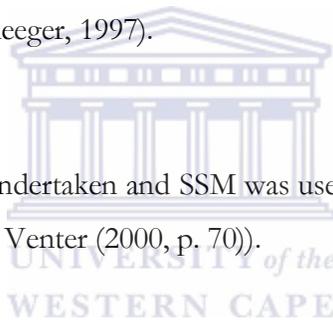
Data was collected between November 2007 to February 2008. Participants were asked to complete the questionnaire and some of the participants were interviewed using probes (see Appendix F). Focus groups helped to obtain a wish-list of weights for the characteristics and sub-characteristics of maintainability, usability and deployability. In addition, the researcher took field notes (i.e. participative observation) during the data collection process. This provided rich data, which informed the research process.

Data preparation

The respondents were required to indicate the level of importance of the software quality characteristics of usability, maintainability and deployability; and their respective sub-characteristics and attributes. The answers the respondent could choose from were: extremely important, very important, important, of little importance; or, not important (see Appendix F). Different scales were used to convert the user verbal description (i.e. views) (about the comparison of software according to software quality) and responses to the questionnaire into a numerical value. After a number of trials using different types of scales (such as nominal, interval, ordinal and ratio scales), the Saaty scale (Saaty, 1980) was chosen to translate the data into a format that could be used in AHP computations. Mapping software quality phenomena to a numerical value is accepted in measurement theory and software measurement (Fenton & Pfleeger, 1997).

How data was analysed

Several cycles of data analysis were undertaken and SSM was used to manage this cyclical data analysis process (see Figure 9 – adapted from Venter (2000, p. 70)).



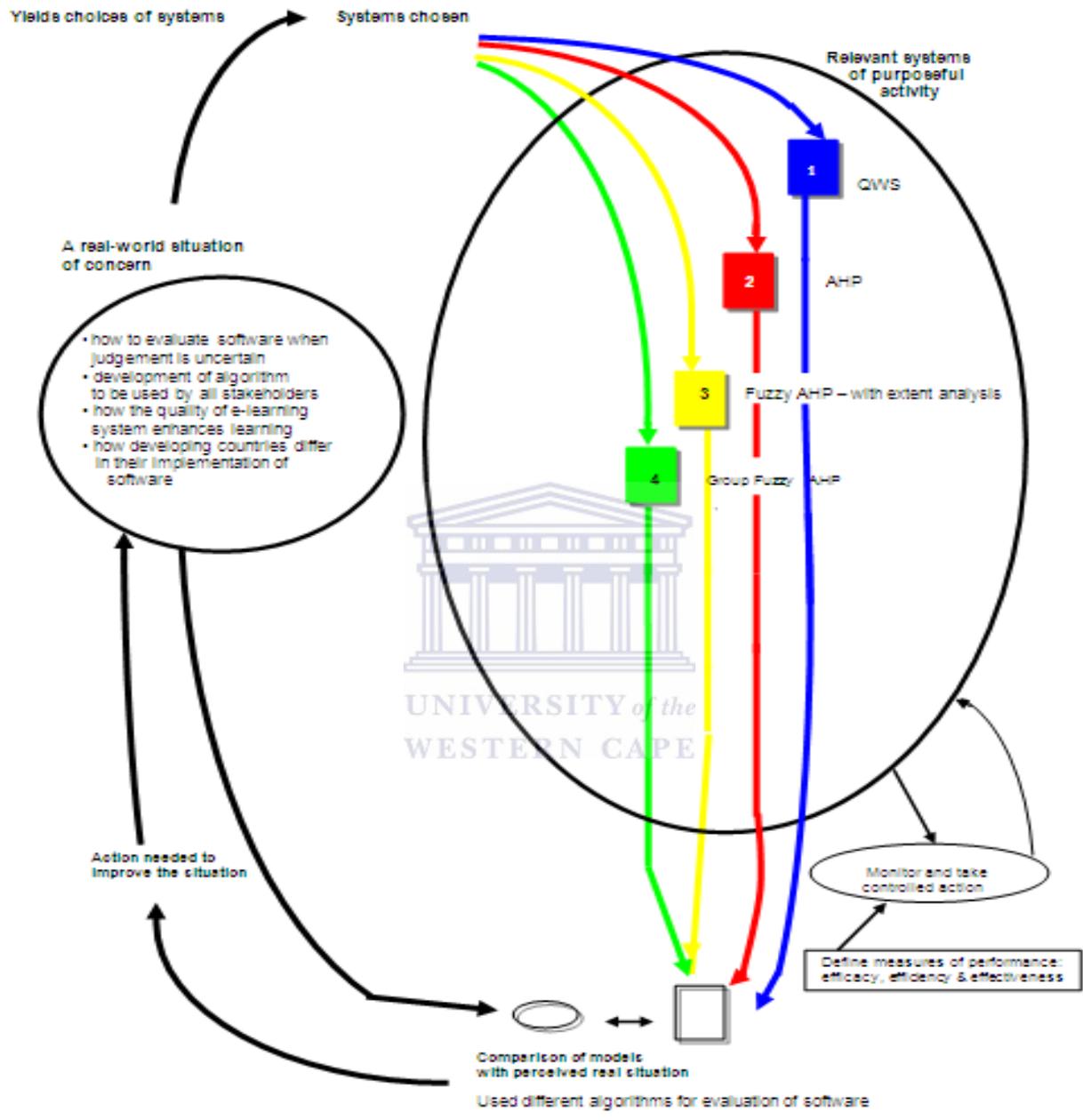


Figure 9: How the SSM was used to manage the research

The strength of the SSM as mentioned by Dorairaja (2008) fits well the unique characteristics of the problem identified in this study, because:

- It helps to achieve the holistic view of the situation under investigation
- The worldview of various participants involved in the situation can be obtained easily
- It helps to identify the conflicting perspectives and issues within the system
- Allows the involvement of all participants in a cycle of action and learning
- Gives the participant a control over the situation and
- Helps to develop relevant system

First cycle of SSM: Replication experiment

The aim of this investigation (see 1 in Figure 9) was to determine whether the results of an empirical evaluation could be confirmed using a different evaluation algorithm, namely AHP. In this experiment, the QWS algorithm, used by the researchers Graf and List (2005) to evaluate several free and open source e-learning software platforms, were studied and replicated with AHP (as shown in detail in Appendix B).

It was found that the ranking of the e-learning platforms, when using the AHP, differs slightly from the outcome of QWS. However, with AHP it is possible to determine the consistency of the evaluation judgements of evaluators (see Appendix G). Both QWS and AHP are useful to evaluate software but it is easier to rank software using AHP.

Second cycle of SSM: Implementation of AHP

The objective of this experiment (see 2 in Figure 9) was to determine whether AHP would be suitable for the evaluation of software by evaluators with little IT experience. It was found that AHP would be useful to determine evaluation preferences by a group of users, however; its weakness is that it gives unreliable results when user judgement are uncertainty. Thus, in order to deal with uncertainty during evaluation there is a need for an algorithm, which can cope with this reality.

Third cycle of SSM: Implementation of Fuzzy AHP (with extent analysis)

The objective of this experiment (see 3 in Figure 9) was to extend the multi-criteria evaluation algorithm, AHP, which is applicable for managing “certain” evaluation judgements, and to imitate the way humans’ reason and judge. Human reasoning and judgement during the evaluation of software is

subjective and could be said to be “uncertain”. Thus, algorithms that could deal with the uncertainty of human judgements would be an improvement on AHP. It was found that fuzzy logic combined with the AHP algorithm, could compensate for the weakness of AHP. The algorithm was developed and implemented (see Appendix H) and the outcome of the Fuzzy AHP (with extent analysis) and the outcome of AHP, were compared. The developed Fuzzy AHP (with extent analysis) does not discard priority weights (of the characteristics, sub-characteristics and attributes) with low numerical values (i.e. ratings).

Although the results were almost similar, it was found that Group Fuzzy AHP would be an improvement on Fuzzy AHP (with extent analysis) since it has less computations. Apart from these mentioned advantages, Group Fuzzy AHP has those advantages of AHP (Sanga & Venter, 2009). The reason for this is that Group Fuzzy AHP is based on principles of AHP. Thus the advantages of Group Fuzzy AHP includes: it is flexible, it integrates deductive approaches, it acknowledges interdependence of elements of software systems, it has a hierarchical structure, measures intangibles, tracks logical consistency, gives an overall estimation, consider relative priorities and improve judgements.

Fourth cycle of SSM: Implementation of Group Fuzzy AHP

In the fourth cycle, Group Fuzzy AHP (see 4 in Figure 9) was developed and applied to see if this algorithm could address the concerns of the researcher namely “how to evaluate software efficiently when a group of users does the evaluation and user judgement is uncertain”—for full detail, see Appendix I.

Further cycles: Proposed future research

Further cycles could still be embarked upon to address the weakness of Group Fuzzy AHP. This can be done by comparing it with other algorithms using: (i) consistency checking, (ii) comparing the correctness of the outcome, (iii) comparison of the time efficiency, (or time complexity) of the algorithms, (iv) comparison of space efficiency, (v) determining its simplicity, and (vi) generality of the algorithms (deduced from suggestions by Levitin (Levitin, 2003) and Saaty and (Saaty, 1980)).

Limitation of the study's methods and methodologies

The limitations of the data collection methods used are:

- (i) The number of respondents (or participants) - Out of the staff and students (i.e. population) who could be used as sample in this study only a few had the necessary knowledge to participate in the study.
- (ii) These participants' past experience in the use of the FOSS e-learning systems were limited.
- (iii) Only one University in Tanzania met the criteria that were necessary to conduct the case study.
- (iv) Data collection methods were inherently susceptible to bias.

But, according to Kunda, any research study will have limitations (Kunda, 2001). The strategies to control the limitations of the study's methods and methodologies were:

- (i) A case study was used since the case study is useful when there are only a few participants.
- (ii) More than one method was used during data collection. Triangulation helped to extend the limited scope of the study.
- (iii) The use of focus group interviews helped the researcher to uncover information, which would not have been possible with other data collection methods.
- (iv) The AHP algorithm has the capability of checking the consistency of an evaluator's judgement. This was used as a technique to control which sets of data to use. If there was inconsistency of the evaluation judgement then either the evaluation process or calculation of consistency checking should be repeated.
- (v) The cyclic analysis of the data, using different algorithms, helped to get more clues about the research problem which could not be revealed if only single algorithm was implemented.
- (vi) Data collection methods were inherently susceptible to bias (from both participant and researcher).

Conducting a pilot study helped to understand the problem and hence the instruments for data collection methods were rectified before being used in the actual study.

- (vii) Fuzzy logic addressed the problem that the algorithm coned only accepts crisp values.

Conclusion

In this Chapter, the research design and methods used, were presented.

The research approach, its epistemology, theoretical perspective, methodology and methods were discussed. The instruments for data collection as well as data analysis were explained. Furthermore, the methodology, SSM, used to manage the data analysis process, was introduced and explained in detail in this chapter. In the next chapter, the results will be presented.



CHAPTER 4

RESEARCH RESULTS

Today's scientists have substituted mathematics for experiments, and they wander off through equation after equation, and eventually build a structure, which has no relation to reality.

Nikola Tesla [1856-1943]

Introduction

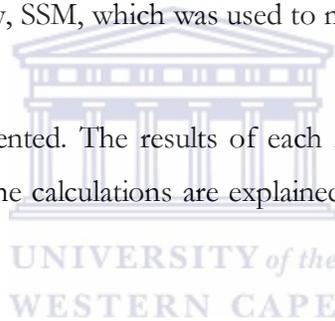
In the previous chapter, the design of the research and the research approach: the epistemology, theoretical perspective, methodology and methods, were presented. The instruments for data collection as well as the methodology, SSM, which was used to manage the data analysis process, were introduced and explained.

In this chapter, the results are presented. The results of each intervention will be discussed in this chapter but most of the steps and the calculations are explained in several appendices (Appendices J — K).

The problem revisited

First cycle of SSM: Replication experiment

The data of Graf and List (2001), the evaluation of 36 FOSS e-learning systems using QWS, was used as input for the AHP algorithm (see in Figure 11). The data used in the Graf and List's QWS method of analysis was qualitative.



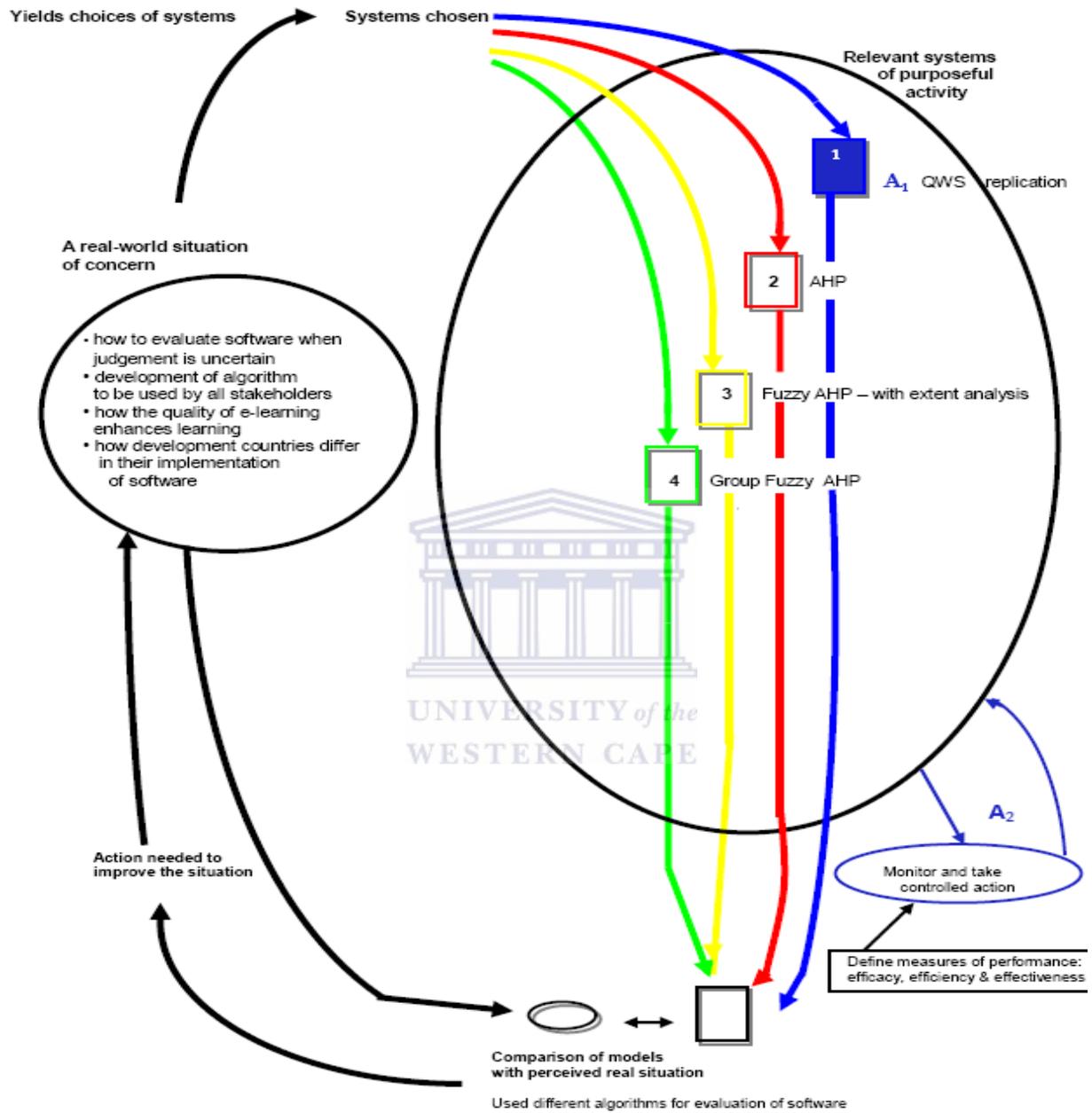


Figure 10: The first cycle of SSM

The first cycle of SSM, depicts the outcome when the data of QWS was used as input to the AHP algorithm.

Outcome of the QWS replication in the first cycle of SSM

The replication of the QWS experiment with AHP, is explained in detail in Appendix B. The outcome of this replication, the execution of the first cycle of SSM, is presented in Table 7.

e-learning system	Total Priority using AHP	Ranking by AHP	Ranking by QWS (Graf & List, 2005)
ATutor	0.8815	5	4
Dokeos	0.9235	3	3
DotLRN	0.8038	8	5
ILIAS	1.0181	2	2
LON-CAPA	0.8877	4	4
Moodle	1.0649	1	1
OpenUSS	0.7935	9	4
Sakai	0.8130	7	5
Spaghettilearning	0.8138	6	4

Table 7: Outcome from first cycle of SSM

Table 7 shows the overall ranking of the e-learning systems after applying AHP. This ranking differs slightly from the results obtained with QWS. The priority vector and total priority as depicted in Appendix B, Moodle is considered the best option, with ILIAS the second and Dokeos in the third positions in the overall evaluation. This ranking is similar to the evaluation of Graf and List according to the specified criteria. Saaty (1980) indicated that evaluation judgements are consistent if the consistency ratio (CR) is less than 0.1 or the consistency index (CI) equals 0. From Appendix B, it can be seen that the evaluation of the e-learning systems (with respect to all categories) was consistent since the consistency indices were equal to 0.

Validation of the results

The validation of the QWS experiment was done as the first cycle of the SSM managed analysis to monitor the execution of the analysis with AHP and its outcome, letter A_2 in Figure 10. The validation of results in a replication investigation is important as it verifies the results and the validity of the original empirical evaluation (Rodríguez et al., 2006). The validation was done using a consistency checking technique described by Teknomo (2006) and explained in the Appendix G. The result of the

replication is shown in Appendix B and the results of the consistency checking is shown in Appendix J.

The validation of the results was confirmed by comparing the overall evaluation ranking results (see Table 7). It is possible that the difference of the QWS results compared to the AHP results is due to the choice of a scale which was used to assign values to the symbols of the Graf and List data (see Table 28). Another reason might be because the instances of the whole former empirical evaluation were not reproduced. According to Shull et al. (2008), a replication can be considered to be successful even if the results differ from the original investigation.

The validation results confirmed that both methods, AHP and QWS, are valid. But, the results did not agree with the principle of external validity. The principle of external validity concerns with the extent (i.e. degree at which) the results from the study can be true for other cases (i.e. generalized) (Schneidewind, 1992; Chin, 2001). This means that the results of Graf and List cannot be generalized in all environments.

Second cycle of SSM: Implementation of AHP

In the second cycle of the SSM managed process, AHP (see Figure 11) was used to analyse data collected at OUT (see Appendix K).

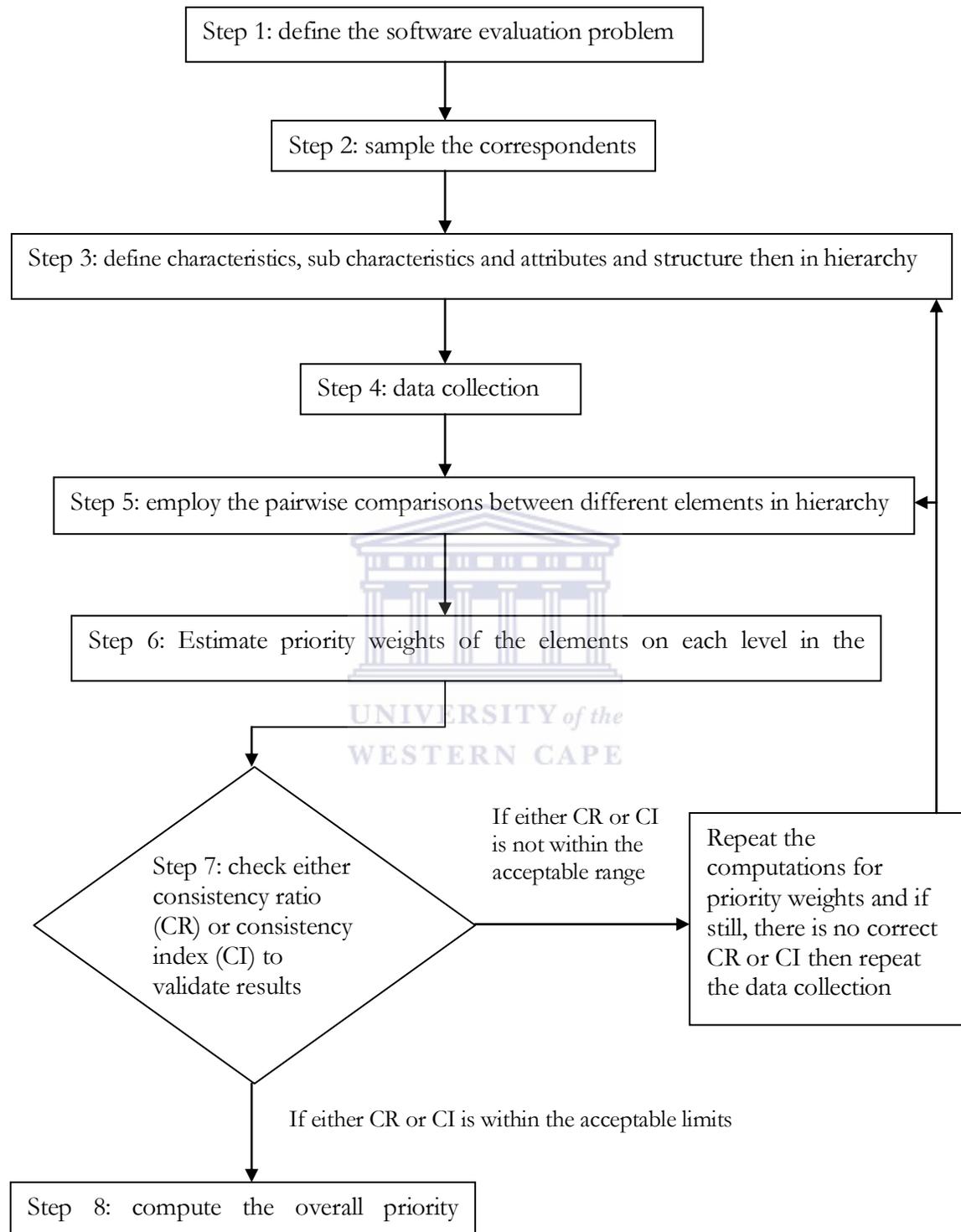


Figure 11: Flowchart of AHP

The evaluation of two e-learning software systems (ATutor and Moodle) being used at OUT, was collected from technical and non-technical IT users. This section presents the outcome of the data analysis using lesson learnt from the first cycle of SSM.

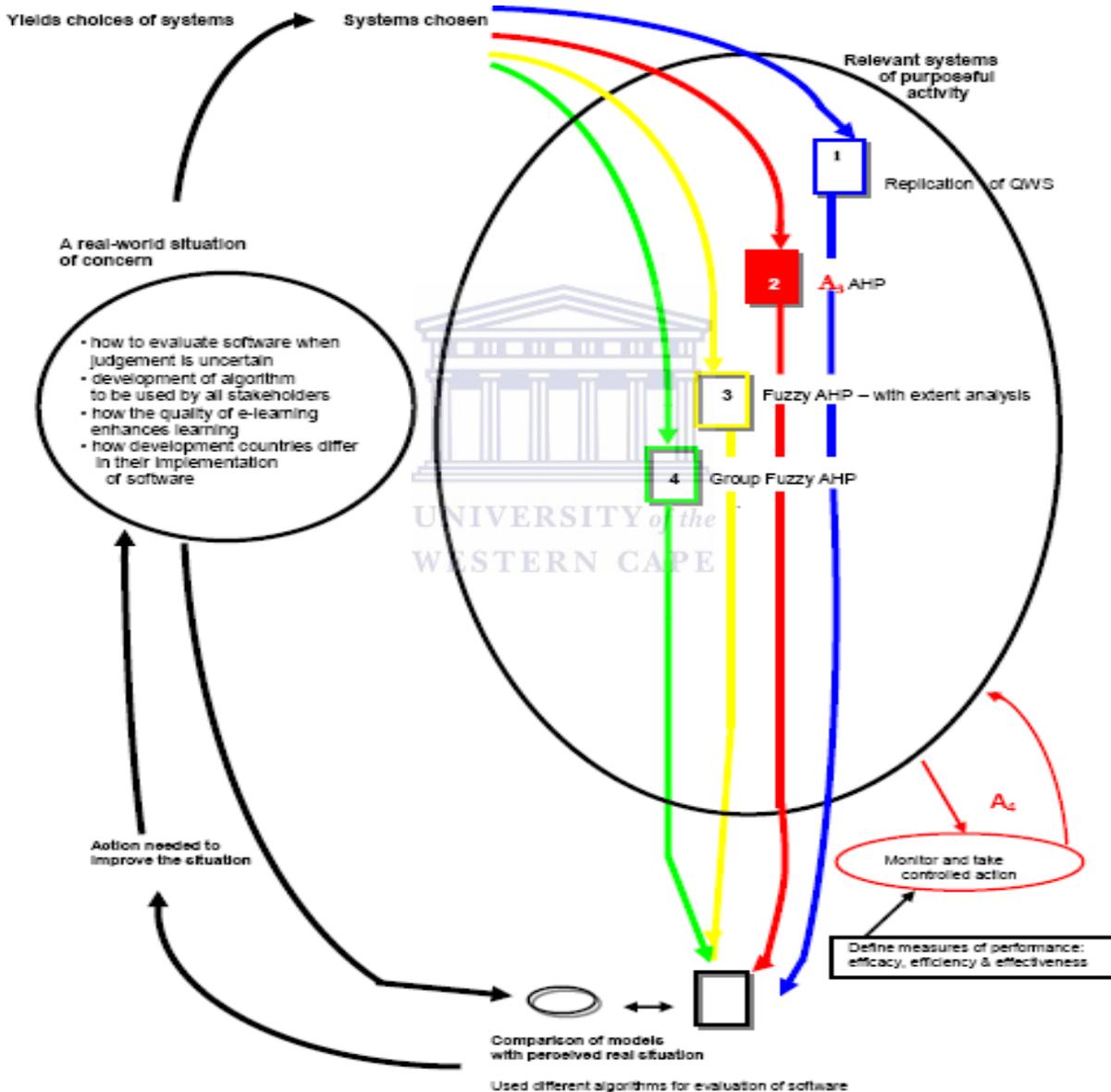


Figure 12: The second cycle of SSM

Thus in order to confirm the usefulness of the AHP the second cycle of the SSM was executed. This is shown as the second cycle of the SSM (see Figure 12) and is indicated by A_3 .

Data modeling and analysis

All the questions in the questionnaire required categorical responses. Numerical values were arbitrarily allocated to these categorical responses (Stevens, 1946, p. 679). Mapping software quality phenomena to a numerical value is accepted in measurement theory and software measurement (Fenton & Pfleeger, 1997).

According to Turban (1993, p. 221) defining intensities (measures) the way it was done in this research, is the best approach to avoid comparison ambiguity associated with large number of criteria and alternatives.

There are two methods for computing the combined group decisions in AHP: (i) using either the arithmetic mean or the geometric mean of individual respondents' comparison judgements (i.e. opinions) and (ii) using either the arithmetic mean or the geometric mean of the individually calculated priorities (Forman & Peniwati, 1998). The first method is used if the group of the respondents wants to act as a unit while the second method is used if the group wants to act as a combination of individuals. In this thesis, the first method was used. It is worth noting that during the analysis of the qualitative data the principles of interpretive field studies were adhered to (see Appendix L).

Outcome of AHP: the second cycle of SSM

The results derived when applying AHP, gives the total priority for each e-learning system (see Appendix K for the detailed explanation). Table 8 shows the ranked results of the two e-learning systems considered. Moodle ranks above ATutor for each of the scales used.

E-learning system	ATutor	Moodle
Total priority	0.469	0.531

Table 8: Outcome from the second cycle of SSM

Finally, the consistency indices for each pairwise comparison were computed and were equal to approximately zero which means that all the pairwise comparison judgements were consistent (Saaty, 1980). (The procedure used in Appendix J was used here as well). It concurs with Moses and Farrow (2008, p. 286) who argue that the best approach to validate results for subjective judgements is to check consistency. It is a means to monitor the effectiveness of the analysis and its outcome (see Figure 12 letter A₄).

Third cycle of SSM: Implementation of Fuzzy AHP (with extent analysis)

The AHP algorithm is effective if the user evaluates the software by giving exact or crisp values (i.e. discrete/defined/certainty) to questions asked about the quality of software. However, it is not always possible to quantify responses. The post positivist approach incorporates qualitative data where as the positivist approach is grounded in assuming that scientific phenomena can be observed and measured (i.e. empirical). The post positivist approach accepts that most scientific observations are uncertain (or fuzzy). It was thus felt that it is important to incorporate fuzzy logic in AHP to address the uncertainty of user judgement (see Appendix H).

Algorithm for the evaluation of software when judgement is not well defined

In this section, the results from the implementation of two algorithms will be presented. It is part of the third and fourth cycles of SSM expressing Fuzzy AHP (with extent analysis) and Group Fuzzy AHP respectively. Even though usability, maintainability and deployability has been identified as the characteristics that are most important for software implementation in a developing country, for simplicity, only the usability characteristic will be demonstrated.

Usability has been depicted in a hierarchical structure (see Figure 13) to give an overview of the multi-criteria evaluation problem (the list and the detailed description of characteristics, sub-characteristics and attributes of usability can be found in Appendix D and Appendix E respectively).

The first level consists of the goal of the evaluation, and represents the usability characteristic/criteria; the second level consists of sub-characteristics/sub-criteria of usability; the third level contains the numbers of the questions asked about the attributes of each sub-characteristic of usability (see the questionnaire, Appendix F). The last level, in the figure below (Figure 13), represents the two alternative software packages (i.e. ATutor and Moodle).

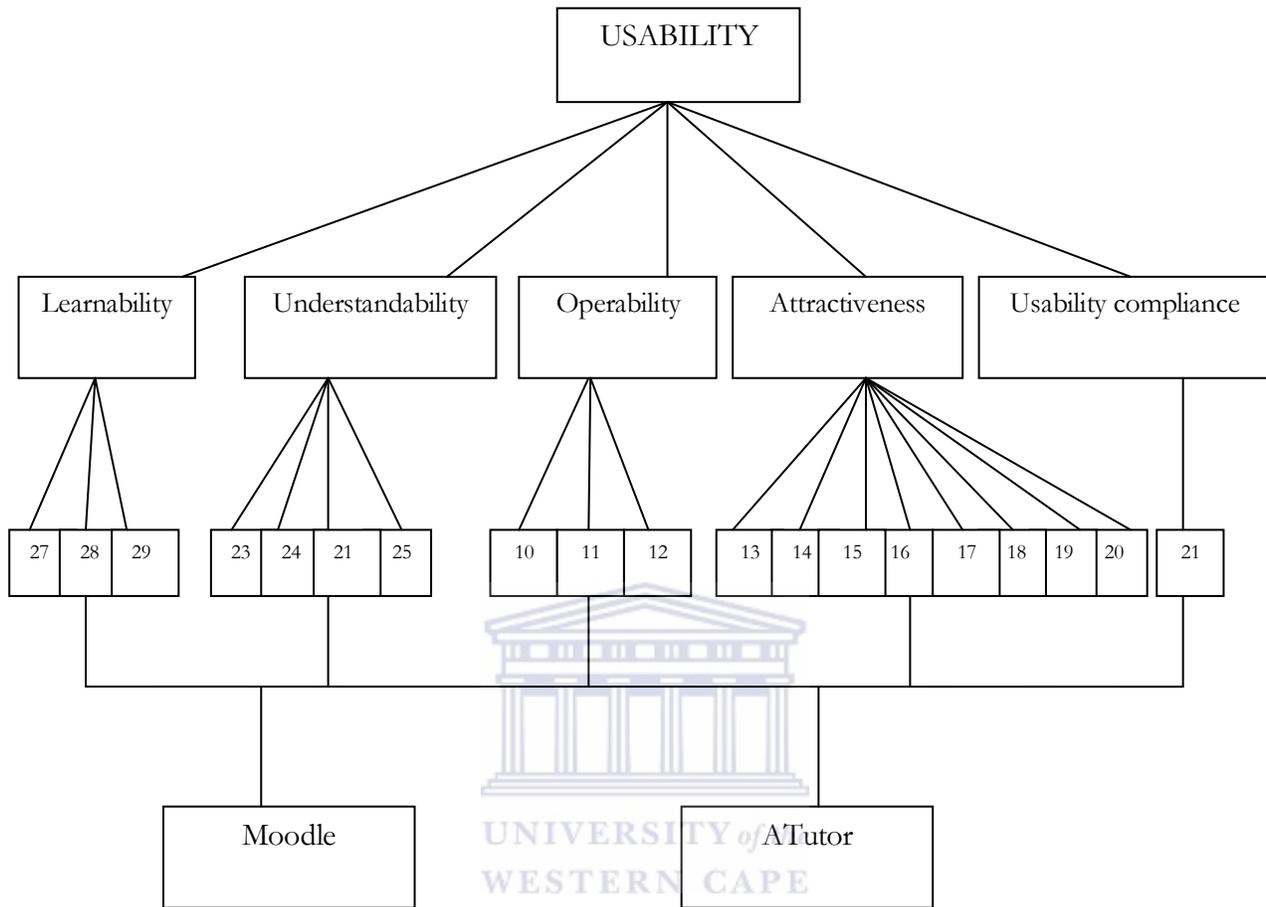


Figure 13: Hierarchical structure of usability characteristic

The results are generated from three different algorithms. First the results from AHP will be presented then the results for the modified Fuzzy AHP (with extent analysis) and after that the results for Group Fuzzy AHP. At the end, the comparisons between these three will be discussed. The discussion is given as the means to evaluate the algorithms. This evaluation is the monitoring and taking controlled action of the relevant system of activities (see A_6) as depicted in Figure 15.

Fuzzy AHP

As stated before, even though QWS and AHP can solve the problem of evaluating software, its results are unreliable when used for evaluations when there is fuzziness and uncertainty. Thus in such a situation, the fuzzy AHP algorithm would be more appropriate.

There are many types of fuzzy AHP algorithms such as: Fuzzy AHP (with extent analysis), Fuzzy goal programming and fuzzy preference programming (Chang, 1996; Wang & Fu, 1997; Zhu et al., 1999; Bozdag et al., 2003; Mikhailov, 2003; Wang & Chin, 2008). The general flow chart for the Fuzzy AHP is presented on Appendix H. It consists of the following steps: fuzzy problem structuring, deriving priorities from fuzzy comparison matrices, computing fuzzy prioritization and computing global priorities. The method used to compute fuzzy prioritization, determines extension of the name of the Fuzzy AHP algorithm. For example, when the extent analysis method is used then the algorithm is named “Fuzzy AHP (with extent analysis)”.

The following section presents the flow chart of Fuzzy AHP (with extent analysis) and Group Fuzzy AHP depicted in Figure 14. The steps for Fuzzy AHP (with extent analysis) starts at box with letter “A” while that of the Group Fuzzy AHP starts at box with letter “B”. The Fuzzy AHP (with extent analysis) computes the prioritization by first fuzzifying the crisp values obtained from the linguistic variables provided by evaluators (i.e. users). Depending on the type of fuzzy scale to be used in fuzzification, the pairwise comparison matrices (PCM) are computed. If the algorithm to be used is Fuzzy AHP (with extent analysis) then minimum (min), mean and maximum (max) operators are applied to obtain the comprehensive Fuzzy PCM. After that, the fuzzy extent analysis is used in assessing the priorities weight of each attributes of software quality. In order to obtain the composite priorities the aggregation of weight is done before defuzzification to obtain the crisp value. The defuzzification process is done by using either total integral or dominance or geometric mean method (Chen & Klein, 1997).

But for the case of Group Fuzzy AHP the procedures differ from that of Fuzzy AHP (with extent analysis). Using Group Fuzzy AHP, after fuzzifying the crisp pairwise comparison matrices then the priority weight is computed. The computation is done using the concept of group AHP. The defuzzification process is done similar to Fuzzy AHP (with extent analysis).

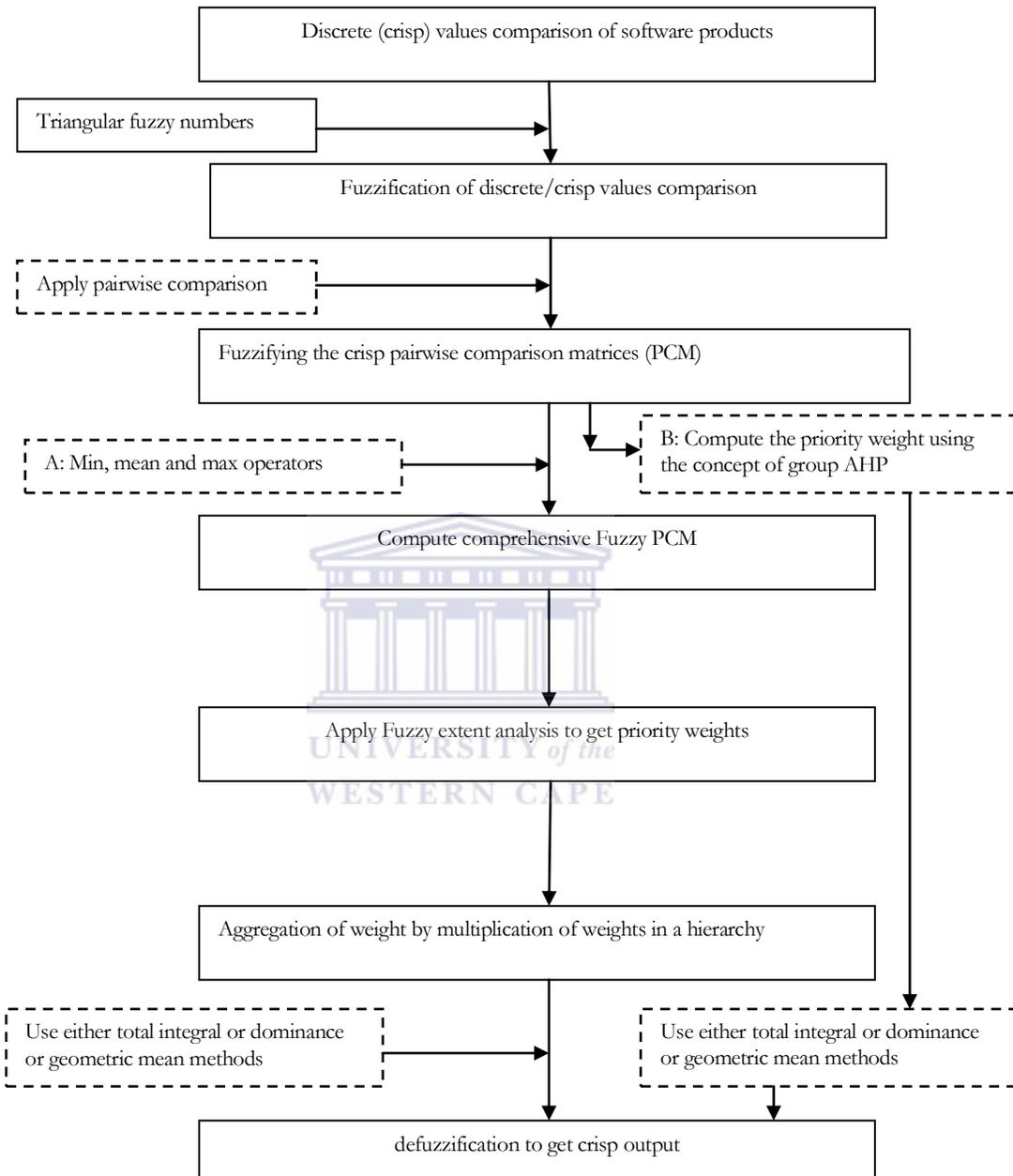


Figure 14: Flowchart of the Fuzzy AHP (with extent analysis) and Group Fuzzy AHP

The yellow line and square in Figure 15 presents the third cycle of SSM using Fuzzy AHP (with extent analysis) and is indicated by the letter A_5 .

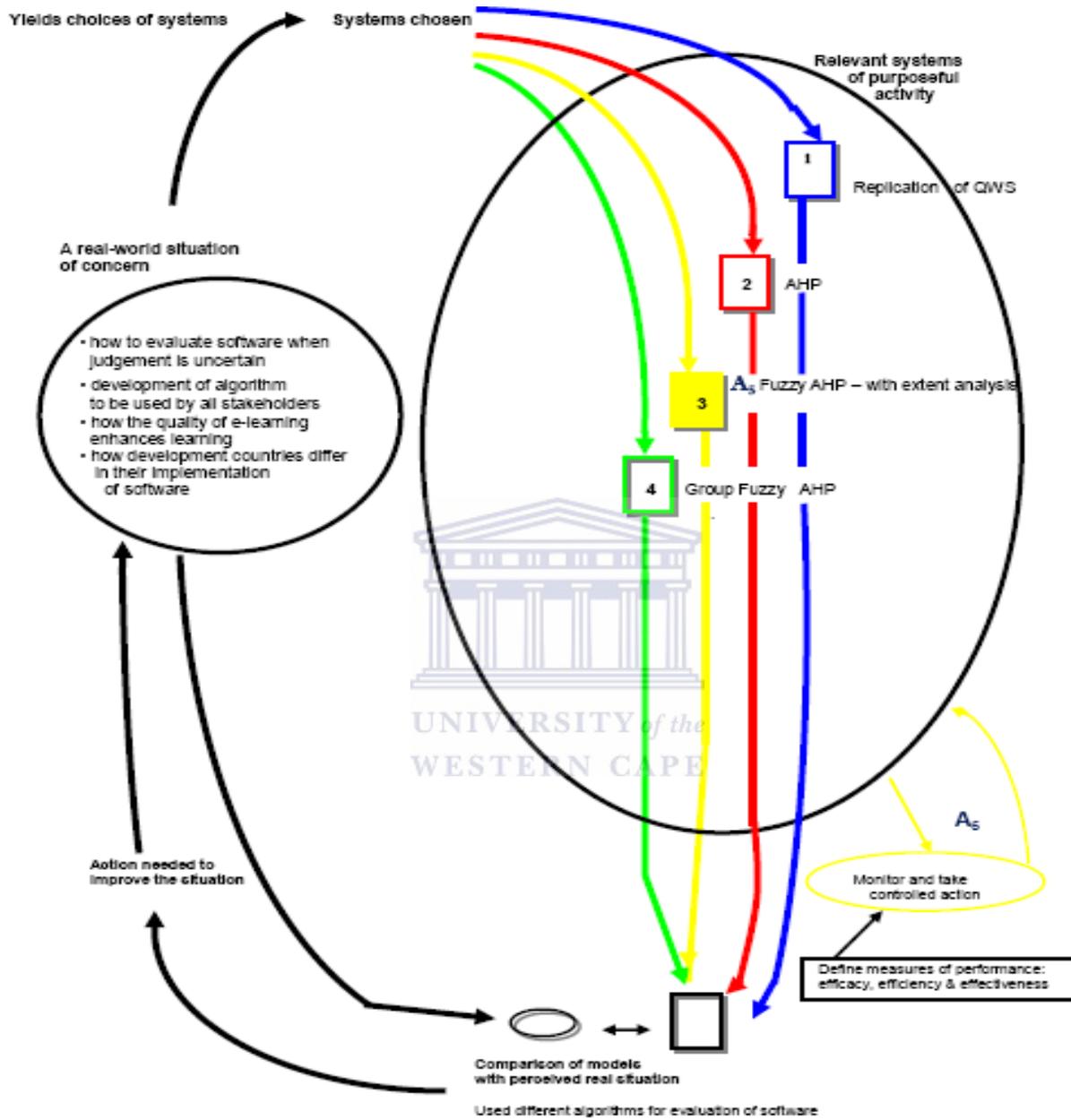


Figure 15: The third cycle of SSM

The outcome of the execution of the Fuzzy AHP (with extent analysis) in the third cycle is shown in Table 9.

Outcome of the Fuzzy AHP (with extent analysis) from the third cycle of SSM

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.325	27	0.340	0.111	0.619	0.381
		28	0.402	0.131	0.798	0.201
		29	0.256	0.083	0.710	0.290
understandability	0.199	23	0.298	0.059	0.824	0.175
		24	0.198	0.040	0.155	0.844
		21	0.130	0.026	0.290	0.710
		25	0.373	0.074	0.381	0.619
operability	0.119	10	0.588	0.070	0.619	0.381
		11	0.080	0.010	0.867	0.131
		12	0.331	0.039	0.131	0.867
attractiveness	0.101	13	0.187	0.019	0.201	0.798
		14	0.105	0.011	0.237	0.763
		15	0.151	0.015	0.619	0.381
		16	0.107	0.011	0.201	0.798
		17	0.059	0.006	0.867	0.131
		18	0.136	0.014	0.155	0.844
		19	0.149	0.015	0.381	0.619
		20	0.106	0.011	0.798	0.201
usability compliance	0.253	22	1.000	0.253	0.139	0.860
				Priority	0.454	0.541
				Total	0.995	
				error	0.005	

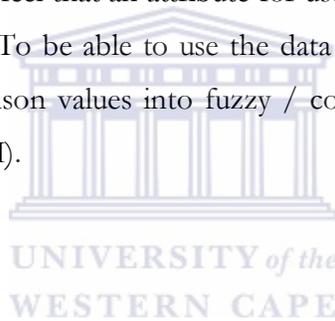
Table 9: Outcome from Fuzzy AHP (with extent analysis)

The overall weight of ATutor was 0.541 and that of Moodle was 0.454. It has an error of 0.005 (i.e. $1 - (0.541 + 0.454)$). This shows that the evaluation judgements were consistent. This has been deduced from the fact that the error is equivalent to consistency (Karlsson et al., 1998).

Fourth cycle of SSM: Group Fuzzy AHP

The evaluators gave their evaluation comparison judgements of the different alternative software products in crisp values. The crisp values of the evaluation judgement were then fuzzified (see Appendix I for full description). Then the fuzzy comparison matrix is split into a crisp pairwise comparison matrix. This is done to get the ranking of each evaluator before the overall ranking results which is combined by using a geometric mean technique. These are the steps of the Group Fuzzy AHP which differ from other algorithms described in the literature (Tang & Zhang, 2007; Ota et al., 2008).

This algorithm addresses the problem of a user not being able to give a crisp value when evaluating software. For example a user might feel that an attribute for user interface is not good, nor it is bad, it might be something in between. To be able to use the data collected at OUT, it was decided to convert the discrete/ crisp comparison values into fuzzy / continuous values using some defined conversion method (see Appendix H).



The flowchart for the steps of the Group Fuzzy AHP is depicted in Figure 16 below:

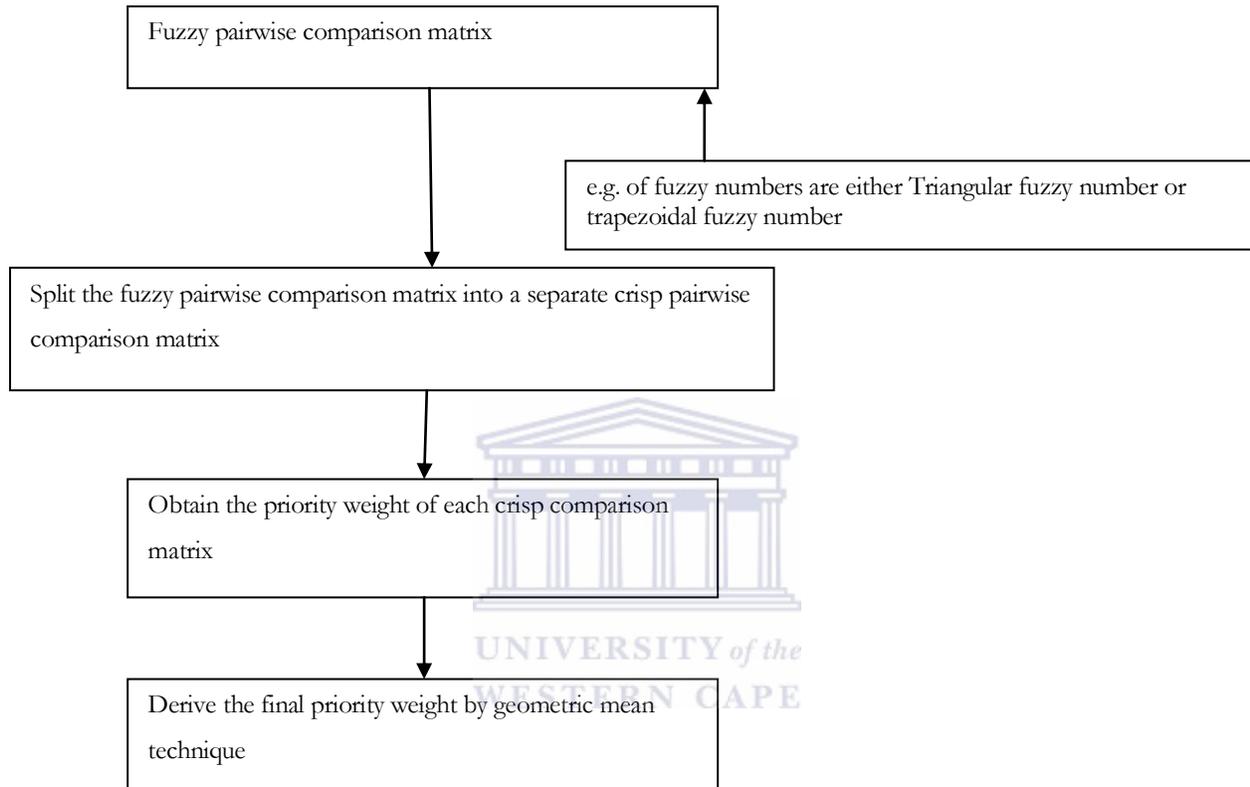


Figure 16: Flowchart of the Group Fuzzy AHP

The detailed presentation how Group Fuzzy AHP was used is shown in the Appendix M.

Group Fuzzy AHP is depicted as A_7 in Figure 17 and is the fourth cycle of SSM.

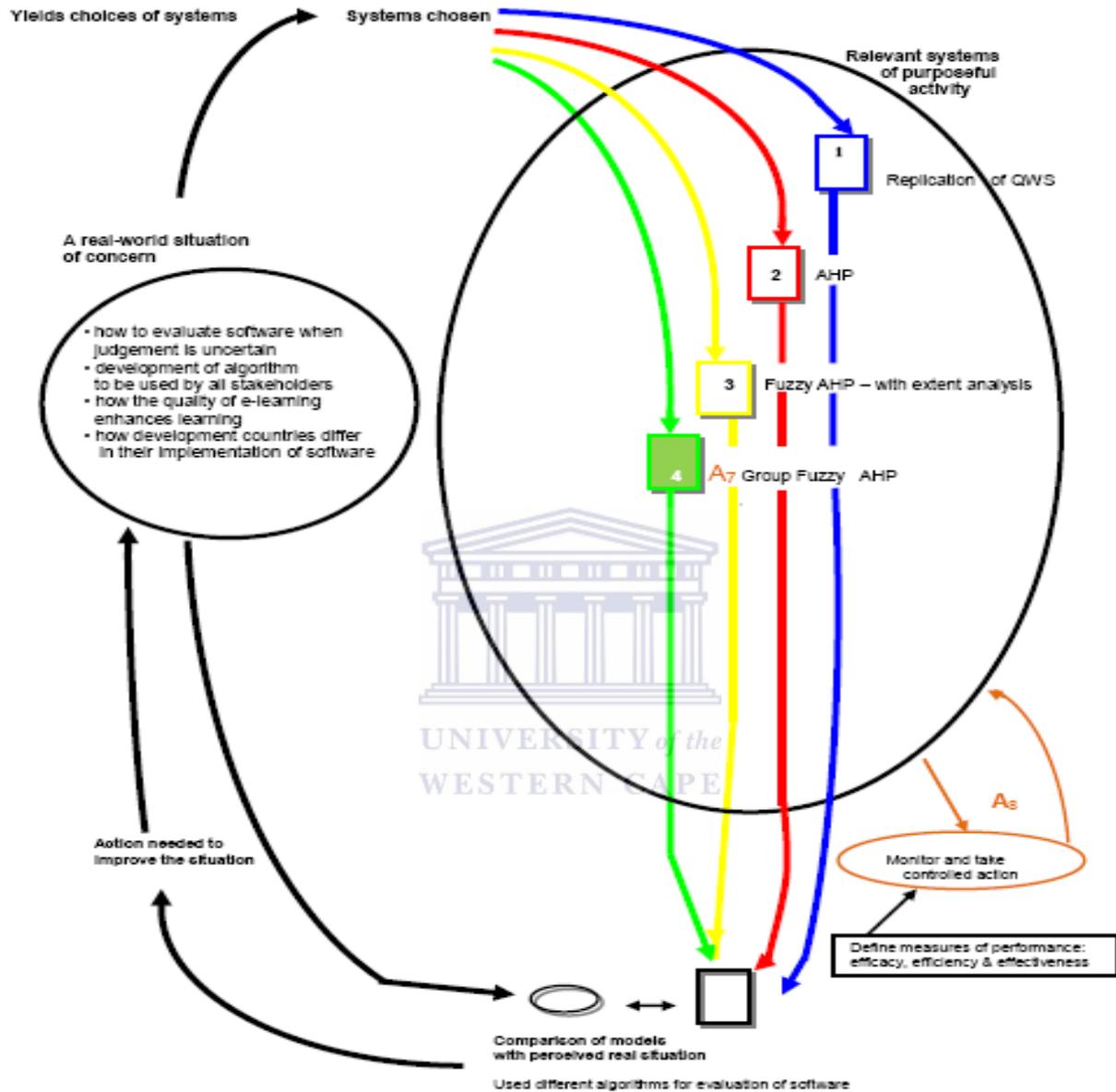


Figure 17: The fourth cycle of SSM

The outcome of the execution of the Group Fuzzy AHP (the fourth cycle) is presented below. Since the crisp scale was converted to the triangular fuzzy scale, the outcome for each value from the comparison instance consists of three values outcomes for lower, middle and upper element.

The following table shows the priority weights for the lower element (for complete calculation see Appendix M).

The outcome for the lower elements

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.280	27	0.333	0.093	0.500	0.500
		28	0.345	0.097	0.800	0.200
		29	0.321	0.090	0.833	0.167
understandability	0.188	23	0.212	0.040	0.667	0.333
		24	0.158	0.030	0.111	0.889
		21	0.183	0.034	0.200	0.800
		25	0.447	0.084	0.250	0.750
operability	0.185	10	0.603	0.111	0.500	0.500
		11	0.111	0.021	0.889	0.111
		12	0.285	0.053	0.100	0.900
attractiveness	0.113	13	0.181	0.021	0.143	0.857
		14	0.093	0.011	0.167	0.833
		15	0.168	0.019	0.500	0.500
		16	0.102	0.012	0.143	0.857
		17	0.079	0.009	0.889	0.111
		18	0.111	0.013	0.111	0.889
		19	0.167	0.019	0.250	0.750
		20	0.098	0.011	0.800	0.200
usability compliance	0.235	22	1.000	0.235	0.100	0.900
				Priority	0.398	0.602
				TOTAL	1	

Table 10: Outcome when using Group Fuzzy AHP for the lower elements

The outcome for the middle elements

The following table shows the priority weights for the middle elements.

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.311	27	0.348	0.108	0.667	0.333
		28	0.343	0.107	0.833	0.167
		29	0.309	0.096	0.857	0.143
understandability	0.202	23	0.245	0.049	0.750	0.250
		24	0.147	0.030	0.125	0.875
		21	0.199	0.040	0.250	0.750
operability	0.182	25	0.408	0.083	0.333	0.667
		10	0.655	0.119	0.667	0.333
		11	0.097	0.018	0.900	0.100
attractiveness	0.095	12	0.248	0.045	0.100	0.900
		13	0.194	0.019	0.167	0.833
		14	0.107	0.010	0.200	0.800
		15	0.187	0.018	0.667	0.333
		16	0.090	0.009	0.167	0.833
		17	0.066	0.006	0.900	0.100
		18	0.101	0.010	0.125	0.875
		19	0.165	0.016	0.333	0.667
usability compliance	0.210	20	0.090	0.009	0.833	0.167
		22	1.000	0.210	0.111	0.889
				Priority	0.483	0.517
				TOTAL	1	

Table 11: Outcome when using Group Fuzzy AHP for the middle elements

The outcome for the upper elements

The following table shows the priority weights for the upper element.

	local weight	Question No.	local weight	global weight (overall)		
					Moodle	ATutor
learnability	0.361	27	0.380	0.137	0.729	0.271
		28	0.339	0.122	0.837	0.163
		29	0.281	0.101	0.856	0.144
understandability	0.215	23	0.311	0.067	0.778	0.222
		24	0.154	0.033	0.163	0.837
		21	0.192	0.041	0.350	0.650
		25	0.343	0.074	0.500	0.500
operability	0.157	10	0.695	0.109	0.729	0.271
		11	0.090	0.014	0.883	0.117
		12	0.214	0.034	0.129	0.871
attractiveness	0.083	13	0.203	0.017	0.222	0.778
		14	0.124	0.010	0.271	0.729
		15	0.195	0.016	0.729	0.271
		16	0.085	0.007	0.222	0.778
		17	0.056	0.005	0.883	0.117
		18	0.093	0.008	0.163	0.837
		19	0.161	0.013	0.500	0.500
		20	0.083	0.007	0.837	0.163
usability compliance	0.184	22	1.000	0.184	0.144	0.856
					0.558	0.442
				TOTAL	1	

Table 12: Outcome when using Group Fuzzy AHP for the upper elements

Outcome of the combined Group Fuzzy AHP

After obtaining the outcome for lower, middle and upper elements, they were combined using the geometric mean. The geometric mean is calculated by multiplying the numbers in the data set and

then the n^{th} root of the resulting product is taken. The geometric mean of a data set $[b_1, b_2, \dots, b_n]$ is given by

$$\sqrt[n]{\prod_{i=1}^n b_i} = \sqrt[n]{b_1 \cdot b_2 \cdot b_3 \dots b_n} \text{ while arithmetic mean is given by } \frac{1}{n} \sum_{i=1}^n b_i$$

Note that the average and geometric mean differs (see Table 13).

Position		Moodle	ATutor
Lower	1	0.398	0.602
Middle	2	0.483	0.517
upper	3	0.558	0.442
geometric mean		0.475	0.516
average		0.480	0.520
Total		0.991	
Error		0.009	

Table 13: Outcome of the combined Group Fuzzy AHP



Note that total is equal to geometric mean of Moodle plus the geometric mean of ATutor. The overall priority weight of ATutor was 0.520 and that of Moodle was 0.48. However since the total was not 1 as expected it seems as if the result has an error of 0.009 (i.e. $1 - 0.991$). Thus, 0.009 was the consistency index.

Comparison of the three algorithms outcomes

To monitor the execution of the analysis and outcomes of AHP, Group Fuzzy AHP and Fuzzy AHP (with extent analysis) algorithms using the same data set in the third and fourth cycles of SSM (See letter A_6 and A_8 respectively in Figure 15 and Figure 17).

Method	Priority weight			
	Moodle	ATutor	Total	Error
Conventional AHP	0.483	0.517	1	0
Group Fuzzy AHP	0.475	0.516	0.991	0.009
Fuzzy AHP (with extent analysis)	0.454	0.541	0.995	0.005

Table 14: Comparison of outcomes

Since the consistency ratio correlate to the judgemental errors in pairwise comparisons (Karlsson et al., 1998) from Table 14, it can be concluded that the column which indicate errors correspond to the consistency ratio (Saaty, 1980).

Conventional AHP algorithm has the smallest error compared to Fuzzy AHP algorithm but cannot be used in situation which has fuzziness (uncertainty). Therefore, fuzzy AHP algorithm (with extent analysis) works better because it has smaller error compared to group fuzzy AHP algorithm. In addition, group fuzzy AHP algorithm can be used when the evaluators judgement have certainty and uncertainty. This is due to the fact that group Fuzzy AHP is based on the principles of AHP. Apart from the correctness of the algorithm, other aspects which can be used to differentiate between the algorithms are: the time efficiency (or time complexity), space efficiency, simplicity and generality.

Both algorithms are effective but fuzzy AHP (with extent analysis) and group fuzzy AHP outweigh conventional AHP in terms of generality. This is because group Fuzzy AHP and fuzzy AHP (with extent analysis) can be used when user judgements are either exact (i.e. certain) or fuzzy (i.e. imprecise).

Time complexity refers to time in which the algorithm runs. It is determined by finding the upper bound on the execution time (Chang, 1996). Chang (1996) found that extent analysis (for n number of criteria) has the time of complexity equals to $n(n + 6)$ and the conventional AHP has a time complexity equal to $(n(n - 1)/2)$. Thus, the group fuzzy AHP (which uses the triangular fuzzy number) has a time complexity equal to three times that of conventional AHP because each element (from the lower, middle and upper position) need to be computed using the AHP.

Conclusion

Evaluation of software (in terms of software quality) is complex. The complexity arises from the multiple characteristics, sub-characteristics and attributes measurements, which needs to be combined and compared as per different software products. The fact is that you cannot obtain complete and comprehensive quality information about each software product. This re-iterates the post-positivist stance, which states that “*all measurement are fallible or imperfect*”. Thus, the evaluation of software is always subjective.

It has been shown how the traditional AHP algorithm can be extended and be used in a situation where the evaluator (user) has imprecise information about evaluation judgements. AHP was chosen after testing its usefulness and applicability in a replication experiment. Then fuzzy logic was incorporated in AHP to address the uncertainty of user judgement during the evaluation of software. This was done using a modified Fuzzy AHP algorithm (with extent analysis) and also by a new algorithm called Group Fuzzy AHP. The validations of these algorithms were done to verify their results. Some errors were identified.

The aim of the scientific experiment is to uncover the objective truth. But, how can the post-positivist stance adhere to this goal of science? Triangulation of research methods (Trochim & Donnelly, 2001) in both data collection and analysis; helps the researcher to better understand what the reality is. Since the evaluations from individuals are biased, due to their culture, experience and worldview, the developed algorithms must allow for the merging of the evaluation from different individuals (Trochim & Donnelly, 2001). Theories, which survive from intense scrutiny of a community of evaluators will hopefully, create a valid or “truthful” knowledge. Such knowledge evolves through the process of variation, selection and retention like species’ survival in the theory of natural selection. In the next chapter, the interpretation of the results will be presented.

CHAPTER 5

THE INTERPRETATION OF THE RESULTS

To raise new questions, new possibilities, to regard old problems from a new angle, requires creative imagination and marks real advance in science.

Albert Einstein [1879-1955]

Introduction

In the previous chapter, the research results were presented.

In this chapter, the findings from each cycle of the SSM and from the research questions will be interpreted. At the end of this chapter, a final framework will be proposed.

Findings of the replication experiment –testing the feasibility of AHP

In this cycle, the illustrations how a replication of data analysis can be used to verify the validity of an empirical evaluation of software were presented. A characteristic of QWS is that it eliminates other alternatives and criteria that do not satisfy the mandatory requirements for the evaluation. This is the reason why Graf and List (2005) did not give the evaluation results of all 36 e-learning systems they initially evaluated, but only of nine that were ranked as the best. It has been shown that by replicating the experiment using AHP, similar results can be reproduced. Thus, it confirms that the Graf and List evaluation using QWS was consistent.

When revisiting the questions “*can AHP algorithm be used in empirical evaluation of software*”, the following was found:

- (i) Although the AHP method ranked the e-learning systems in a slightly different order than QWS, the first four positions in the ranked list were the same.
- (ii) The AHP algorithm is easier to implement than QWS. With AHP, comparisons are easier: the consistency of the evaluation judgements can be checked and the ranking of e-learning systems can be done using priority weight calculation. With QWS, the evaluation is done by summing the symbols (assigned to each e-learning system). Therefore, it is difficult to draw conclusions when using the QWS method. For example which of $3E+2*$ and $2E+3*+4Q$ is the greater.

(iii) The results of the QWS method was validated using consistency checking and by comparing the results.

The consistency of the evaluation judgements done by Graf and List using QWS, was checked and the consistency indices (CI) were equal to zero (see Appendix B). This means that Graf and List's evaluation was consistent. The consistency of judgement was probably due to the small number of criteria/categories and sub-criteria/sub-categories that were compared. If there is an increase in the number of criteria and sub-criteria, the pairwise comparisons will increase geometrically. Researchers have used criteria, sub-criteria and characteristics, sub-characteristics interchangeably. This can lead to inconsistency or failure of the AHP algorithm (Cheng et al., 1999).

The ranking results of both QWS and AHP were similar up to the fourth position. Thus, the outcome of AHP agrees with that of QWS to a certain extent. The difference in ranking results might have arisen because it was not an exact replication testing (Rodriguez et al., 2006).

This section shows the importance of validating the results of an empirical evaluation through replication. It also confirms that replication is useful to validate results and to evaluate decision-making algorithms. The replication of the analysis with AHP provides insight into how the multi-criteria evaluation tool, AHP, can be used to evaluate software.

Findings of the implementation of AHP

The objective of this case study was to determine the suitability of the AHP algorithm in the evaluation of FOSS when most of the evaluators are users with little technical IT experience. In the evaluation, qualitative methods (using participative observation and focus group interviews) were complimented by quantitative methods (questionnaires analysed using AHP).

It has been shown that AHP is suitable for evaluating software in a developing country where IT experience is limited because of its accuracy and flexibility in making a logical, consistent and informed decision.

Since the AHP algorithm involves linear modeling, contributing metrics values can in certain cases create a non-compensatory effect. This means that a system being evaluated may be ranked higher than another, even though one or more of its constituent criteria, sub-criteria, and attributes have

lower weights than the system it is compared to. The use of the analytical network process algorithm (ANP) may remedy this effect as it takes the dependence of criteria, sub-criteria, and attributes at the same level of the hierarchy, into account.

AHP deals with crisp (real) values of evaluation judgements, but human reasoning is imprecise, uncertain and fuzzy (Mikhailov and Tsvetinov, 2004, p. 23). Furthermore, when the number of criteria considered increases, the number of pairwise comparisons increases geometrically. This can lead to inconsistencies or even that the AHP algorithm fails completely. Fuzzy AHP could address this problem and is proposed as an alternative method for imprecise problems or problems with more criteria. The results of this study agree with the results of Graf and List (2005, p. 165) who found in their study (using QWS) that Moodle is preferred above 36 FOSS e-learning systems. The stakeholders at OUT regarded usability as the most important criterion required for the successful implementation of an e-learning system (see Appendix M).

In our study, it has been shown that—with a well planned evaluation process, good data collection and analysis methods—both novice and technical users can be successfully involved in the evaluation of software. The results contribute to the theory and methodology of the evaluation of FOSS and can be used to design and develop a framework for the evaluation of FOSS for developing countries.

Thus, using AHP in a developing country is suitable because it simplifies a complex problem by breaking it up into smaller steps (see Figure 20) that can help in visualizing the problem. In, Figure 11 the eight steps of the AHP is depicted. The steps of AHP includes: defining the goals and outcomes of the problem; decomposing the problem into a hierarchical structure of criteria, sub-criteria, attributes, and alternatives; computing pairwise comparisons; employing the Eigenvalue method to estimate relative weights; checking consistency and finally combining the relative weight to obtain the overall rating for the alternatives.

Deployability, maintainability and usability were the only characteristics considered in this evaluation as they are considered as the most important characteristics needed for the implementation of software in a developing country.

The subjective evaluation was consistent for the attributes identified for the criteria deployability; maintainability and usability. The evaluation was shown to be consistent since the computed

consistency indices, for all pairwise comparisons, were equal to zero thus confirming that all stakeholders can participate in the evaluation process. Involving all stakeholders in the selection of software has the added advantage of greater acceptance of the system.

Fuzzy AHP was implemented in a next cycle of research to see whether it can address the weakness of AHP.

Findings of the implementation of Fuzzy AHP (with extent analysis) and Group Fuzzy AHP

Using Fuzzy AHP (with extent analysis) and Group Fuzzy AHP, it has been shown how preference and consensus can be attained if a group decision-making process is used during the evaluation of software. It differs from the traditional method, which uses preferences and consensus generated from crisp values (Ross, 2004) to evaluate software.

The level of accuracy of the prioritization outcome when Group Fuzzy AHP was applied, was 99.991% where as when fuzzy AHP (with extent analysis) gave 0.005% greater accuracy (namely 99.995%).

It can be concluded from the third and fourth cycle of SSM that both algorithms are useful and can be incorporated in the design and development of new techniques for for the evaluation of software (specifically in developing countries). However, it is argued that group Fuzzy AHP is an improvement to Fuzzy AHP (with extent analysis because has less computations. Apart from these mentioned advantages, Group Fuzzy AHP has those advantages of conventional AHP (Sanga & Venter, 2009), which are: it is flexible, it integrates deductive approaches, it acknowledges interdependence of elements of software systems, it has a hierarchical structure, measures intangibles, tracks logical consistency, gives an overall estimation, consider relative priorities and improves judgements.

The limitations of Group Fuzzy (with extent analysis) and Group Fuzzy AHP should probably be addressed in the next cycle. Examples of limitations are: (i) checking if Group Fuzzy (with extent analysis) and Group Fuzzy AHP, preserve the consistency of the evaluator's judgement; and (ii) whether Group Fuzzy (with extent analysis) and Group Fuzzy AHP ignore the dependence between the elements at the same level of the hierarchy, as is the case with AHP. This can be addressed by the incorporation of the Analytic Network Process (ANP) in both these algorithms.

Research questions revisited

Several questions were posed as sub questions to the problem statement. The following research questions were posed in chapter 1:

- a. What are the key attributes to consider (in terms of usability, deployability and maintainability) in order to evaluate the e-learning system options for OUT?
- b. How should the attributes be measured?
- c. How can the measured attributes be used to create a framework for the evaluation of e-learning systems?

The following section presents how each of these research questions were addressed in this thesis.

Attribute identification

Posed question

What are the key attributes to consider (in terms of usability, deployability and maintainability) in order to evaluate the e-learning system options for OUT?

Findings

The software quality characteristics, sub-characteristics and attributes for usability, deployability and maintainability were identified by users and some were adopted from the literature. The fully representative list is shown in Appendix D and Appendix E. These findings represent the second cycle of the SSM (see Figure 12).

Measurement of attributes

Posed question — How should the attributes be measured?

Findings

In chapter 4, it was shown how subjective software measurement can be translated from linguistic descriptions to discrete values, which in turn was extended to continuous values. This was done using fuzzy logic (see Appendix H). Fuzzy values were used in the algorithms to reflect the uncertain judgement of stakeholders. This is represented as the third and fourth cycle of SSM (see Figure 12).

Framework formulation

How can the measured attributes be used to create a framework for the evaluation of e-learning systems?

Findings

The stepwise procedures for the formulation of the final framework, which incorporate algorithms that can handle uncertain of user judgement, will be presented. The framework was formed after the execution of the first, second, third and fourth cycles of the SSM. The application of the framework in our case study resulted into the following findings. In the first cycle of SSM, it was found that, Moodle outweigh ATutor. In the second cycle of SSM when AHP is used the results shows Moodle is preferable than ATutor. In the third cycle of SSM when Fuzzy AHP (with extent analysis) was used ATutor outweigh Moodle in usability characteristics. The ATutor outweigh Moodle in usability characteristic when the Group Fuzzy AHP was used in the fourth cycle. The novel property of the Fuzzy AHP (with extent analysis) and the Group Fuzzy Group is that it can give a computational results with expression of the degree of uncertainty (i.e. either pessimistic, moderate or optimistic).

In the chapter of literature review, the gaps in the existing frameworks were identified to enhance metrics analysis and decision analysis. In order to fill the gaps in current knowledge, two software systems were evaluated in terms of software quality. The evaluation used the subjective software measurement from both qualitative and quantitative methods to collect the data. In this study, the objective software measurement was not covered but a database for it was developed.

Figure 18 shows the proposed database relationship diagram representing the typical representation of objective software measurement.

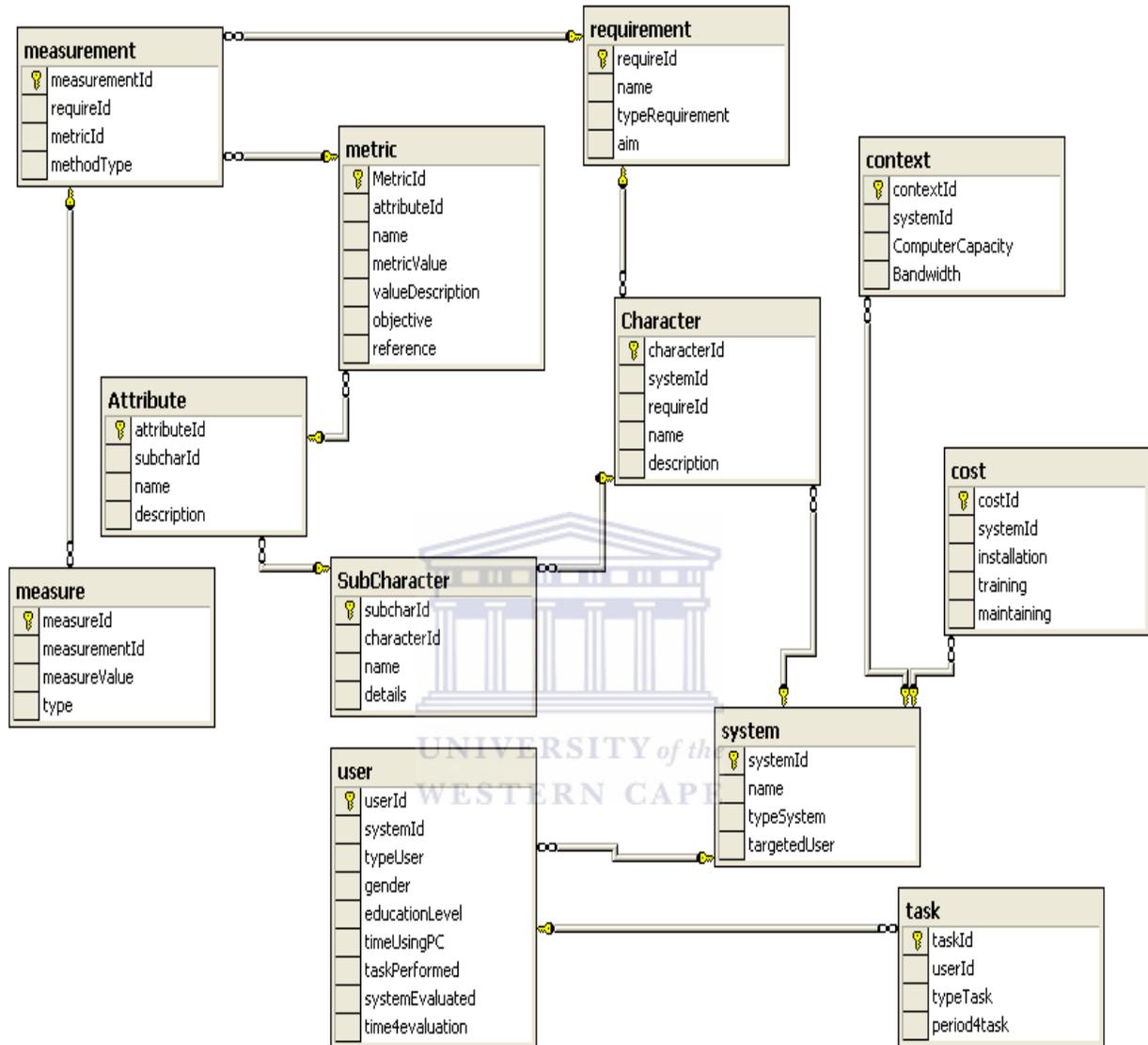


Figure 18: Proposed database relationship

The database relationship diagram formulated corresponding to the proposed conceptual framework – which deals with subjective evaluation of software according to software quality. The Figure 19 depicts the conceptual framework:

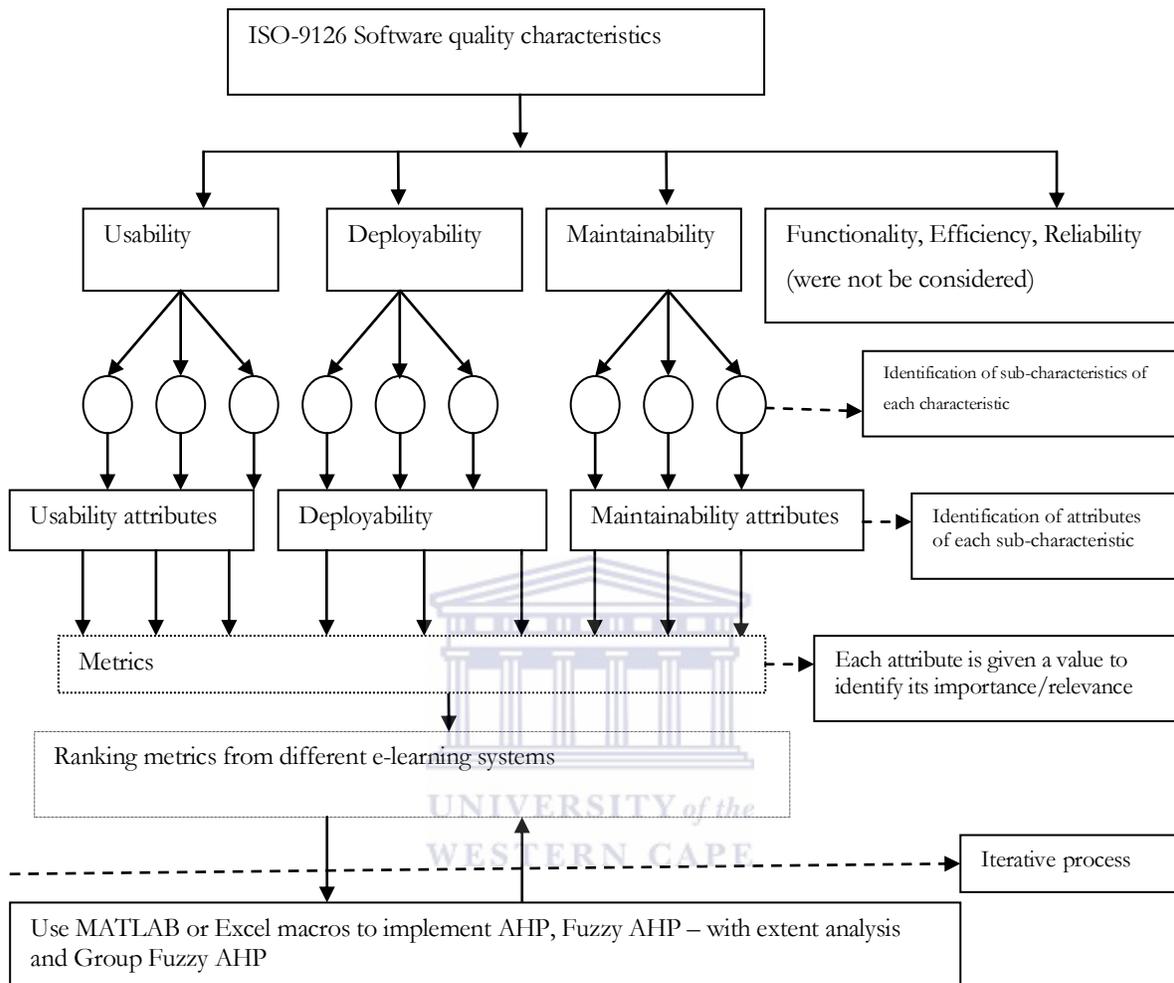


Figure 19: Conceptual framework

The aim was to design a framework for decision analysis during software evaluation to enhance decision making in situation where there is insufficient information, ambiguity, fuzziness and vagueness. In order to work towards development of the final framework all the processes involved in the research study (from conceptual framework) were combined. The final design of the framework consists of many steps as depicted in Figure 20. The framework can be expanded to include the steps for AHP, Fuzzy AHP (with extent analysis) and Group Fuzzy AHP.

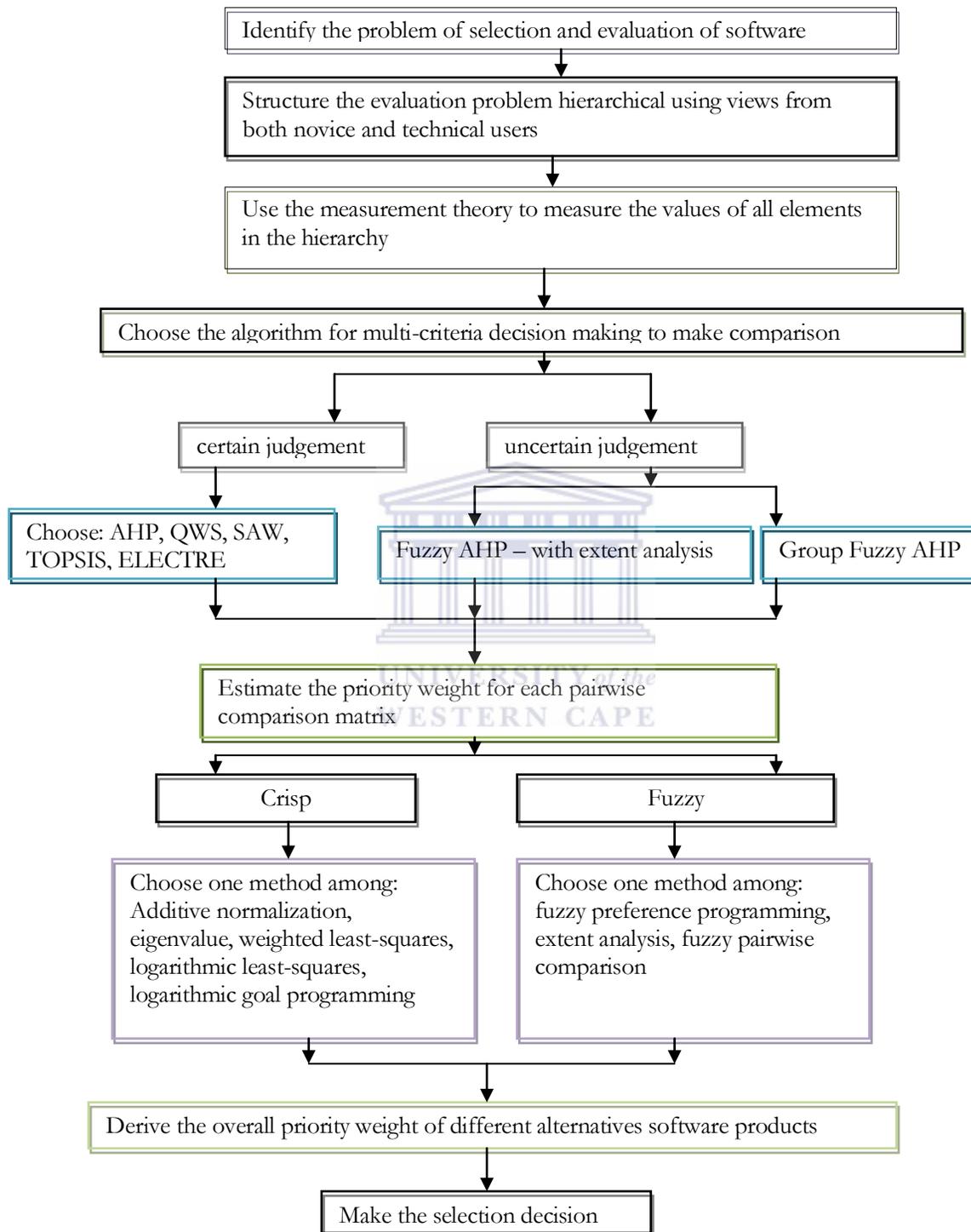


Figure 20: Final framework

The steps for these algorithms are shown in Figure 11, Figure 14 and Figure 16. The framework can be enlarged to include the steps of other algorithms such as simple additive weighting (SAW); a technique for order preference by similarity to ideal solution (TOPSIS) and elimination et choice translation reality (ELECTRE) (Chou et al., 2008).

Motivation for using mixed research methods

This study has shown how post-positivist and mixed research methods can be effectively used. For findings generated from any study undertaken from a post-positivist and mixed research stance, to be accepted by experienced Computer Science researchers, external observers and the general IT community, rigor in evaluation is needed. The recommendations for evaluating the rigor of positivist case studies in Information Systems, as identified by Dubé and Paré (2003), are also valid for the case study based research in Computer Science. The recommendations mentioned by Dubé and Paré (2003), are classified into design issues, data collection and data analysis aspects:

Design issues

- Clear research questions were identified
- Clear rationales for single case study has been specified
- Pilot case study was implemented in order to help refine the design and the data collection plans
- Data collection using different methods were conducted. Thus, this helped to exploit the richness of the various data collection methods when examining phenomena as they unfold

Data collection issues

- Detailed information with respect to the data collection methods in the appendices, separate report, paper/articles published in conferences and journal and procedures for sampling strategies were provided.
- Tables was used to summarize information about the strength of the research methodology and methods used in data collection process
- Triangulation of data were done in order to increase internal validity of the findings and provide clear explanations on how the triangulation process was achieved

Data analysis issues

- Clear descriptions of the analytic methods were provided and sufficient relevant information were given so that any reader can follow the derivation of evidence from initial research questions to conclusions and vice-versa
- Provision of some example of how the results were obtained. This was used as a means to reflect the validity of the results.
- Quotes were presented so that external observers can reach an independent judgment regarding the merits of the analysis
- Findings from existing literature (both similar and conflicting) in case study research were provided so as to increase the confidence in the findings

Conclusion

In this Chapter, the interpretations of the results have been presented. In the first cycle of the SSM, the usefulness of AHP was learnt. The lesson learnt lead to the implementation of it in the case study – in second cycle of SSM. The weakness of AHP necessitated developing an algorithm to handle uncertain judgements of users. This is the reason that Fuzzy AHP (with extent analysis) and Group Fuzzy AHP were implemented in the third and fourth cycles of SSM. All these cyclic interventions were done to answer the research questions, which dealt with attributes identification, attributes measurement and framework formulation to aid users in evaluation of software according to quality. The unique feature of the framework is that it allows the identification of software quality attributes, sub-characteristics and characteristics that are of interest to user in a specific environment, in this thesis, a case study of developing country's University was considered. In addition, it incorporates algorithms which compute prioritization of user judgements when the evaluation situation is either certain or uncertain. This has been shown using Fuzzy AHP (with extent analysis) and group Fuzzy AHP. The combination of these algorithms allows the inclusion of views from all stakeholders. The combination of the algorithms was termed as Fuzzy Free Open Source Software Evaluation Technique (FOSSET).

The next Chapter will present the discussion and concluding remarks.

CHAPTER 6

DISCUSSION AND CONCLUDING REMARKS

Every theory presented as a scientific concept is just that; it's a theory that tries to explain more about the world than previous theories have done. It is open to being challenged and to being proven incorrect.

Marvin Harris [1927-2001]

Introduction

In the previous chapter, the interpretations of the research results were presented and a framework for the evaluation of software was developed.

The focus of this chapter is discussing and evaluating the contribution of this thesis to the body of knowledge of software quality.

Why this thesis is a contribution

For a thesis to be a contribution, it must add value to the body of knowledge that exists in the specific field under investigation.

What is knowledge?

Lincoln and Guba define knowledge as follows (Lincoln & Guba, 1985):

Knowledge consists of those constructions about which there is a relative consensus (or at least some movement towards consensus) among those competent (and in the case of more arcane material, trusted) to interpret the substance of the construction. Multiple “knowledges” can coexist when equally competent (or trusted) interpreters disagree (Guba and Lincoln, 1985, p. 113).

In order for knowledge to be created, either new theories can be formulated or existing theories can be modified. Theory developments often arise from research questions asked to clarify a problem and which can be presented as gaps in the literature. The research study must show how the researcher identified and reacted to the gaps in the literature. In this thesis, gaps in the knowledge regarding the “evaluation of software according to quality of software” were identified as lack of research dealing

with the uncertainty of evaluators judgements during the evaluation of Free and open source e-learning systems in developing countries.

What theories were considered?

Scientific theories are linguistic statements put forward to predict, explain, describe and explore phenomena in the real world, theories can be classified into: (i) exploration (ii) description (iii) explanation and (iv) prediction (see Table 15). A review of the literature regarding the classification of theories could be found in the field of management sciences and information systems which, in my opinion, also hold true for the sciences (including Computer Science), (Gregor, 2006) and is presented in Table 15.

Theory type	Building block of theory	Distinguishing attributes
Analysis	<i>"what is"</i>	The theory under this category does not extend beyond analysis and description. Thus no causal relationships and no predictions among phenomena are specified.
Explanation	<i>"what is", "how", "why", "when", and "where"</i>	The theory under this category provides explanations (or deduction) but does not provide predictions. Therefore, there are no testable propositions.
Prediction	<i>"what is" and "what will be"</i>	The theory under this category provides predictions and hence do provide testable propositions. On the other hand does not provide causal relationship.
Explanation and prediction (EP)	<i>"what is", "how", "why", "when", "where" and "what will be"</i>	The theory under this category does provide predictions and has both testable propositions and causal relationship.
Design and action	<i>"how to do something"</i>	The theory under this category gives precise prescriptions (e.g., methods, techniques, principles of form and function) for developing an artifact.

Table 15: Taxonomy of theory - Adapted from (Gregor, 2006)

In this thesis, prediction, and design and action theories were applied. An algorithm was developed that can be used to assist user to predict which kind of software would be most successful in a specific environment.

The developed flowcharts of the algorithms of AHP (see Figure 11), Fuzzy AHP (see Figure 14) and Group Fuzzy AHP (see Figure 16) give a precise the prescriptions of the steps of the techniques that

can be incorporated in the design and development of new techniques for evaluating and selecting software.

What counts as knowledge?

Gregor (2006) argue that knowledge is considered to be a contribution depending on the plausibility, credibility, consistency and transferability of the linguistic statements (i.e. arguments) made by the researcher. Gibbons et al. (1994) clarifies these terms by stating that credibility comes after some of the research results are presented in journal or conferences paper / article or technical report; while, consistency and plausibility must be verified in accordance with the technical and social norms governing science. This ensures that knowledge is generated within legitimate boundaries after surviving the scrutiny and criticism from a broader community of knowledge seekers. Thus, comments from technical reviewers of different forums (i.e. workshop, conference, and journal) act as a quality control assurance. Furthermore, Gibbons et al. argue that transferability deals with ensuring that the knowledge production does not occur in vacuum (i.e. knowledge is part of society and any person can reproduce/replicate the research study).

The following are the gauges, from those identified by Gregor (2006) and Gibbons et al. (1994), which have been used to evaluate this thesis. The comments, criticism, suggestions and advice received from publications were incorporated in the development of the algorithms which in turn improved the results. Parts of the thesis have either been presented or published in conferences, at workshop and in an accredited journal. These are:

1. Sanga, C. & Venter, I. M. (2006). Using Soft Systems Methodology to Understand how to Exploit Learning Technologies in Developing Countries, *Education Students' Regional Research Conference organized by Five Higher Learning Institutions of Western Cape*, November 2006, PET-University of the Western Cape, South Africa
2. Sanga, C., Lwoga, E.T., & Venter, I.M. (2006). Open Courseware as a Tool for Teaching and Learning in Africa, TEDC, pp. 55-56, ISBN: 0-7695-2633-0, *Fourth IEEE International Workshop on Technology for Education in Developing Countries (TEDC'06)*. DOI: <http://doi.ieeecomputersociety.org/10.1109/TEDC.2006.23>
3. Sanga, C., Venter, I.M., & Agbinya, J.I. A. Comparative Study Between Qualitative Weight Sum and Analytic Hierarchy Process Methodologies, *Annual Conference of the South African Institute of Computer Scientists and Information Technologists*, Fish River Sun, Sunshine Coast, South Africa, Sept 30 - 3 October, 2007. URL: <http://www.nmmu.ac.za/saicsit/2007/programme.htm>

4. Sanga, C. & Venter, I. M. (2009). Is a multi-criteria evaluation tool reserved for experts? The *Electronic Information Systems Evaluation (EJISE) journal*, Volume 12, Issue 2, pp. 165 – 176. URL: <http://www.ejise.com/>
5. Sanga, C. & Venter, I. M. “What is the value of replicating the data analysis of an empirical evaluation?” *Conference of Technology for Innovation and Education in Developing Countries (TEDC 2010)*, February, 2010. Mozambique. URL: <http://cs.joensuu.fi/tedc2010/call.html>

Contextualising knowledge and its contribution in Computer Science

Tedre (2006) classifies Computer Science as either narrow or broad interpretations of knowledge in computer science. This is as represented diagrammatically below (Tedre, 2006):

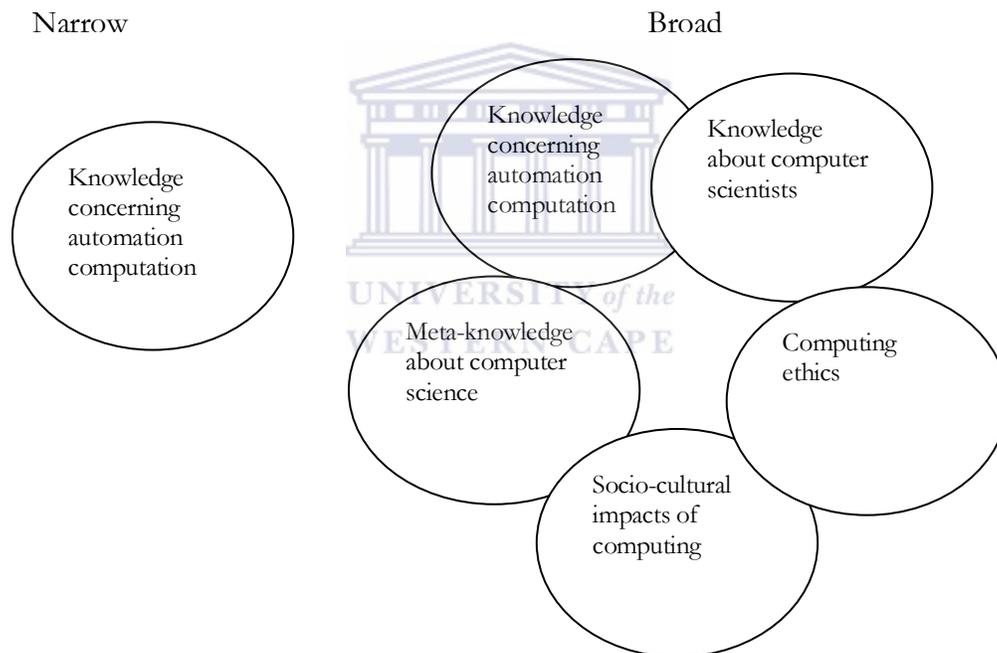


Figure 21: Interpretations of knowledge in Computer Science

Computer Science is different from other subjects which are disciplinary based, computer science is interdisciplinary and hence judgement of what might be called knowledge contribution is very difficult. Diversity of knowledge production comes from a number of disciplines intertwined such as computer science, statistics, management science, operational research, psychology, education and mathematics (to mention a few) (Gibbons et al., 1994).

Tedre (2007) summarises the above by arguing that mixing disciplines is a problem, that researchers in computer science face, and he proposes that in order to avoid flawed research (that is if a researcher adopts a research method from another discipline), the researcher must also adopt its epistemology, methodologies and validation techniques.² According to Tedre, the official ACM curriculum does not have a research methodology as one of the components of its taxonomy. He further classifies the computer science into the three intertwined disciplines:

DISCIPLINE	BEDROCK	AIM
Theoretical	Mathematical science	Articulate theories
Engineering	Engineering	Develop systems that solve problems
Empirical	Natural science	Investigate and explain

Table 16: Diversity of disciplines encompassed in computer science

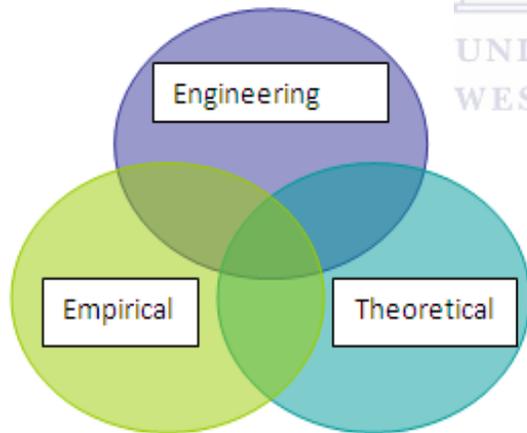


Figure 22: Diversity of disciplines encompassed in Computer Science

Therefore – as shown before in Figure 22 – Computer Science has a bond with engineering, empirical and theoretical disciplines (Denning, 2003; Eden, 2007). This diversity of disciplines needs to be assessed when the knowledge contribution of a study is sought by the researcher (Abran et al., 2001).

² <http://www.computer.org/portal/web/publications/acmtaxonomy>

Analysing the knowledge contribution of this thesis, it was found that it deals with software quality which – resides within the software engineering discipline, – which is also part of computer science (Bourque et al., 1999). This thesis seeks to contribute knowledge towards this area by filling knowledge gaps in software quality analysis (Moody, 2005). Some of the open research questions identified by Moody (2005) are: (i) proliferation of proposals (i.e. no consensus on common framework), (ii) lack of empirical testing, lack of adoption in practice, (iii) different levels of generality, (iv) lack of agreement on concepts and terminology, (v) lack of consistency with related fields and standards, (vi) lack of measurement, (vii) lack of evaluation procedures, (viii) lack of guidelines for improvement, (ix) focus on static models (data and information)–they must include dynamic model (i.e. functionality of systems), (x) focus on product quality, and lack of knowledge about practices. In order to answer these open research questions different disciplines must be studied.

The interdisciplinary nature of Computer Science has caused it to grow faster than other science subjects (Tedre, 2007). Thus in order to avoid misunderstanding of any claim (research results) of new knowledge, the thesis or article or paper must specify how the theory was developed and validated. The Table 17 depicts the classification of the body of knowledge of software quality measurement (Schneidewind, 2002).

Issue	Function	Body of knowledge concerned
Goals	Analyse quality goals and specify quality requirements	Quality engineering, requirement engineering
Cost and risk	Evaluate economics and risk of quality goals	Economic analysis, risk analysis
Context	Analyse the applications environment	Systems analysis, software design
Operational profile	Analyse the software environment	Probability and statistical models
Models	Models quality and validate the model	Probability and statistical models
Data requirement	Define data types, phase, time and frequency of collection	Data analysis
Type and granularity of measurement	Define the statistical properties of the data	Measurement theory
Process and product test and evaluation	Analyse the relationship between product quality and process stability	Inspection and test methods
Process and product quality prediction	Assess and predict software quality	Measurement tools

Table 17: Body of knowledge for software quality measurement

UNIVERSITY of the
WESTERN CAPE

Assessment of the knowledge contribution in theory

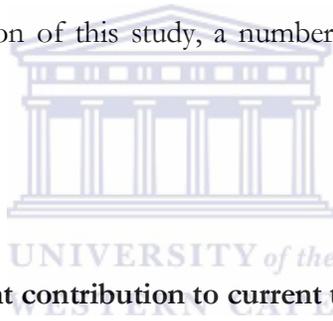
Whetten (1989) argue that for a legitimate theory contribution in any science subject, the researcher must clearly specify the four building blocks of theory development for knowledge. The building blocks are:

- *What*: This deals with which factors (constructs, variables, concepts) are part of the explanation (or theory) of social or individual phenomena of interest?
- *How*: This deals with the relationship of the mentioned factors. In addition, it shows causality depending on how the researcher conceptualises the pattern of factors. If *what* is combined with *how* then the subject or domain of theory contributed must be constituted. This combination answers the patterns or discrepancies or gaps –in the knowledge in empirical experiments (i.e. survey, observation etc).
- *Why*: This must give the justification behind the selection of constitutes factors' assumptions and how they have been modeled. Models or frameworks are formed from the combination

of “*What and How*”. From the model the testing of logical proposition can be done. Thus *why* must provide a logical justification of knowledge underlying the construction of model (which reconstitute *what and How*).

- *Who, where and when*: These contextual factors set the boundaries or limitation of generalizability from the developed framework or model. Essentially this must answer who are involved in the phenomena, where the phenomena occur and when the phenomena happened. The best way to identify “*who, where and when*” of the theory (i.e. limitation of theory) is by the initial test of the “*What, How and Why*”. In order to test the context sensitivity in a research, the research must set different settings of ideas until the underlying limitations conditions are found.

To assess the knowledge contribution of this study, a number of questions, as coined by Whetten (1989), will be used.



“What’s new?”

“Does the study make a significant contribution to current thinking?”

The research contributes to the development of new techniques for addressing consensus and judgement for group evaluation of software.

It provides a link between decision theory and computer science. Using the developed algorithms / techniques the best software for a specific application and environment can be selected, chosen or predicted. In addition, the developed techniques/algorithms can predict the quality of the software. The identification of the low quality characteristics, sub-characteristics and attributes collected from end-user evaluations can be used by the technical IT users to review, revise, modify or re-engineer the concerned software to improve its quality.

“So what?”

“Will the theory change the design and development of decision support systems? Are linkages to research evident? Are solutions proposed for remedying alleged deficiencies in current theories?”

The research undertaken by researchers and scholars interested in developing techniques which mimic the way evaluation judgement done by humans, show that use of real time multi-criteria decision making algorithm and fuzzy models is the future of this research field (Bonissone et al., 2009).

The traditional solutions using classical set theory have proved not to be conforming to reality, the way human beings perceive the software during evaluation. Instead of having only two choices of instances (for e.g. 0 or 1, true or false, yes or no), human beings perceive events or phenomena in many ways (for e.g. yes, may be, no). The use of fuzzy logic can address the uncertainty, incompleteness of information, randomness of ideas and imprecision of phenomena. Zadeh argue that there is a need for paradigm shift in development of software to assist human in evaluation (Zadeh, 2008). Recently, Wu (2009) researched multi-criteria decision-making (MCDM) “under uncertainties”, in particular the linguistic uncertainties. He proposed the incorporation of fuzzy logic in analytic hierarchy process algorithm. This thesis addressed some solutions identified in the thesis of Wu (2009) as area for future study.

Wu (2009) concurs with Saaty and Tran (2007) who oppose the fuzzy AHP algorithm because the algorithm capture first the certain and crisp judgements, the captured judgements are then fuzzified to be used in the algorithm. According to them, it is possible and more reasonable to obtain these uncertain judgements directly from the evaluators.

But, this research has shown that linguistic judgements can be fuzzified, that is: It does not fuzzify crisp numbers obtained directly from users, instead the users, who are the evaluators, provides the pairwise comparison in linguistic terms. These are then modeled using fuzzy logic.

“Why so?”**“Are the underlying logic and supporting evidence compelling?”**

The evaluation of software in this thesis has been viewed from a *post-positivism* stance. Soft system methodology was used to manage the research analysis process. In Chapter 3 the epistemological stance, theoretical perspective methodologies and methods were explained. Furthermore, the mix of research methods adopted in this research study was explained and motivated.

The results were discussed in Chapter 4 and reflect the research questions which lead to the research. Research has shown the importance of involving all stakeholders in the selection and evaluation of software. This is important for its sustainability in term of the stakeholders being able to maintain, use, and deploy the software.

The results from this study confirms the conclusion by Ramesh et al. (2004) that

"CS research focuses on a variety of technical topics, using formulative approaches to study new entities that are either computing elements or abstract concepts, principally using mathematically-based research methods" (Ramesh et al., 2004, p. 11)

According to Ramesh et al. (2004), one of the areas of Computer Science research that receives little research attention is in terms of research approaches on the use of evaluative methodologies. The thesis also addresses this area, which has got insufficient emphasis.

Further study is required to check if the developed technique from this thesis is generalizable. However, this study has established that SSM is legitimate methodology in Computer Science, if it is driven by justified research approach, epistemology, research methodology and methods.

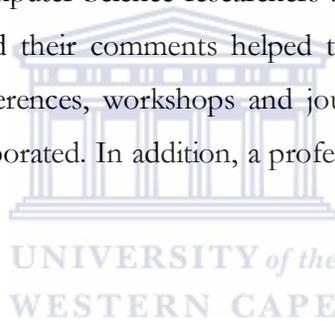
“Well done?”**“Does the thesis reflect seasoned thinking, conveying completeness and thoroughness?”**

The problem of evaluating of software has been tackled holistically using a mixture of research methodologies and methods. The thesis presents a topical subject. In the literature review, (Chapter 2)

recent research in the design and development of framework for evaluation of software has been discussed. Furthermore, the gaps in knowledge about formulation of framework were identified.

Through systematic and cyclical application of SSM, three algorithms for the evaluation of software were developed. Each cycle of SSM addressed some gaps in current knowledge identified as research questions. Those areas that needed further intervention in the next cycle of each algorithm were determined. For example, the inability of AHP (in the first cycle) to handle uncertain user judgement during evaluation of the software were addressed by the development of the Fuzzy AHP (with extent analysis) and Group Fuzzy AHP (in the third and fourth cycle of SSM respectively). Thus, the cyclic intervention in addressing the evaluation of software problem confirms the completeness of the thesis.

A positivist researcher and two Computer Science researchers scrutinized the research findings. The results were presented to them and their comments helped to improve the results. Similarly, the comments obtained from the conferences, workshops and journals where parts of the thesis was presented and published were incorporated. In addition, a professional science editor helped to check the logical flow of ideas.



“Done well?”

“Is the thesis well written? Does it flow logically? Are the central ideas easily accessed?”

Each chapter of the thesis was written and then submitted to the supervisor who edited it after incorporating the suggested changes it was sent to a professional editor. Thus, the thesis is well written. In addition, it flows logically because it has been split into several chapters for easy understanding of the central ideas. Furthermore, each chapter has running heads in the header to direct the reader. In the first Chapter, the research study was contextualised. The background information about the problem of how to evaluate FOSS e-learning systems in a developing country was briefly introduced. The rationale for the research was presented and the problem statement, namely, the evaluation of software under uncertain judgement, was explained. In addition, the research objective – how to develop an evaluation framework for a multi-criteria evaluation problem – was outlined. In order to allow a systematic investigation of the study, research questions were defined.

The questions deal with software quality attributes identification; software quality attributes measurement and framework formulation for software quality.

In Chapter 2, the literature that informs each research question was presented.

Chapter 3 contains the research design and the methodologies undertaken to answer the research questions. The research methods consist of a mix of qualitative and quantitative methods. The qualitative methods included in a case study where participant observation and focus group interviews were used. The quantitative methods entailed the use of questionnaires and different algorithms. The methodologies chosen for data collection for this research were a survey and ethnographic study while SSM was used to manage the analysis. The epistemological stance for the study was *objectivism* while the theoretical perspective for the research approach was *post-positivism*.

Chapter 4 presents the results in terms of the four cycles of SSM.

In Chapter 5, the findings are discussed in terms of the research questions asked, and conclusions are drawn.

Chapter 6 presents how the contributions of this study were evaluated. Possible future study, in the research field, is also mentioned.

This division of the thesis into chapters makes assessing the central ideas to be easy.

“Why now?”

“Is the topic of contemporary interest to scholars in this area?”

Many Free and Open Source e-learning systems are available on the Internet and can be adopted by universities without having to purchase it. The problem is however how does a university decide which quality Free and Open Source e-learning system to adopt? Thus, universities in developing countries face the challenge of evaluating Free and Open Source e-learning system in order to choose a system that is appropriate for its needs.

This thesis emphasis the use of an evaluation technique which allows both technical and non technical IT users to participate in the evaluation. The views of all users are included in the evaluation of

software since the technique accepts verbal descriptions. The verbal description are analysed using fuzzy modeling. The evaluation technique incorporates fuzzy logic into the AHP algorithm.

In the literature review chapter, extensive research was done and several articles and studies consulted and, to our knowledge, there is no study, which suggests the use of the Group Fuzzy AHP algorithm. In addition, the modification of the Fuzzy AHP (with extent analysis) is also new. Moreover, according to the literature the combination of algorithms is unique. The algorithms that were combined are AHP, fuzzy AHP (with extent analysis) and Group Fuzzy AHP. It was developed using for the specific conditions of Tanzania as input, but is applicable to all developing countries with a similar problem (i.e. ICT illiteracy and scarcity of ICT resources, etc.). There is currently research interest in contextualising ICT4D problems in developing countries. The ACM and IEEE both have work groups for ICT4D. They have conferences as well as journals which deal with themes such as ICT4D, Free and open source software, fuzzy systems and multi-criteria decision making.

For example, there is an International Workshop on Technology for Education in Developing Countries - <http://www.cs.joensuu.fi/tedc2006/index.htm>

Also, there are Free open source software conferences such as

- a. FOSS meets academia - <http://ocs.inrialpes.fr/2009/>
- b. The fifth international open source systems (by IFIP) - <http://oss2009.org/> and <http://www.ua.ac.be/main.aspx?c=kris.ven&n=75956>

Lastly, there are conferences and journals, which deal with the fuzzy systems. These are found at <http://iee-cis.org/pubs/>

“Who cares?”

“Do we have any academic community interest in this topic?”

There is a great need for the development of techniques for solving evaluation and selection problems (Chang & Dillon, 2006; Chang et al., 2008; Manieri et al., 2008). The computer societies of academics, scholars and researchers have come up with new approach to address this problem. These new approaches are published in the IEEE computational intelligence journal and IEEE computational

intelligence magazine (Bonissone, Subbu, & Lizzi, 2009). In a recent publication in IEEE's computational magazine the multi-criteria decision making and fuzzy modeling have been identified by researchers as method to solve hard science problems (if it can well be incorporated into decision support system). One of the identified areas is in air transport: the system deals with the real time prediction and detection of storm weather or turbulence disturbance (ibid.).

Limitation of the research study

The limitation of this study and how it can be addressed, was explained in chapter 3. The limitations are there were a limited numbers of participants, data were collect from only one university, data collection methods were susceptible to bias and some of the developed algorithms accepted only exact or crisp values.

More than one method for data collection (triangulation) was used to address the problem of the limited number of participants. Cyclic analysis (using SSM) helped to get more insight about the research problem and this helped to compensate for the problem of only a few organizations being part of the research study. Biasness from the instruments used in data collection was mitigated before conducting the actual study through the use of a pilot study. By means of consistency checking it was possible to evaluate if the user judgement given during evaluation of software, was not contradictory. The limitation of the some of the algorithms (i.e. AHP) to accept only crisp value was controlled through the integration of fuzzy logic and AHP (i.e. Fuzzy AHP).

Suggesting directions for future research

There is a need to develop an ontology of metrics which can incorporate our developed framework. It could answer some of the open questions identified by the Moody (2005) i.e. no consensus on common framework; lack of empirical testing and lack of adoption in practice; different levels of generality; lack of agreement on concepts and terminology; lack of consistency with related fields and standards; lack of measurement and lack of evaluation procedures.

Another avenue for future study is to consider how the results of this study could be used for e-learning software and/or software products in general. It may also be useful to revisit the choice of quality characteristics in terms of the results.

Concluding remarks

The thesis shows that the adoption of FOSS in developing countries can be improved by providing a technique/framework/guideline/roadmap/paradigm, which can aid/guide users in the selection/evaluation of quality software which is usable, maintainable and deployable.

For software developers such evaluation technique could help in prediction of the quality of software. Prediction helps to gain insight and provide strategies to improve the quality of software for developing countries while it enhances software acceptance by the end-user.



BIBLIOGRAPHY

- Abran, A., Bourque, P., Dupuis, R., & Moore, J. W. (2001). *Guide to the Software Engineering Body of Knowledge- SWEBOK*. IEEE Press.
- Ahluwalia, M., Carter, N., & Chenery, H. (1976). Growth and Poverty in Developing Countries. *Journal of Development Economics* , 6.
- Ahmad, A., Basir, O., & Hassanein, K. (2004). Adaptive user interfaces for intelligent e-learning: issues and trends. In: *Proceedings of the fourth international conference on electronic business (ICEB 2004)*, (pp. 925–934). Beijing.
- Alvaro, A., Almeida, E., & Meira, S. (2005). Quality attributes for a component quality model. In *Tenth International Workshop on Component-Oriented Programming*. Glasgow, Scotland: WCOP 2005.
- Alves, C., & Finkelstein, A. (2002). Challenges in COTS decision-making: A goal-driven requirements engineering perspective. in *Workshop on Software Engineering Decision Support, in conjunction with SEKE'02*. Ischia, Italy.
- Andreou, A. S., & Tziakouris, M. (2007). A quality framework for developing and evaluating original software components. *Information and Software Technology* , 49 (2), 122-141.
- Ardito, C., Costabile, M. F., De Angeli, A., & Lanzilotti, R. (2006). Systematic evaluation of e-learning systems: an experimental validation. In A. M. Mørch (Ed.), *In Proceedings of the 4th Nordic Conference on Human-Computer interaction: Changing Roles*. 189, pp. 195-202. Oslo, Norway: ACM Press, New York, NY.
- Bakari, J.K., Mbwete, T.S.A., & Shemwetta, D. (2008). Policies, Master plans and a Rolling Strategic Plan in Effective Implementation of ICT Infrastructure and Services: Case Study of the Open University of Tanzania, URL: http://wikieducator.org/images/5/54/PID_434.pdf
- Bandalaria, M. (2007). Impact of ICTs on Open and Distance Learning in a Developing Country Setting: The Philippine experience. *The International Review of Research in Open and Distance Learning* , 8 (1), [Online] . Available: <http://www.irrodl.org/index.php/irrodl/article/view/334>.
- Basili, V. (1985). Quantitative Evaluation of Software Engineering Methodology. in *Proc. First Pan Pacific Computer Conference*. Melbourne, Australia: University of Maryland.
- Basili, V., Caldeira, G., & Rombach, H. (1994). The Goal Question Metric Approach. In i. J. (ed.) (Ed.). Wiley.
- Baxter, P. & Jak, S. 2008. Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report* 13(4):544-559
- Becker, J. & Niehaves, B. (2007). Epistemological Perspectives on IS Research – A Framework for Analysing and Systematising Epistemological Assumptions, *Information Systems Journal*, 17(2), 197-214.
- Bertoa, M., & Vallecillo, A. (2002). Quality attributes for COTS components. in: *Proceedings of the Sixth ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2002)*, (pp. 54-66). Malaga, Spain.
- Bevan, N. (1999). Quality in use: Meeting user needs for quality. *Journal of Systems and Software* , 49 (1), 89-96.

- Blake, E. (2009). Design doctorate in computing: a defence of "doing cool stuff". In *Proceedings of the 2009 Annual Conference of the Southern African Computer Lecturers' Association (Eastern Cape, South Africa, June 29 - July 01, 2009)*. SACL A '09 (pp. 110-110.). New York, NY: ACM.
- Blin, M., & Tsoukiàs, A. (2001). Multi-Criteria Methodology Contribution to the Software Quality Evaluation. *Software Quality Control* , 9 (2), 113-132.
- Boehm, B. W. (1981). *Software engineering economics*. NJ: Prentice-Hall, Englewood Cliffs.
- Boehm, B. W., Brown, J. R., & Lipow, M. (1976). Quantitative evaluation of software quality. In *Proceedings of the 2nd international Conference on Software Engineering* (pp. 592-605). San Francisco, California, United States: IEEE Computer Society Press, Los Alamitos, CA.
- Boloix, G. (1997). System evaluation and quality improvement. *Journal of Systems and Software* , 36 (3), 297-31.
- Bonissone, P., Subbu, R., & Lizzi, J. (2009). Multi Criteria Decision Making (MCDM): A Framework for Research and Applications. *IEEE Computational Intelligence Magazine* , 4 (3), 48-61.
- Bortolan, G., & Degani, R. (1985). A review of some methods for ranking fuzzy subsets. *Fuzzy Sets and Systems* , 15 (1), 1-19.
- Bourque, P., Dupuis, R., Abran, A., Moore, J., & Tripp, L. (1999). The Guide to the Software Engineering Body of Knowledge. *IEEE Software* , 35-44.
- Bozdag, C., Kahraman, C., & Ruan, D. (2003). Fuzzy group decision making for selection among computer integrated manufacturing systems. *Computers in Industry* , 51 (1), 13-29.
- Brooks, A., Daly, J., Miller, J., Roper, M., & Wood, M. (1996). *Replication of experimental results in software engineering*. ISERN Technical Report ISERN-96-10.
- Bruggink, M. (2003). Open Source in Africa: Towards Informed Decision-Making, *IICD Research Brief* No-7, August 2003.
- Cables, A. U. (2009). *African Undersea Cables*. Retrieved July 2009, from <http://manypossibilities.net/african-undersea-cables/>.
- Câmara, G., & Fonseca, F. (2007). Information policies and open source software in developing countries. *Journal of the American Society of Information Science and Technology* , 58 (1), 121-132.
- Carney, D., & Wallnau, K. (1998). A basis for evaluation of commercial software. *Information and Software Technology* , 40 (14), 851-860.
- Cavano, J. P., & McCall, J. A. (1978). A framework for the measurement of software quality. In S. Jackson, & J. A. Lockett (Ed.), *In Proceedings of the Software Quality Assurance Workshop on Functional and Performance Issues*, (pp. 133-139).
- Chang, C., Wu, C., & Lin, H. (2008). Integrating fuzzy theory and hierarchy concepts to evaluate software quality. *Software Quality Control* , 16 (2), 263-276.
- Chang, D. (1996). Applications of the extent analysis method on fuzzy AHP. *European Journal of Operational Research* , 95 (3), 649-655.

- Chang, E., & Dillon, T. S. (2006). A usability-evaluation metric based on a soft-computing approach. *IEEE Trans. on Sys., Man, and Cyber.*, , 36, 356 - 372.
- Checkland, P. (2000). Soft Systems Methodology: A Thirty Year Retrospective. *Systems Research* , 17, 11– 58.
- Checkland, P., & Scholes, J. (1990). *Soft Systems Methodology in Action*. Toronto: John Wiley and Sons.
- Chen, C.-B., & Klein, C. M. (1997). An efficient approach to solving fuzzy MADM problems. *Fuzzy Sets and Systems* , 88 (1), 51-67.
- Cheng, C., Yang, K. L., & Hwang, C. (1999). Evaluating attack helicopters by AHP based on linguistic variables weight. *European Journal of Operational research* , 116 (2), 423 -435.
- Chin, D. N. (2001). Empirical Evaluation of User Models and User-Adapted Systems. *User Modeling and User-Adapted Interaction* , 11 (1-2), 181-194.
- Chou, S.-Y., Chang, Y.-N., & Shen, C.-Y. (2008). A fuzzy simple additive weighting system under group decision-making for facility location selection with objective/subjective attributes. *European Journal of Operational Research* , 189 (1), 132-145.
- Colace, F., DeSanto, M., & Vento, M. (2003). Evaluating on-line learning platforms: a case study. In *Proceedings of the 36th Annual Hawaii international Conference on System Sciences (Hicss'03) - Track 5 (January 06 - 09, 2003)*. HICSS (pp. 5, 154). Washington, DC: IEEE Computer Society.
- Coleman, D., Ash, D., Lowther, B., & Oman, P. (1994). Using metrics to evaluate software system maintainability. *Computer* , 27 (8), 44-49.
- Comella-Dorda, S., Dean, J. C., Morris, E., & Oberndorf, P. (2002). A Process for COTS Software Product Evaluation. *Proceedings of the Second International Conference on COTS Based Software Systems* (pp. 176-187). Springer-Verlag LNCS 2255.
- Correa, C. (2000). *Intellectual property rights, the WTO and developing countries: The TRIPS Agreement and policy options*. London and New York: Zed Books.
- Côté, M., Suryn, W., & Georgiadou, E. (2007). In search for a widely applicable and accepted software quality model for software quality engineering. *Software Quality Control* , 15 (4), 401-416.
- Covella, G. J., & Olsina, L. A. (2006). Assessing quality in use in a consistent way. In *Proceedings of the 6th international Conference on Web Engineering* (pp. 1-8). Palo Alto, California, USA: ACM Press, New York, NY.
- Crotty, M. (1998). *The foundations of social research: Meaning and perspective in the research process*. London: Sage.
- Creswell, J. 1998. *Research design: Qualitative, quantitative, and mixed methods approaches* (2nd ed.). Thousand Oaks, CA: Sage.
- Creswell, J. W. (2003). *Research Design: Quantitative, Qualitative, and Mixed Methods Approaches*. SAGE. Thousand Oaks. USA.
- Csutora, R., & Buckley, J. J. (2001). Fuzzy hierarchical analysis: the Lambda-Max method. *Fuzzy Sets Syst.* , 120 (2), 181-195.

- Czogala, E., & Pedrycz, W. (1981). Some problems concerning the construction of algorithms of decision-making in fuzzy systems. *International Journal of Man-Machine Studies*, 15 (2), 201-211.
- Davison, R. (2002). Ethics and Research Methods. In *Proceedings of the 35th Annual Hawaii international Conference on System Sciences (Hicss'02)*. 8, p. 253.2. Washington, DC: HICSS,IEEE Computer Society.
- Denning, P. J. (2003). In *Encyclopedia of Computer Science*, A. Ralston, E. D. Reilly, and D. Hemmendinger, Eds. 4th ed. John Wiley and Sons Ltd., Chichester, 405-419.
- Dick, R. (1993). *Subjective Software Measurement*. Technical Report, University of Strathclyde, Dept. of Computer Science, Glasgow.
- Dorairaja, A. (2008). Applying soft systems methodology (ssm) for designing Malay language learning portal, MSc thesis, University of Malaya, Kuala Lumpur
- Dubé, L. & Paré, G. (2003). Rigor in Information Systems Positivist Case Research: Current Practices. *MIS Quarterly*. 27(4).
- Dubois, D. and Prade, H. (1979). Fuzzy real algebra: Some results. *Fuzzy Sets Syst.*, 2 (4), 327-348.
- Eden, A. H. (2007). Three Paradigms of Computer Science. *Minds & Machines*, 17, 135-167
- Fagerholm, F. (October 9, 2007). *Measuring and tracking quality factors in Free and Open Source Software projects*. Master's Thesis, University of Helsinki.
- Feller, J., & Fitzgerald, B. (2000). A framework analysis of the open source software development paradigm. In *Proceedings of the Twenty First international Conference on information Systems (Brisbane, Queensland, Australia)* (pp. 58-69). Atlanta, GA: Association for Information Systems.
- Fenton, N. E., & Neil, M. (2000). Software metrics: roadmap. In *Proceedings of the Conference on the Future of Software Engineering (Limerick, Ireland, June 04 - 11, 2000), ICSE '00* (pp. 357-370). New York, NY: ACM.
- Fenton, N. E., & Pfleeger, S. (1997). *Software Metrics: A Rigorous and Practical Approach* (2nd Edition ed.). PWS.
- Fitzgerald, B. & Howcroft, D. (1998). Towards Dissolution of the IS Research Debate: From Polarisation to Polarity, *Journal of Information Technology*, 13 (4), 313-326
- Finnie, G. R., Wittig, G., & Petkov, D. (1993). Prioritizing software development productivity factors using the analytic hierarchy process. *Journal of Systems and Software*, 22 (2), 129-139.
- Forman, E., & Peniwati, K. (1998). Aggregating individual judgments and priorities with the analytic hierarchy process. *European Journal of Operational Research*, 108 (1), 165-169.
- Franch, X., & Carvallo, J. P. (2003). Using Quality Models in Software Package Selection. *IEEE Software*, 20 (1), 34-41.
- Fuggetta, A. (2003). Open source software--an evaluation. *Journal of Systems and Software*, 66 (1), 77-90.
- Gable, G. G. (1994). Integrating case study and survey research methods: an example in information systems. *European Journal of Information Systems* 3(2),112-126.

- Garcia, F., Piattini, M., Ruiz, F., & Visaggio, C. A. (2005). Maintainability of Software Process Models: An Empirical Study. *Ninth European Conference on Software Maintenance and Reengineering (CSMR'05)* (pp. 246-255). csmr.
- Garvin, D. (1984). What does product quality really mean?, in *MIT Sloan Management Review*. Boston: Massachusetts Institute of Technology, 26, 25 - 43.
- Gerea, M. (2006). *Selection and Evaluation of Open Source Components*. part of course TDT4735 Depth Project in Software Engineering(15th Dec.), Department of Computer and Information Science. Norwegian University of Science and Technology (NTNU): <http://www.idi.ntnu.no/grupper/su/fordypningsprosjekt-2006/gereafordyp06.pdf>.
- Gibbons, M., Limoges, C., Nowotny, H., Schwartzman, S., Scott, P., & Trow, M. (1994). *The new production of knowledge: the dynamics of science and research in contemporary societies*. London: Sage.
- Golany, B., & Kress, M. (1993). A multicriteria evaluation of methods for obtaining weights from ratio-scale matrices. *European Journal of Operational Research* , 69 (2), 210-220.
- Graf, S., & List, B. (2005). An Evaluation of Open Source E-Learning Platforms Stressing Adaptation Issues. In *Proceedings of the Fifth IEEE international Conference on Advanced Learning Technologies(July 05 - 08, 2005)* (pp. 163-165). Washington: IEEE Computer Society.
- Graham, C. (2006). *Blended Learning Systems: Definition, Current Trends, and Future Directions; in Handbook of Blended Learning: Global Perspectives, Local Designs.*, Pfeiffer Publishing, San Francisco, USA.
- Gregor, S. (2006). The Nature of Theory in Information Systems. *MIS Quarterly* , 3 (30), 611-642.
- Hayes, J. H., & Zhao, L. (2005). Maintainability Prediction: A Regression Analysis of Measures of Evolving Systems. In *Proceedings of the 21st IEEE international Conference on Software Maintenance (September 25 - 30, 2005)*. ICSM (pp. 601-604). IEEE Computer Society, Washington, DC.
- Heeks, R. (2008). ICT4D 2.0: The Next Phase of Applying ICT for International Development. *IEEE Computer* , 41 (9), 26-33.
- Hersh, M. A., & Tucker, W. D. (2005). Ethics and Mono-disciplinarily: Positivism, Informed Consent and Informed Participation. in *Proc. 16th IFAC World Congress*. Prague, Czech Republic.
- Hirschheim, R. A. (1992). 'Information Systems Epistemology: an Historical Perspective'. *Information Systems Research: Issues, Methods and Practical Guidelines*. Galliers, R. (Ed). Blackwell Scientific Publications, Oxford: 28-60.
- IEEE. (1998). *Std. 1061-1998 IEEE standard for a software quality metrics methodology*.
- IMF. (2009). *International Monetary Fund*. Retrieved July 20, 2009, from How does the WEO categorize advanced versus emerging and developing economies?: <http://www.imf.org/external/pubs/ft/weo/faq.htm#q4b>
- Isamuyo, Z. (2006). *Open access ICT infrastructure in rural Tanzania: prototype design*. Masters Thesis, Royal Institute of Technology (KTH), Stockholm, Sweden.
- ISO. (2001). *Software Engineering - Product Quality - Part 1: Quality model*. ISO/IEC 9126-1. Geneva: International Organization for Standardization.

- Johnson, R. B., & Onwuegbuzie, A. J. (2004). Mixed methods research: A research paradigm whose time has come. *Educational Researcher*, 33 (7), 14-26.
- Jørgensen, M. (1999). Software quality measurement. *Adv. Eng. Softw.*, 30, 907-912.
- Kampenès, V. (2007). *Quality of design, analysis and reporting of software engineering experiments a systematic review*. Ph.D. Thesis, University of Oslo, Department of Informatics, Norway.
- Kan, S. H. (2003). *Metrics and Models in Software Quality Engineering*, Second ed.: AddisonWesley.
- Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and Software Technology*, 39 (14-25), 939-947.
- Karsten, R., & Garvin, T. (1996). The use of the analytic hierarchy process in the selection of participants for a telecommuting pilot project. In *Proceedings of the 1996 ACM SIGCPR/SIGMIS Conference on Computer Personnel Research (Denver, Colorado, United States, April 11-13, 1996)* (pp. 152-160). New York, NY: ACM Press.
- Khosravi, K., & Guéhéneuc, Y. (July 2005). Open Issues with Quality Models. In Fernando Brito e Abreu, Coral Calero, Michele Lanza, Geert Poels, & Houari A. Sahraoui (Ed.), *Proceedings of the 9th ECOOP workshop on Quantitative Approaches in Object-Oriented Software Engineering*. Springer-Verlag.
- Kinuthia, W., & Dagada, R. (2008). E-learning incorporation: an exploratory study of three South African higher education institutions. *International Journal on E-Learning*, 7 (4), 623-639.
- Kitchenham, B. (2008). The role of replications in empirical software engineering--a word of warning. *Empirical Softw. Engg.*, 13 (2), 219-221.
- Klein, H. K., & Myers, M. D. (1999). A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS Q.*, 23 (1), 67-93.
- Koscianski, A., & Costa, J. C. (1999). Combining Analytical Hierarchical Analysis with ISO/IEC 9126 for a Complete Quality Evaluation Framework. *Fourth IEEE International Symposium and Forum on Software Engineering Standards*, (p. 218).
- Kunda, D. (2001). *A socio-technical approach to selecting software supporting COTS- Based Systems*. PhD thesis, University of York, Department of computer science.
- Kwong, C., & Bai, H. (2002). A fuzzy AHP approach to the determination of importance weights of customer requirements in quality function deployment. *Journal of Intelligent Manufacturing*, 13 (5), 367-377.
- Lai, S.-K. (1995). A preference-based interpretation of AHP. *Omega*, 23 (4), 453-462.
- Lee, M., & Jefferson, T. L. (2005). An Empirical Study of Software Maintenance of a Web-Based Java Application. *21st IEEE International Conference on Software Maintenance (ICSM'05)* (pp. 571-576). icsm.
- Leech, N. L., & Onwuegbuzie, A. J. (2005). *A typology of mixed methods research designs*, Quality & Quantity, 43 (2), 265-275.
- Leung, L. C., & Cao, D. (2000). On consistency and ranking of alternatives in fuzzy AHP. *European Journal of Operational Research*, 124 (1), 102-113.

- Levitin, A. (2003). *Introduction to the Design & Analysis of Algorithms*. Addison-Wesley.
- Li, P. L., Herbsleb, J., & Shaw, M. (2005). Finding Predictors of Field Defects for Open Source Software Systems in Commonly Available Data Sources: A Case Study of OpenBSD. *11th IEEE International Software Metrics Symposium (METRICS'05)* (p. 32). metrics.
- Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Beverly Hills, CA: Sage Publications.
- Liou, T.-S., & Wang, M.-J. (1992). Ranking fuzzy numbers with integral value. *Fuzzy Sets and Systems*, 50 (3), 247-255.
- Lucca, G. A. (2004). Towards the Definition of a Maintainability Model for Web Applications. *Eighth Euromicro Working Conference on Software Maintenance and Reengineering (CSMR'04)*, csmr, (p. 279).
- Manieri, A., Begin, M., Meglio, A. D., & Rippa, A. (2008). Automation in quality assurance environments. *The First International Conference on Open Source, Qualipso Conference 2008*. Rome, Italy .
- Mayes, J., & Fowler, C. (1999). Learning technology and usability: A framework for understanding courseware. *Interacting with Computers*, 11 (5), 485-497.
- McCarty, S. (2005). Cultural, disciplinary and temporal contexts of e-learning and English as a foreign language. *eLearn 2005*, 4 (1).
- McHarazo, A., & Olden, A. (2000). Fulfilling the learning resource requirements of students at the Open University of Tanzania. *Journal of Librarianship and Information Science*, 32 (4), 204-214.
- Mebrate, T. (2010). A framework for evaluating academic website's quality from students' perspective, MSc. thesis, Department: Computer Science/Information Architecture, Faculty: Electrical Engineering, Mathematics and Computer Science, Delft University of technology
- Mehra, B., Merkel, C., & Bishop, A. P. (2004). The Internet for Empowerment of Minority and Marginalized Users. *New Media & Society*, 6 (6), 781-802.
- Mikhailov, L. (2003). Deriving priorities from fuzzy pairwise comparison judgments. *Fuzzy Sets and Systems*, 134 (3), 365 - 385.
- Mikhailov, L. (2002). Fuzzy analytical approach to partnership selection in formation of virtual enterprises. *Omega*, 30 (5), 393-401.
- Mikhailov, L., & Tsvetinov, P. (2004). Evaluation of services using a fuzzy analytic hierarchy process. *Applied Soft Computing*, 5 (1), 23-33.
- Mmari, G.R.V. (1997). Library services for the Open University of Tanzania: experiences of the first year. In Watson, E.F. & Jagannathan, N. (eds.) *Library services to distance learners in the Commonwealth: a reader*. Vancouver: the Commonwealth
- Moody, D. L. (2005). Theoretical and practical issues in evaluating the quality of conceptual models: Current state and future directions. *Data and Knowledge Engineering*, 55 (3), 243-276.
- Moses, J., & Farrow, M. (2008). Tests for consistent measurement of external subjective software quality attributes. *Empirical Softw. Engg.*, 13 (3), 261-287.
- Mutagahywa, B., & et.al. (2003). *Application and Institutional Integration of Distance Education and ICTs at University of Dar Es Salaam*. Consultative meeting on the role of modern distance education and ICTs

- in enhancing access to quality tertiary Education in Sub-Sahara Africa. (Retrieved on 10th May 2007 from World Wide Web): http://www.worldbank.org/afr/de_ict/ghana_0903/dar_ICT_pres.pdf.
- Myers, M. (1999). Investigating information systems with ethnographic research. *Commun. AIS* 2, 4es, Article 1
- Myers, M. D. & Newman, M. (2007). The qualitative interview in IS research: Examining the craft. *Inf. Organ*, 17 (1), 2-26.
- Nichols, D., & Twidale, M. B. (2003). The usability of open source software. *First Monday* , 8 (1).
- Niehaves, B. (2005). Epistemological Perspectives on Multi-Method Information Systems Research, Thirteenth European Conference on Information Systems, Regensburg, Germany
- Olivier, M. S. (2004). *Information Technology Research: A Practical Guide for Computer Science and Informatics* (Second ed.). Pretoria: Van Schaik Publishers.
- Oman, P., & Hagemester, J. (1992). Metrics fo Assessing a Software System's Maintainability. *Proceedings of IEEE International Conference on Software Maintenance* (pp. 337-344). Los Alamitos, CA.: IEEE Computer Society Press.
- O'Reilly, T. (1999). Lessons from open-source software development. *Commun.* , 42 (4), 32-37.
- Ossadnik, W., & Lange, O. (1999). AHP-based evaluation of AHP-Software. *European Journal of Operational Research* , 118 (3), 578-588.
- Osterweil, L. (Dec. 1996). Strategic directions in software quality. *ACM Comput. Surv.* , 28 (4), 738-750.
- Ota, K., Taira, N., & Miyagi, H. (2008). Group Decision-Making Model in Fuzzy AHP Based on the Variable Axis Method. *IEEEJ Transactions on Electronics, Information and Systems* , 128 (2), 303-309.
- OUT e-learning, (2010). URL: <http://elms.out.ac.tz/>
- OUT Website, (2010). URL: <http://www.out.ac.tz>
- Padayachee, I., Kotze, P. & van der Merwe, A. (2010). ISO 9126 external systems quality characteristics, sub-characteristics and domain specific criteria for evaluating e-Learning systems. In: Proceedings of SACLA 2010 Conference.
- Paul, R. A., Kunii, T. L., Shinagawa, Y., & Khan, M. F. (1999). Software Metrics Knowledge and Databases for Project Management. *IEEE Trans. on Knowl. and Data Eng.* , 11 (1), 255-264.
- Pedrycz, W. (2002). Computational Intelligence and visual computing: an emerging technology for software engineering. *Soft Computing - A Fusion of Foundations, Methodologies and Applications* , 7 (1), 33-44.
- Polancic, G., Horvat, R., & Rozman, T. (2004). Comparative assessment of open source software using easy accessible data. *26th International Conference on Information Technology Interfaces*. 1, pp. 673-678. IEEE Society Press.
- Pressman, R. S. (c2005). *Software Engineering: A Practitioner's approach*. (5th, Ed.) Boston : McGraw-Hill Higher Education.

- Pruengkarn, R., Praneetpolgrang, P., & Srivihok, A. (2005.). An Evaluation Model for e-Learning Websites in Thailand University. *Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05)* (pp. 161-162). ACM.
- Pscheidt, M. (2008). Sustainability Factors for Information Systems in Developing Countries – Academic Registry Information System in Mozambique, 5th Prato Community Informatics & Development Informatics Conference 2008: ICTs for Social Inclusion: What is the Reality? 27 October-30 October, Monash Centre, Plato Italy
- Ramesh, V., Robert Glass, L. , Vessey, Iris. (2004). Research in computer science: an empirical study, *Journal of Systems and Software*, 70 (1-2), 165-176.
- Rand, A. (1979). *Introduction to Objectivist Epistemology*. New York: Meridian.
- Rodriguez, D., Sicilia, M., Cuadrado, J., & Pfahl, D. (2006). e-Learning in Project Management using Simulation Models: A Case Study based on the Replication of an Experiment. *IEEE Transactions on Education. IEEE Education Society* , 49 (4), 451-463.
- Ross, T. J. (2004). *Fuzzy logic with engineering applications (second ed.)*. Wiley.
- Rothfuss, G. (2002). *A framework for Open Source projects*. Master Thesis in Computer Science, University of Zurich, Department of Information Technology.
- Runeson, P. & Höst, M. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*. 14, 131-164.
- Ryu, Y. (2005). *Development of questionnaires for electronic mobile products and decision making methods*. PhD Thesis, Virginia Polytechnic Institute and State University.
- Saaty, T. L. (1994). How to make a decision: the analytic hierarchy process. *Interfaces* , 24 (6), 19–43.
- Saaty, T. L., & Tran, L. T. (2007). On the invalidity of fuzzifying numerical judgments in the Analytic Hierarchy Process. *Mathematical and Computer Modelling, Decision Making with the Analytic Hierarchy Process and the Analytic Network Process* , 46 (7-8), 962-975.
- Saaty, T. (1980). *The analytic hierarchy process: Planning, priority setting, resource allocation*. New York: McGraw-Hill International.
- Sambrook, S. (2003). E-learning in small organisations. *Education and Training* , 45 (8), 506-516.
- Samoladas, I., Stamelos, I., Angelis, L., & Oikonomou, A. (2004). Open source software development should strive for even greater code maintainability. *Commun. ACM (Oct. 2004)* , 47 (10), 83-87.
- Sanga, C., & Venter, I. M. (2009). Is a multi-criteria evaluation tool reserved for experts? *The Electronic Journal of Information Systems Evaluation (EJISE)* , 12 (2), 165 – 176.
- Sharp, H., Rogers, Y. & Preece, J. (2007). Interaction Design - beyond human-computer interaction (2nd Edition ed., pp. 181–217). John Wiley & Sons.
- Schneidewind, N. F. (2002). Body of knowledge for software quality measurement. *IEEE Computer* , 35 (2), 77-83.

- Seffah, A., Kececi, N., & Donyaee, M. (2001). QUIM: A Framework for Quantifying Usability Metrics in Software Quality Models. *Second Asia-Pacific Conference on Quality Software (APAQ'S'01)*, (p. 0311).
- Seidel, J. V. (1998). *Qualitative data analysis*. Retrieved September 29, 2008, from In The Ethnograph v5 user's manual: www.qualisresearch.com
- Shee, D. Y., & Wang, Y.-S. (2008). Multi-criteria evaluation of the web-based e-learning system: A methodology based on learner satisfaction and its applications. *Computers & Education*, 50 (3), 894-905.
- Shull, F. (1998). *Developing Techniques for Using Software Documents: A Series of Empirical Studies*. College Park. University of Maryland.
- Shull, F. J., Carver, J. C., Vegas, S., & Juristo, N. (2008). The role of replications in Empirical Software Engineering. *Empirical Softw. Engg.*, 13 (2), 211-218.
- Sommerville, I. (2004). *Software Engineering (International Computer Science Series)* (6th ed.). Addison-Wesley.
- Soriyan, H. A., Korpela, M. J., Mursu, A. S., & Kailou, K. (1999). Information Systems Development for Healthcare in Africa: The INDEHELA-Methods Project. *3rd International Conference on Health Informatics in Africa, HELINA'99*. Harare, Zimbabwe: <http://www.uku.fi/atkk/indehela/inde-h99.pdf>.
- South African Boards of Standards. (2008). *Conformity, Assessment and Accreditation*. Retrieved 09 14, 2009, from SABS: <https://www.sabs.co.za/index.php?page=accreditations>
- Srdjevic, B. (2005). Combining different prioritization methods in the analytic hierarchy process synthesis. *Comput. Oper. Res.*, 32 (7), 1897-1919.
- Stevens, S. (1946). On the theory of scales of measurement. *Science*, 103, 677-680.
- Tan, Y., & Mookerjee, V. S. (2005). Comparing uniform and flexible policies for software maintenance and replacement. *IEEE Transactions on Software Engineering*, 31 (3), 238-255.
- Tang, H., & Zhang, J. (2007). Study on Fuzzy AHP Group Decision-Making Method Based on Set-Valued Statistics. *FSKD* (3), 689-693.
- Tedre, M. (2007). Know Your Discipline: Teaching the Philosophy of Computer Science. *Journal of Information Technology Education*, 6 (1), 105-22.
- Tedre, M. (2006). *The Development of Computer Science: A Sociocultural Perspective*. Finland: University of Joensuu.
- Tedre, M., Bangu, N., & Nyagava, S. I. (2009). Contextualized IT Education in Tanzania: Beyond Standard IT Curricula. *Journal of Information Technology Education*, 8, 101-124.
- Teknomo, K. (2006). *Analytic hierarchy process (AHP) tutorial*. Retrieved from <http://people.revoledu.com/kardi/tutorial/ahp/>

- Triantaphyllou, E., Kovalerchuk, B., Mann Jr. L., & Knapp, G. (1997). Determining the most important criteria in maintenance decision making. *Journal of Quality in Maintenance Engineering* , 3 (1), 16–28.
- Trochim, W., & Donnelly, J. (2001). *The Research Methods Knowledge Base*. . Atomic Dog Publishing.
- Turban, E. (1993). *Decision Support and Expert Systems: Management Support System* (third ed.). New York: Macmillan.
- Van Vuuren, D., & Maree, A. (1999). Survey methods in market and media research: Applied methods for the Social Sciences. (M. Terre Blanche, & K. Durrheim, Eds.) *Research in Practice* , 281.
- Venter, I. (2000). *Group constitution for small group learning in the field of information technology*. Faculty of engineering, Built Environment and Information Technology. Cape Town: University of the Pretoria.
- Vila, J., & Beccue, B. (1995). Effect of visualization on the decision maker when using analytic hierarchy process. *28th Hawaii International Conference on System Sciences, hicc* (p. 992).
- Vlahavas, I., Stamelos, I., Refanidis, I., & Tsoukias, A. (1999). ESSE: an expert system for software evaluation. *Knowledge-Based Systems* , 12 (4), 183-197.
- Wang, H. F., & Fu, C. C. (1997). A generalization of fuzzy goal programming with preemptive structure. *Computers & Operations Research* , 24 (9), 819-828.
- Wang, Y., & Chin, K. (2008). A linear goal programming priority method for fuzzy analytic hierarchy process and its applications in new product screening. *Int. J. Approx. Reasoning* , 49 (2), 451-465.
- Wang, Y.-M., Luo, Y., & Hua, Z. (2008). On the extent analysis method for fuzzy AHP and its applications. *European Journal of Operational Research* , 186 (2), 735-747.
- Warfield, D. M.S. (2010). IS/IT research: a research methodologies review, *Journal of Theoretical and Applied Information Technology*, 13 (1) , 28-35
- Wesson, J. (2003). Usability issues for e-commerce and e-learning: a developing country perspective. *Proceeding of the International workshop on Utility, Usability and Complexity of e-Information systems*, (pp. 43-55).
- Whetten, D. A. (1989). What constitutes a theoretical contribution? *Academy of Management Review* , 14, 490–495.
- Wu, D. (2009). *Intelligent systems for decision support*. University of Southern California. http://www-scf.usc.edu/~dongruiw/files/PhD%20Dissertation_WU.pdf.
- Yager, R. R. (1988). On ordered weighted averaging aggregation operators in multi-criteria decision making. *IEEE Transactions on Systems, Man and Cybernetics* , 18, 183-190.
- Yaghini, M., Bourouni, A. & Amiri, R.H. (2009). A Framework for Selection of Information Systems Development Methodologies, *Computer and Information Science*, 2 (1), 3-11
- Yeh, C. H., & Deng, H. (2004). A practical approach to fuzzy utilities comparison in fuzzy multicriteria analysis. *International Journal of Approximate Reasoning* , 35 (2), 179-194.

- Yin, R. (1994). *Case Study Research: Design and Methods* (Second ed.). Thousand Oaks: Sage Publications.
- Yin, R.K. (2002). *Case Study Research: Design and Methods* (3rd ed.). Thousands Oaks, CA: Sage Publications, p. 86.
- Yin, R.K. 2003. *Case study research. Design and methods*. London, Sage, 3rd edition.
- Yu, L., Schach, S., & Chen, K. (November 2005). Measuring the maintainability of open-source software. *in: Proceedings of the 4th International Symposium on Empirical Software Engineering, 287-293*. Noosa Heads, Queensland, Australia.
- Zadeh, L. A. (1994). Fuzzy logic, neural networks, and soft computing. *Commun. ACM* , 37 (3), 77-84.
- Zadeh, L. (1965). Fuzzy sets, *Information and Control*. 8 (3), 338-353.
- Zadeh, L. (1983). The role of fuzzy logic in the management of uncertainty in expert systems. *Fuzzy Sets and Systems* , 11 (1-3), 197-198.
- Zadeh, L. (2008). Toward human level machine intelligence – Is it achievable? - The need for a paradigm shift. *IEEE Computational Intelligence Magazine* , 3 (3), 11–22.
- Zaharias, P. (2004). Usability and e-learning: the road towards integration. *eLearn* , 6 (4).
- Zhang, D., Zhao, J. L., Zhou, L., & Nunamaker, J. F. (2004). Can e-learning replace classroom learning? *Commun.* , 47 (5), 75-79.
- Zhu, K., Jing, Y., & Chang, D. (1999). A discussion on extent analysis method and applications of fuzzy AHP. *European Journal of Operational Research* , 116, 450-456.

APPENDICES



APPENDIX A

AHP

AHP is a multi-criteria decision-making (MCDM) algorithm that uses pairwise comparisons to derive weights of importance from a multi-level hierarchical structure of objectives, characteristics, sub-characteristics and alternatives depending on the problem (Saaty, 1980). In cases where the comparisons are not perfectly consistent, AHP provides an uncomplicated method for improving the consistency of the comparisons, by using the Eigenvalue method and consistency checking method. The hierarchical structure fit well with the hierarchical structure of the software quality model. This is the reason why it was chosen to be used in the analysis.

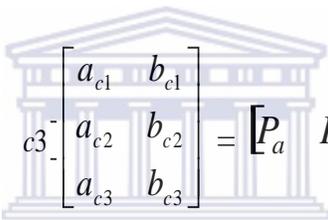
According to Saaty (1980) and Finnie et al. (1993), the analytic hierarchy process algorithm has the following steps:

- (i) Define the unstructured problem and state clearly the goal/objectives and outcomes;
- (ii) Decompose the complex problem into a hierarchical structure of goal, characteristics, sub-characteristics, attributes and alternatives;
- (iii) Employ pairwise comparisons and form pair-wise comparison matrices;
- (iv) Use the Eigenvalue method to estimate the relative weights;
- (v) Check the consistency of the decision judgements;
- (vi) Aggregate the relative weights to obtain the overall rating for alternatives.

These steps of the algorithm can be summarized into three (Saaty, 1980; Vila & Beccue, 1995; Finnie, Wittig & Petrov, 1993): **Firstly**, the problem is decomposed into a number of hierarchical levels. The objective is the highest level, the decision criteria are at the next level, and sub-criteria and the decision alternatives (under each criterion) are at the lowest level of the hierarchy. In our case, (see Appendix D) the multi-criteria evaluation problem was translated into this hierarchical level approach: with the first level being the evaluation of e-learning systems, etc. **Secondly**, a pairwise comparison matrix (using the pairwise comparison (or judgements) of the decomposed hierarchical levels) is formed. This step reduces the complexity of the multi-criteria multi-decision to a simple set of pairwise

comparisons. A rating scale is used to indicate the preference of one criterion over another instead of comparing all alternatives (with respect to all criteria) simultaneously. **The third** step is called synthesization. It is obtained when the relative importance of each level and each element within the level is merged in order to obtain a prioritization alternative (Karsten & Garvin, 1996).

To summarise: assume you have a hierarchical model of two alternatives **a** and **b** with respect to a specific objective, which must be evaluated. Let's assume the weight of criterion 1 is c_1 , the weight of criterion 2 is c_2 and that of criterion 3 is c_3 . If the priority of alternatives **a** and **b** with respect to criterion 1 is (a_{c_1}, b_{c_1}) and the priorities for criterion 2 and criterion 3 are (a_{c_2}, b_{c_2}) and (a_{c_3}, b_{c_3}) alternatively then equation 1 shows the mathematical expression for computing the priorities for the alternative **a** and **b**.



$$\begin{bmatrix} 1 & c_2 & c_3 \\ a_{c_1} & b_{c_1} \\ a_{c_2} & b_{c_2} \\ a_{c_3} & b_{c_3} \end{bmatrix} = \begin{bmatrix} P_a & P_b \end{bmatrix}$$

The priority with the highest value is taken as the best alternative.

The pairwise comparison process is done at each level to bring forth preferences for each decision criterion with respect to another decision criterion. As mentioned before, human judgement is associated with inconsistency, so what can be done to prove that the results of such subjective evaluations are consistent? Consistency deals with how the users (i.e. evaluators) relate their experiences during the evaluation judgements. This is done to avoid contradictions or difference or conflicts of the user judgement. Proving consistency of the evaluation judgements of different evaluators was important in order to visualize the trend of data if they were pointing to the right direction (i.e. association of the dataset). AHP checks for consistency of judgement using mathematical formulae as illustrated in the Appendix G.

The following section gives steps for weight vector computation in AHP (Saaty, 1980; Saaty, 1994).

Problem structuring

According to Vila and Beccue (1995), the first step for AHP is to decompose a problem into a number of hierarchical levels. At the highest level of the hierarchy the objectives is placed, then decision criteria are at the next level, and sub-criteria and decision alternatives (under each criterion) are at the lowest level of the hierarchy. Each of the criteria and sub-criteria is normally associated with a weight that indicates its significance in relation to other criteria. The following is an illustrative example of a hierarchical model of two alternatives A and B with respect to a specific objective to be evaluated (see Figure 23). Let's assume the weights of the criteria c_1 , c_2 , and c_3 , respectively. Furthermore, let the priority of alternatives A and B with respect to criterion 1 be a_{c_1} and b_{c_1} , similarly a_{c_2} and b_{c_2} for criterion 2 and a_{c_3} and b_{c_3} for criterion 3, respectively.

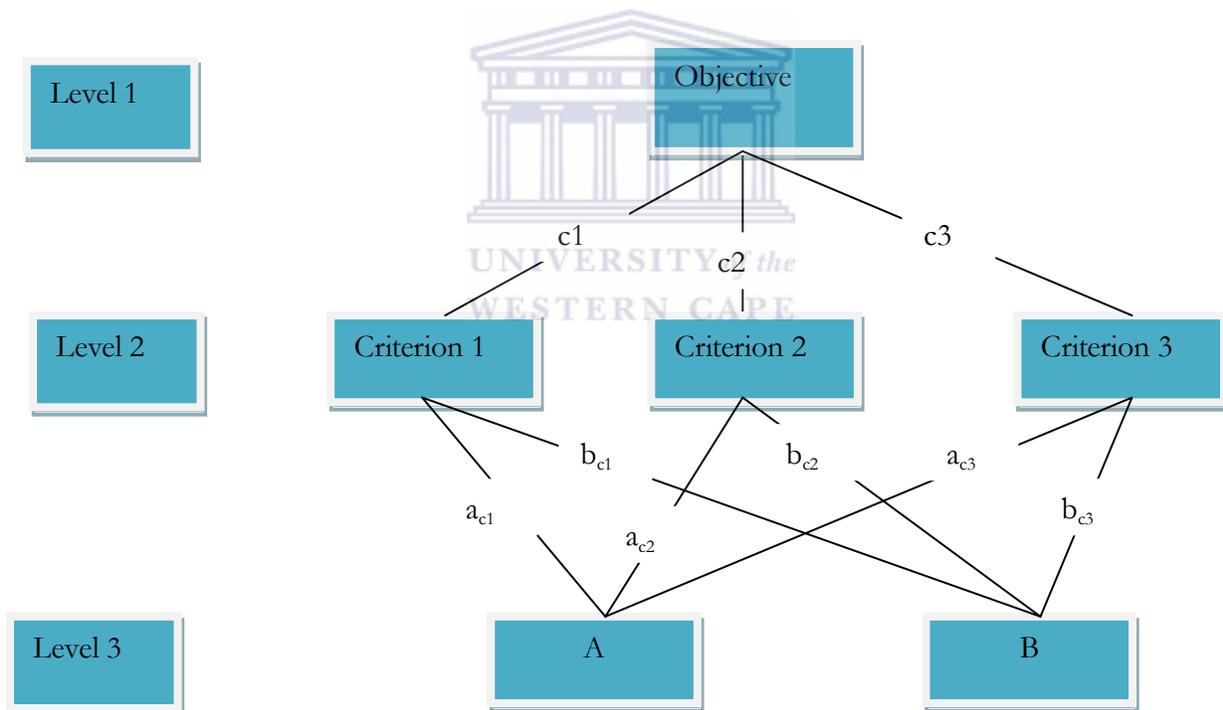


Figure 23: Simple hierarchical structure

Assessment of Local Priorities

Assessment of priorities is performed through pair-wise comparison of the alternatives, using the Saaty nine-point scale (see Table 18). This results in so called, comparison matrices of pairwise comparisons of alternatives at the same hierarchy level.

The pairwise comparisons between elements that are in the same level of hierarchy are done using Saaty scale.

Intensity of importance	Definition
1	Equal importance
3	Moderate importance of one over another
5	Essential or strong importance
7	Very strong or demonstrated importance
9	Absolute importance
2,4,6,8	Intermediate values between adjacent scale values

Table 18: Saaty scale

Saaty (1980) proposes the Eigenvalue method to compute pairwise comparison matrix.

A is the value for comparison of i over j.

$$A = a_{ij} \tag{1}$$

$$a_{ij} = \frac{w_i}{w_j} \tag{2}$$

$$\sum_{j=1}^n a_{ij} w_j = n w_i \tag{3}$$

For $i=j=1$

$$A w = n w \tag{4}$$

Maximum Eigenvalue (λ_{max})

$$A w = \lambda_{max} w \text{ where } \lambda_{max} \geq n \tag{5}$$

It is this step of assessing local priorities, in which the complexity of the multi-criteria multi-decision alternative problem is reduced to simple pairwise comparisons (Vila & Beccue, 1995). A rating scale gives preference to one criterion over another (see Table 18). This makes the process easier than if comparison were to be done by comparing all alternatives with respect to all criteria simultaneously.

The first pairwise comparison matrix is obtained by comparison of criteria with respect to the objective.

Objective function constraint matrix

$$\begin{array}{l}
 \text{criterion 1} \\
 \text{criterion 2} \\
 \text{criterion 3}
 \end{array}
 \begin{array}{l}
 \frac{c_1}{c_1} a_{11} \quad \frac{c_1}{c_2} a_{12} \quad \frac{c_1}{c_3} a_{13} \\
 \frac{c_2}{c_1} a_{21} \quad \frac{c_2}{c_2} a_{22} \quad \frac{c_2}{c_3} a_{23} \\
 \frac{c_3}{c_1} a_{31} \quad \frac{c_3}{c_2} a_{32} \quad \frac{c_3}{c_3} a_{33}
 \end{array}$$

(6)

Then three pairwise comparison matrices are computed with respect to the three criteria.

Pairwise matrix for alternative A and B with respect to criterion 1

<i>criteria</i>	A	B
A	$\frac{a_{c1}}{a_{c1}}$	$\frac{a_{c1}}{b_{c1}}$
	$\frac{a_{c1}}{b_{c1}}$	$\frac{b_{c1}}{b_{c1}}$
B	$\frac{b_{c1}}{a_{c1}}$	$\frac{b_{c1}}{b_{c1}}$



(7)

Pairwise matrix for alternative A and B with respect to criterion 2

<i>criteria 2</i>	A	B
A	$\frac{a_{c2}}{a_{c2}}$	$\frac{a_{c2}}{b_{c2}}$
	$\frac{a_{c2}}{b_{c2}}$	$\frac{b_{c2}}{b_{c2}}$
B	$\frac{b_{c2}}{a_{c2}}$	$\frac{b_{c2}}{b_{c2}}$

(8)

Pairwise matrix for alternative A and B with respect to criterion 3

<i>criteria 3</i>	A	B
A	$\frac{a_{c3}}{a_{c3}}$	$\frac{a_{c3}}{b_{c3}}$
	$\frac{a_{c3}}{b_{c3}}$	$\frac{b_{c3}}{b_{c3}}$
B	$\frac{b_{c3}}{a_{c3}}$	$\frac{b_{c3}}{b_{c3}}$

(9)

After obtaining the pairwise comparison matrices, the next step is to normalise the matrices with respect to the weights, and then obtain the principal eigenvector – which is the priority vector. We do this process for level 1, and then for the level 2. Then we compute the overall composite weight for the alternatives A and B, which is a normalization of weighted priority vectors.

Normalization is obtained first by computing the total of each column j in pairwise comparison matrix (Saaty, 1980) and then each row in a column j is divided by the column total ($\sum_{i=1}^n a_{i j}$).

Normalization is given by the following expression:

$$a_{ij}^{\bar{}} = \frac{a_{ij}}{\sum_{i=1}^n a_{i j}}, i, j=1, 2, \dots, n, \quad (10)$$

Objective criteria priority vector

$$\begin{matrix} \text{criteria 1} & \frac{a_{11}}{\sum_{i=1}^3 a_{i1}} & \frac{a_{12}}{\sum_{i=1}^3 a_{i2}} & \frac{a_{13}}{\sum_{i=1}^3 a_{i3}} \\ \text{criteria 2} & \frac{a_{21}}{\sum_{i=1}^3 a_{i1}} & \frac{a_{22}}{\sum_{i=1}^3 a_{i2}} & \frac{a_{23}}{\sum_{i=1}^3 a_{i3}} \\ \text{criteria 3} & \frac{a_{31}}{\sum_{i=1}^3 a_{i1}} & \frac{a_{32}}{\sum_{i=1}^3 a_{i2}} & \frac{a_{33}}{\sum_{i=1}^3 a_{i3}} \end{matrix} \quad (11)$$

After normalization, the sum of each column (j) of the matrix is equivalent to 1.

$$\sum_{i=1}^n a_{i j} = 1$$

Objective criteria weight

$$\begin{matrix} \text{criteria 1} & \frac{a_{11}}{a_{11} + a_{21} + a_{31}} & \frac{a_{12}}{a_{12} + a_{22} + a_{32}} & \frac{a_{13}}{a_{13} + a_{23} + a_{33}} \\ \text{criteria 2} & \frac{a_{21}}{a_{11} + a_{21} + a_{31}} & \frac{a_{22}}{a_{12} + a_{22} + a_{32}} & \frac{a_{23}}{a_{13} + a_{23} + a_{33}} \\ \text{criteria 3} & \frac{a_{31}}{a_{11} + a_{21} + a_{31}} & \frac{a_{32}}{a_{12} + a_{22} + a_{32}} & \frac{a_{33}}{a_{13} + a_{23} + a_{33}} \end{matrix} \quad (12)$$

In order to get the weight vector (w_i) for the criterion 1, criterion2 and criterion 3, we take the average of each row of the normalization matrix (assume the normalized matrix is a'_{ij}). Therefore

$$\frac{1}{n} \cdot \sum_{i=1}^n a'_{ij} = w_j \quad (13)$$

This process is done for the other matrices in order to obtain the weight vector for each.

That is the weight vector for each pairwise matrix which represents the relative weights of each alternative p_{ij} with respect to criterion i ($i=1, 2, 3$).

Computing global priorities

This is the step where by the relative importance of each element within the level (local priorities) is merged/multiplied with the relative importance of each element in the parent level. This gives the global priorities for each alternative.

The computation for each alternative j is done in order to obtain the overall composite weight vector, which is given by $P_j = \sum_{i=1}^n P_{ij} \cdot w_i$, where $j=1, 2, \dots, n$. This equation is equivalent to the following mathematical expression which gives the summary for computing priorities for the alternative A and B (shown in Figure 23), which is:

$$\begin{bmatrix} c1 & c2 & c3 \end{bmatrix} \begin{bmatrix} a_{c1} & b_{c1} \\ a_{c2} & b_{c2} \\ a_{c3} & b_{c3} \end{bmatrix} = P_A \ P_B \quad (14)$$

Finally, the alternative weight vector with the highest value is selected.

Because human being judgements are associated with inconsistency therefore Saaty (1980) suggested consistency checking after the weight vector have been computed.

Drawbacks of the AHP algorithm

Cheng et al. (1999) identified the shortcomings of AHP as follows:

- (i) It is used in nearly crisp (exact) decision applications;
- (ii) Deals with unbalanced scale of judgements (1 up to 9);

- (iii) Does not take into account any uncertainty associated when mapping human judgement to a number scale;
- (iv) The ranking of AHP is imprecise or inexact; and
- (v) The subjective assessment of decision makers, and change of scale have great influence on the AHP outcome; and

Furthermore, Wang et al. (2008) found that the increase in the number of characteristics geometrically increases the number of pairwise comparisons by $O(n^2/2)$ which can lead to inconsistency or failure of the algorithm. Also, AHP can't solve non linear models (Cheng, Yang, & Hwang, 1999).



APPENDIX B

EVALUATION RESULTS OF E-LEARNING SYSTEMS

The following Table 19, taken from Graf & List (2005), is the data that was used as input for AHP in the first cycle of SSM.

C a t e g o r i e s	Communication tools		Learning objects		Management of user data		Usability		Adaptation		Technical Aspects		Adminis- tration		Course manage- ment	
	Subcategories															
Λ		#			Q	Q	*		*		*	Q	+	*		*
D	+	*	Q		+	Q	*	*	*	Q	+	*	+	*	Q	+
DT	#	Q		+	Q	Q	Q		Q	Q	+		Q	Q	+	+
I	+	*		Q	Q	Q	*	*		Q	+	*		Q	+	+
L	+	*			Q	Q	*	+				*		Q	+	Q
M	*	*	Q	+	Q	+	*	*	*	#	+	*	*	#	+	+
O	#	*	Q	+	Q		*	Q		Q	+	+	+	+	+	+
S	#	*	Q		Q	Q	*	Q	*	#		*	Q	Q	+	+
SP		*			Q	Q	*	+	Q	Q		*	+	Q	Q	+

Table 19: QWS datasets

Legends used in the table: E implies essential, * extremely valuable, # very valuable, + valuable, | marginal and Q not valuable.

A implies ATutor, similarly D represents Dokeos, DT dotLRN, I ILIAS, L LON-CAPA, M Moodle, O OpenUSS, S Sakai, and SP Spaghettilearning.

Replication experiment

The testing of AHP using the data of Graf & List’s QWS calculation presents the first cycle of SSM

The experiment explained

In the following section, a stepwise explanation will be given of the replication experiment in which the QWS and AHP method were compared.

A hierarchical structure was formed using the QWS datasets (see the Table 19). Six qualitative levels namely: essential, extremely valuable, very valuable, valuable, marginally and not valuable were assigned values from a scale (See Table 20). Using these assigned values, a total value for each e-learning system was computed for each of the 8 categories defined in Table 19.

For example for the adaptation category, w_1 = value of adaptability + the value of personalization + the value of extensibility + the value of adaptivity for the particular e-learning package. Thus the value calculated for ATutor is 12 ($w_1 = 2+4+4+2$), while for Dokeos it is 11 ($w_2 =2+1+5+3$), dotLRN is 12 ($w_3 =3+3+5+1$), etc. AHP was applied to these datasets. To create the pairwise comparison matrix for adaptation (A) we start by entering the totals (w_i) for each e-learning system for the adaptation criterion into each cell of column 1 of the matrix. The value for A_{11} thus is $w_1 =12$, A_{21} is $w_2 =11$, A_{31} is $w_3=12$, etc.

Symbol	E	*	#	+		Q
Interpretation	Essential	Extremely valuable	Very valuable	Valuable	Marginally	Not valuable
Value	6	5	4	3	2	1

Table 20: Symbols with their given weights

To obtain a value of 1 for A_{11} , the denominator of the whole column is taken as 12 (or w_1 , which is the value for adaptation for ATutor) and the numerator is taken as the corresponding value of the row, for example A_{21} is $11/12$ or w_2/w_1 , $A_{31} = 12/12$ or w_3/w_1 , etc. The rest of the table was calculated by taking as numerator the corresponding value of the column with the denominator as the value of the row. This principle for computing a pairwise comparison matrix is adapted from Saaty (1980). The pairwise comparison matrix for adaptation category is shown in Table 21. The pairwise comparisons

for the remaining seven categories were calculated in the same way. The normalization matrix was obtained by dividing each element of matrix A by the summation of the column of matrix A (Ahmad and Laplante, 2006).

	ATutor	Dokeos	dotLRN	ILIAS	LON-CAPA	Moodle	Open-USS	Sakai	Spaghetti learning
ATutor (w_1)	$w_1/w_1 = 1$	1.0909	1.0000	0.9231	0.9231	0.8571	0.9231	1.5000	1.0909
Dokeos (w_2)	$w_2/w_1 = 0.9167$	1	0.9167	0.8462	0.8462	0.7857	0.8462	1.3750	1.0000
dotLRN (w_3)	$w_3/w_1 = 1.0000$	1.0909	1	0.9231	0.9231	0.8571	0.9231	1.5000	1.0909
ILIAS (w_4)	$w_4/w_1 = 1.0833$	1.1818	1.0833	1	1.0000	0.9286	1.0000	1.6250	1.1818
LON-CAPA (w_5)	$w_5/w_1 = 1.0833$	1.1818	1.0833	1.0000	1	0.9286	1.0000	1.6250	1.1818
Moodle (w_6)	$w_6/w_1 = 1.1667$	1.2727	1.1667	1.0769	1.0769	1	1.0769	1.7500	1.2727
OpenUSS (w_7)	$w_7/w_1 = 1.0833$	1.1818	1.0833	1.0000	1.0000	0.9286	1	1.6250	1.1818
Sakai (w_8)	$w_8/w_1 = 0.6667$	0.7273	0.6667	0.6154	0.6154	0.5714	0.6154	1	0.7273
Spaghetti-learning (w_9)	$w_9/w_1 = 0.9167$	1.0000	0.9167	0.8462	0.8462	0.7857	0.8462	1.3750	1

Table 21: Pairwise comparison matrix for adaptation category



Table 22 shows the normalized matrix for the adaptation category and Table 23 shows the priority vector. The priority vector was derived by computing the row average of the normalized matrix. The priority vector is also called an eigenvector or weight vector. The eigenvector determines the relative ranking of alternatives for each sub criterion.

The total weight vector was calculated by finding the sum of each row of Table 22, which shows the relative importance of each e-learning system against the eight categories. Each system was ranked using the weight vector as depicted in Table 24.

	ATutor	Dokeos	dotLRN	ILIAS	LON-CAPA	Moodle	Open-USS	Sakai	Spaghetti-learning
ATutor	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121
Dokeos	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028
DotLRN	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121	0.1121
ILIAS	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215
LON-CAPA	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215
Moodle	0.1308	0.1308	0.1308	0.1308	0.1308	0.1308	0.1308	0.1308	0.1308
OpenUSS	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215	0.1215
Sakai	0.0748	0.0748	0.0748	0.0748	0.0748	0.0748	0.0748	0.0748	0.0748
Spaghettilearning	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028	0.1028

Table 22: Normalized matrix for adaptation category

e-learning system	Categories							
	Learning Objects	Course management	Administration	Usability	Adaptation	Communication tools	Management of user data	Technical aspects
ATutor	0.1176	0.1231	0.0877	0.1170	0.1121	0.1012	0.1348	0.0879
Dokeos	0.1397	0.1231	0.1228	0.1383	0.1028	0.1190	0.0899	0.0879
DotLRN	0.0662	0.1077	0.1228	0.0851	0.1121	0.0774	0.0787	0.1538
ILIAS	0.1176	0.1385	0.1930	0.0851	0.1215	0.1071	0.1124	0.1429
LON-CAPA	0.1029	0.1538	0.1228	0.0957	0.1215	0.1131	0.0899	0.0879
Moodle	0.1618	0.0923	0.1053	0.1489	0.1308	0.1369	0.1461	0.1429
OpenUSS	0.0809	0.1077	0.0526	0.1170	0.1215	0.1250	0.0899	0.0989
Sakai	0.1250	0.0769	0.1053	0.0957	0.0748	0.1131	0.1124	0.1099
Spaghetti-learning	0.0882	0.0769	0.0877	0.1170	0.1028	0.1071	0.1461	0.0879
CI	0	0	0	0	0	0	0	0

Table 23: Priority vector for each of the categories of the evaluated e-learning system

The consistency of all pairwise comparison matrices were checked using the procedures outlined in Appendix G and Appendix J.

e-learning System	Total AHP	Priority using	Ranking by AHP	Ranking by QWS (Graf & List, 2005)
ATutor	0.8815		5	4
Dokeos	0.9235		3	3
DotLRN	0.8038		8	5
ILIAS	1.0181		2	2
LON-CAPA	0.8877		4	4
Moodle	1.0649		1	1
OpenUSS	0.7935		9	4
Sakai	0.8130		7	5
Spaghettilearning	0.8138		6	4

Table 24: Overall evaluation results

Table 24 shows the overall ranking of the e-learning systems using AHP. This ranking differs slightly from the results obtained with QWS. When considering the priority vector and total priority as depicted both in Table 23 and Table 24, Moodle is leading with ILIAS the second and Dokeos the third positions in the overall evaluation. This ranking is similar to an evaluation of Graf and List according to the specified criteria. Saaty (1980) indicated that evaluation judgements are consistent if the CR is less than 0.1 or the CI equals 0. From Table 23, it can be deduced that the evaluation of the e-learning systems with respect to all categories was consistent since the consistency indices were equal to 0.

APPENDIX C

ETHICAL CONSIDERATION

Procedure for ethical clearance

Ethical approval was sought from the University of the Western Cape Science Faculty Research Ethics Committee before commencement of the research activities in year 2006.

Procurement of consent and voluntary participation

The participants were invited to participate in the research by means of a consent form, which explained the objective of the research. It gave the assurance to the participants that all information provided will be treated as confidential. The participants signed the consent form indicating that they are willing to participate. In the consent form, each participant was informed that s/he has a right to withdraw at any time, thus participation was voluntarily without any kind of coercion or deception (Davidson, 2002; Hersh & Tucker, 2005).

Participant confidentiality agreement

In order to ensure privacy and confidentiality, the collected data were kept in a secure place and destroyed at the end of research study. Each questionnaire was numbered and did not identify any participant. The results from the study were published in conference and journal without revealing particulars of the participants.

Humanitarian Considerations: Risk and Benefits

The data collection methods that were used in this study did not risk or interfere with the mental or physical integrity of the participants. The participants were informed about the objective of the research; they had sufficient and appropriate information in order to make informed decisions (Hersh & Tucker, 2005).

Consent Form

The main objective of this study is to develop a framework for the selection and evaluation of software products in terms of software quality when the evaluators' judgements are uncertain. The development of such framework will be useful for universities (e.g. Open University of Tanzania (OUT) in developing countries when evaluating software according to specific software quality characteristics, sub-characteristics and attributes of interest. Universities in developing countries have a challenge of evaluating quality Free and Open Source e-learning system for their implementation to enhance learning. There are many Open Source e-learning systems which are available on the Internet and can be adopted by universities without having to purchase the system, but how does a university decide which quality Open Source e-learning system to adopt? The research aim at working toward this end.

I, _____, understand that my participation in this e-learning project is solely for the collection of data to improve the quality of software in general and I agree to participate. I understand that all information that I will provide will be kept confidential, and that my identity will not be revealed in any publication resulting from the research (unless I choose to give permission). Moreover, all recorded interviews and its transcripts plus data from questionnaires will be destroyed after they have been analysed. I am also free to withdraw from the project at any time.

For further information, please do not hesitate to contact:

Camilius Sanga

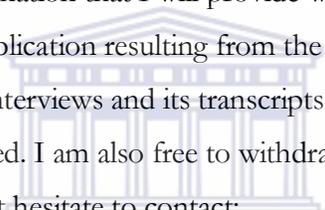
Department of Computer Science

University of the Western Cape

Private Bag X17

Bellville 7535

Email: 2657406@uwc.ac.za



UNIVERSITY of the
WESTERN CAPE

APPENDIX D

PROPOSED SOFTWARE QUALITY ATTRIBUTES

LEVEL 1	LEVEL 2	LEVEL 3	
criteria	sub criteria	attributes	
usability	learnability	time to configure	
		time to expertise	
		time to use	
	understandability	assistance / training	user documentation
			help system
			demonstration coverage
			effort to operate
	operability	tailorability	administrability
			confidence
	attractiveness	error correction and prevention	user control
			satisfaction
			general user support
			informative feedback to user
compatibility with user conventions and expectations			
consistency of screen presentation			
usability (conformance)			compliance
maintainability	complexity	complexity of the provided interface	
	stability	occurrence of error	
	analyzability	tracing error	customizability
			extensibility
			portability
	changeability	observability	controllability
			accessibility
			look & feel
	testability	scalability	
	trackability		
flexibility			

	upgradeability	easy to upgrade
deployability	portability	software system independence machine independence
	installability	ease of installation
	adaptability	suitability for personalization adaptivity
	configurability	technical documentation
	distributability	distributed system

Table 25: The hierarchical structure of the quality attributes for the proposed evaluation



APPENDIX E

EXPLANATION OF: SOFTWARE QUALITY CHARACTERISTICS, SUB-CHARACTERISTICS AND ATTRIBUTES

2. Usability refers to the feature of the software that provides the easiness of how to use it. Thus, the software must also be understandable, configurable, learnable and configurable under specific condition.

The following are the sub characteristics of usability with their respectively attributes (Seffah, Kececi, & Donyae, 2001)

- a. Learnability: it refers to the time required for a user to learn to use, configure and administer software.
 - i. Time to configure: refer to time for a user to learn how to configure the software.
 - ii. Time to expertise: refer to time required for a user to administer any task on the software
 - iii. Time to use: refer to time required for a user to learn how software can be used.
- b. Understandability refers to the capability of software to provide all necessary documentation for assisting user in using the software in the simplest manner.
 - i. Assistance / training: refers to the extent, depth and effectiveness of the guidance provided through different help systems
 - ii. User documentation: this refers to the completeness, readability, clarity, usefulness and understandability of the user manual, installation and administration manuals.
 - iii. Help system: refers to the completeness of the help files provided to the user to discover and understand its services/functions
 - iv. Demonstration coverage: this refers to the thoroughness of the provided demos (i.e. snapshots), tutorials, sample codes, online support (i.e. discussion groups, error tracking systems). These provide demos must be compared to all that are available to the interface of software.
- c. Operability: this refers to the ability of the software to indicate the level of effort needed to operate and administer.
 - i. Effort to operate: it refers to effort required to operate the software. E.g., task operated manually for software to execute. This determines the user satisfaction.
 - ii. Tailorability: refers to effort require to customize the software by configuring its packages/parameters
 - iii. Administrability: refer to effort required for administration tasks.
- d. Attractiveness: this covers the extent at which the software product attract user through layout, color and graphics designs.
 - i. Confidence: this refers to whether user are at ease when using the software (i.e. simplicity of the interface)
 - ii. Error correction and prevention: the capability of the software interface to monitors different operation from user and identify error, in case of its occurrence, as well as its ease to rectify error by user and prevention repetition of error

- iii. User control: it deals with whether the user are in control (i.e. manage) when performing different operation on the software
 - iv. Satisfaction: this is concerned with the ability of software to satisfy specific user in particular environment (i.e. user feelings when using the software)
 - v. General user support: ease at which the interface of the software is supportive to fulfill the intended aim/goal
 - vi. Informative feedback to user: This is concerned with the reaction/response of the software when user is using to perform certain tasks.
 - vii. Compatibility with user conventions and expectations: this deal with providing a proper meaning which matches the graphics of the interface. Also refers to ability of software to complete each task executed by user successful.
 - viii. Consistency of screen presentation: this refers to the uniformity of the user interface in conveying the intended meaning.
 - e. Usability compliance (conformance): refers to capability of software to comply/adhere to standards, guidelines, conventions related to usability set by /certified by external or internal organization or standardization body
 - i. Standard: this refers adherence of the software to the usability standard set by ISO 9126
3. Maintainability is the ability of the software to be corrected, adapted and improved depending on the changing user requirements or functional specifications (i.e. all these terms means modification). The following are the sub characteristics of maintainability with their respectively attributes (Bertoa & Vallecillo, 2002; Khosravi & Gueheneuc, 2004),
- a. Complexity: this refers to the complexity of packaging and using the system
 - i. Complexity of the provided interface: this refers to ability of user to operate the software without ambiguity. It reflects the number of the provided interfaces of the software.
 - b. Stability: the ability of the software to withstand unexpected effects from the modification process
 - i. Occurrence of error: refers to ability of the software to recover from error. (i.e. self handling of error without crashing/hang)
 - c. Analyzability: this refers to the ability of the software to be able to indicate the errors, deficiencies (weakness or causes of errors/failures) or artifacts which need to be modified
 - i. tracing error: the ease at which the errors can be pinpointed when they do occur in the production environment
 - d. Changeability: refers to ability of easiness and effort required to enable modification of the software.
 - i. Customizability: the ability of software to be able to be customized according to the evolving user needs.
 - ii. Extensibility: refers to ability of the software to be able to adapt new features / functions as the improvement is done. This is usually done to extend its functionality. Furthermore, the aspects of the compatibility of the new version with the previous versions are included in extensibility.
 - iii. Portability: the easy at which the modified software can be transferred to certain environment

- e. Testability: this refers to the attributes that facilitate the modification and testing of the functionality of the software.
 - i. Observability: this indicates if the start up self-test have been provided to check the performance / functionality of the software during modification
 - ii. Controllability: this indicates if the software was extensive tested before release using test suite of its packages.
 - iii. Accessibility: this refers to the kind of environment under which the testing of the software was done. This is a good indicator of the type of the environment which the software will operate without problem. Are start up self-test and test suite accessible?
 - f. Trackability
 - i. Look and feel: tracing/ monitoring /reporting the software modification
 - g. Flexibility: this refers to ease at which the software can be modified to adapt it to other environment or to different application which was not meant for.
 - i. Scalability: the ease of software to support the incremental growth of data volumes from user as well as modify to expand its capabilities. Also regardless of the data growth the software must be efficiency (i.e. processing capacity)
 - h. Upgradeability
 - i. Easy to upgrade: refers to easiness of the software to be upgraded to a new version of the software.
4. Deployability refers to ability of software to be packaged for a usage in a certain environment. The following are the sub characteristics of deployability with their respectively attributes.
- a. Portability is the ability of the software to be transferred from one environment to another.
 - i. software system independence: this refer to the extent at which the software is independent of operating system and other environment constraints (type of databases management system, programming languages)
 - ii. machine independence: the ease at which the software can run in any machines with minimum consumption of resources (software, hardware, network)
 - b. Installability: the ability to be installed to suite a particular environment
 - i. ease of installation: this refers to the effort required for the software packages to be installed in a production environment
 - c. Adaptability: the ability of software to be installed in the different environment without requiring any additional actions from those specified in the user manual.
 - i. Suitability for personalization: this express the capability of the software to be able to provide a specific user functionalities according to experience.
 - ii. Adaptivity: this refers to the capability of the software to accommodate the needs/requirements of different user.
 - d. Configurability: the capability of the software packages to be setup for running in production environments
 - i. Technical documentation: the ease at which the technicality of the software is explained in the manual
 - e. Distributability: the ability of the software to run at different locations concurrent.
 - i. distributed system: the ease at which the system can be deployed as distributed system

APPENDIX F

QUESTIONNAIRES AND OPEN-ENDED QUESTIONS

Questions for interview and summary of response

Usability

(a) What is the organisation of menus? Are they clearly shown and consistent?

Response: *The organisation of menus are logical and descriptive, also they are consistent and shown clearly. Moodle outweigh ATutor in terms of the organisation of menus.*

(b) What is the page layout? Is it structured?

Response: *Both systems (ATutor and Moodle) have page layout which structured and easy to follow.*

(c) When using the software, are users given feedback of what is happening? Response: *For both systems, not always feedback is given to the user.*

(d) Does the software provide instruction to the user about how to continue with task in each page?

Response: *Both systems provides directive how to proceed with the execution of the task.*

(e) Does the software require the user to have prior-knowledge of the software to use the software appropriately?

Response: *Both systems require some prio-basic knowledge to use them appropriately*

(f) Does the icon and images represented on each page reflect intended meaning?

Response: *Both systems have icons and images to reflect the meaning.*

(g) Does the software indicate / show errors when it occurs? does it also give some indication of the problem and some suggestion how the problem should be addressed?

Response: *Errors are shown when they do occur but does not give hint how to solve / rectify / prevent the error.*

Deployability and maintainability

(a) Did you test if the chosen software is friendly usable / accessible to different users?

Response: *Partial testing was done to check both systems if they are user friendly. This was done by IT experts, lecturers and students.*

(b) Does the software run on different operating system, and browser?

Response: *Both systems can run in heterogeneous environment*

(c) A selection of code was selected and reviewed depending on (i) structure – how well is the code structured? (ii) commenting – how clear are the comments in the codes? (iii) readability – how readable are the codes? And iv) error handling

Response: Moodle outweigh ATutor in terms of the structure of codes, commenting of codes, readability of codes and its ability to handle errors.

- (d) Do you have any documentation for the codes, software installation, user manual and troubleshooting?

Response: Documentation is available for both systems online. Low bandwidth affect accessibility / usability of these documents.

- (e) Are the programs commented in such a way that it is easy to trace in case of errors?

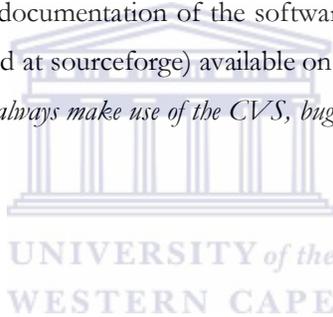
Response: Even though both systems are commented but the technical IT personnel does not make use of them in removing errors.

- (f) Do you have technical personnel who can maintain the software?

Response: There are few technical IT users to maintain the software

- (g) Do you make use of available documentation of the software (such as control version system (cvs), bugs tracing and other deposited at sourceforge) available on Internet? If yes, how?

Response: The IT personnel do not always make use of the CVS, bugs tracking systems or other documents deposited at sourceforge.



QUESTIONS FOR FOCUS GROUP – INTERVIEW

In the next part, you have been given three characteristics and their sub characteristics and you will be required to grade them according to their levels of importance.

- A. Please consider the usability, maintainability and deployability of the software in general and then indicate the level of importance of it

		Extremely important	Very important	Important	Of little important	Not important
1.	Usability	A	B	C	D	E
2.	Deployability	A	B	C	D	E
3.	Maintainability	A	B	C	D	E

- B. Please consider the usability of software and then indicate how important you feel the following sub-characteristics are:

		Extremely important	Very important	Important	Of little important	Not important
1.	Operability	A	B	C	D	E
2.	Learnability	A	B	C	D	E
3.	Attractiveness	A	B	C	D	E
4.	Understandability	A	B	C	D	E
5.	Complexity	A	B	C	D	E
6.	Usability compliance	A	B	C	D	E

C. How important do you feel are these sub-characteristics of deployability?

		Extremely important	Very important	Important	Of little important	Not important
7.	Portability	A	B	C	D	E
8.	Installability	A	B	C	D	E
9.	Adaptability	A	B	C	D	E
10.	Configurability	A	B	C	D	E
11.	Distributability	A	B	C	D	E

D. Please consider the maintainability of software in general and then indicate the level of importance of the following sub characteristics of maintainability:

		Extremely important	Very important	Important	Of little important	Not important
12.	Stability	A	B	C	D	E
13.	Analyzability	A	B	C	D	E
14.	Changeability	A	B	C	D	E
15.	Testability	A	B	C	D	E
16.	Trackability	A	B	C	D	E
17.	Upgradeability	A	B	C	D	E
18.	Flexibility	A	B	C	D	E

Questionnaire for Experts

Dear respondent,

This questionnaire is for a research study being undertaken at the Open University of Tanzania (OUT) as part of a PhD research study at University of the Western Cape, South Africa.

The objective of the study is to develop a framework for quantifying software quality attributes.

Your participation is entirely voluntary and your responses will be kept confidential. Thank you very much for completing this questionnaire.

Camilius Sanga

e-mail: csanga@uwc.ac.za

SECTION A: PERSONAL PARTICULARS

Please enter answers on the pink sheet provided, except when stated

1. Your profile, gender

Male	A	Female	B
------	---	--------	---

2. Highest academic qualification

Advanced Level Certificate	A
Diploma	B
B.A or M.A	C
B.Sc or M.Sc	D
PhD	E

3. How long have you been using a computer?

Less than 3 months	A
3 months to 6 months	B
More than 6 months but less than 1 year	C
1 year or more	D

4. Please indicate which software package you are evaluating.

ATutor	A
Moodle	B
KEWLNextGen	C

5. How long have you worked with this software package?

Less than 3 months	A
3 months to 6 months	B
More than 6 months but less than 1 year	C
1 year or more	D

The sorts of task you carry out with the software.

I use the software:

		A	B
6.	as a communication tool e.g. forum, chat, e-mail	YES	NO
7.	for learning activities e.g. tests, assignment, quiz, examination and reading course material	YES	NO
8.	for technical administration e.g. installation, maintenance and security	YES	NO
9.	for managing my course e.g. uploading course material etc.	YES	NO

SECTION B: EVALUATION OF THE QUALITY OF THE E-LEARNING SYSTEM

TICKED IN QUESTION 3

In this questionnaire the terms usability, maintainability and deployability is defined as follows:

Usability:

<p>“The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”.</p> <p style="text-align: right;">ISO 9241-11</p>

In the next part, you are given a number of usability attributes and you are required to give your opinion by grading each sub characteristic.

NO		Strongly Agree	Agree	Disagree	Strongly Disagree	I do not have enough information to answer
	Usability characteristic					
	Operability sub-characteristic					
10.	This software responds fast enough to inputs (effort to operate)	A	B	C	D	E
11.	It is relatively easy to move from one part of a task to another (tailorability)	A	B	C	D	E
12.	I feel in command of this software when I am using it (administrability)	A	B	C	D	E
	Attractiveness sub characteristic					
13.	I enjoy my sessions with this software (confidence)	A	B	C	D	E
14.	The software allows me to explore different menus tabs without fearing errors (error correction and prevention)	A	B	C	D	E
15.	This software seems not disrupt the way I normally like to arrange my work (user control)	A	B	C	D	E
16.	Working with this software is satisfying (satisfaction)	A	B	C	D	E
17.	There is enough information on the screen when it's needed (general user support)	A	B	C	D	E
18.	Software gives adequate messages showing the status of operation or process in progress (informative feedback to user)	A	B	C	D	E
19.	The software has always done what I was expecting (compatibility with user conventions and expectations)	A	B	C	D	E
20.	The software has a very attractive presentation (visual clarity on screen / consistency of screen presentation)	A	B	C	D	E
	Usability compliance sub characteristic					
21.	The interface of the software adhere well to standards, conventions, style guides and regulations (standardization)	A	B	C	D	E
	Complexity sub characteristic					
22.	The user interface is very descriptive in such a way that it does not require too much reading of user manual before you can use the software (complexity of provided interface)	A	B	C	D	E
	Understandability sub characteristic					
23.	The organization of the menus or information lists seems logical (training / assistance)	A	B	C	D	E
24.	The software documentation is informative (user documentation)	A	B	C	D	E
25.	The way that system information is presented is clear and	A	B	C	D	E

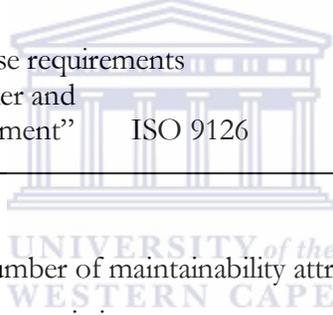
	understandable (demonstration coverage)					
26.	The instructions and prompts are helpful (help system / support)	A	B	C	D	E
	Learnability sub characteristic					
27.	Learning how to use new functions is not difficult (time to configure)	A	B	C	D	E
28.	After mastering how to use the software, I don't often have to go back to look at the guides in documentation (time to expertise)	A	B	C	D	E
29.	It does not take long to learn the software commands / instructions (time to use)	A	B	C	D	E

SECTION C

In this questionnaire the term maintainability is defined as follows:

Maintainability:

“The ease at which the software product can be modified for the purpose of
 (a) Correcting errors or defects
 (b) Meeting new or evolving end use requirements
 (c) Making future maintenance easier and
 (d) Coping with a changed environment” ISO 9126



In this part, you have been given a number of maintainability attributes and you are required to give your opinion by grading each sub characteristic.

		Strongly Agree	Agree	Disagree	Strongly Disagree	I do not have enough information to answer
	Maintainability characteristic					
	Stability sub characteristic					
30.	Unexpected errors do not occur often e.g. the system does not hang / crash	A	B	C	D	E
	Analyzability sub characteristic					
31.	The software has the capability to diagnose the cause of errors or failures (tracing errors)	A	B	C	D	E
	Changeability sub characteristic					
32.	The software is flexible and can be easily customized (customizability)	A	B	C	D	E
33.	New functionality can be added or modified to the software easily to satisfy changing user requirements without difficulty	A	B	C	D	E

	(extensibility)					
34.	It is easy for the software to be transferred to another environment (portability)	A	B	C	D	E
	Testability sub characteristic					
35.	The behavior of the software to test itself to the environment that it can work smoothly before installation (observability) - start-Up-Self-Test	A	B	C	D	E
36.	The behavior of system that it can provide test suites for checking functionality and some of its properties (controllability)- tests-suite-Provided	A	B	C	D	E
37.	The user interface of the software product is adaptable enables users with disabilities (accessibility)	A				E
	Trackability sub characteristic					
38.	The look and feel (in terms of resource usage and performance) of upgraded software resemble the initial software package	A	B	C	D	E
	Flexibility sub characteristic					
39.	The software can be adapted easily to accommodate changes such as the number of users using resource (scalability)	A	B	C	D	E
	Upgradeability sub characteristic					
40.	It is easy to upgrade the software from one version to the new version	A	B	C	D	E

In this questionnaire the term deployability is defined as follows:

Deployability:

“A set of attributes that bear on the ability of software to be transferred from one environment to another”
 ISO 9126

In this part, you have been given a number of deployability attributes and you are required to give your opinion by grading each sub characteristic.

		Strongly Agree	Agree	Disagree	Strongly Disagree	I do not have enough information to answer
	Deployability characteristic					
	Portability sub characteristic					
41.	Not much effort is needed to transfer the system from one	A	B	C	D	E

	environment (e.g. Windows) to another environment (e.g. Linux) (software system independence)					
42.	The software product is not coupled to the machine on which it operates (machine independence)	A	B	C	D	E
	Installability sub characteristic					
43.	It is easy to install the software (ease of installation)	A	B	C	D	E
	Adaptability sub characteristic					
44.	The software has features which allows personalization (suitability for personalization)	A	B	C	D	E
45.	The software has been designed to suit the needs of different types of users (adaptivity)	A	B	C	D	E
	Configurability sub characteristic					
46.	The technical documentation provide enough information how to configure the software	A	B	C	D	E
	Distributability sub characteristic					
47.	It is possible for the system to be deployed as a distributed system	A	B	C	D	E

48. Please, can you give your comments for the general quality of the software?

.....

If you have any general comments about the questions in this questionnaire³ and if you would like to receive feedback on this study after it is completed, please mail your request to the researcher at csanga@uwc.ac.za

THANK YOU FOR YOUR TIME

³ Some questions has been adapted from **Software Usability Measurement Inventory (SUMI)** questionnaire available at <http://sumi.ucc.ie/>

Questionnaire for Lecturers and Students

Dear respondent,

This questionnaire is for a research study being undertaken at the Open University of Tanzania (OUT) as part of a PhD research study at University of the Western Cape, South Africa.

The objective of the study is to develop a framework for quantifying software quality attributes.

Your participation is entirely voluntary and your responses will be kept confidential. Thank you very much for completing this questionnaire.

Camilius Sanga

e-mail: csanga@uwc.ac.za

SECTION A: PERSONAL INFORMATION

Please enter answers on the pink sheet provided, except when stated otherwise

1. Gender

Male	A	Female	B
------	---	--------	---

2. Your highest academic qualification

Advanced Level Certificate	A
Diploma	B
B.A or M.A	C
B.Sc or M.Sc	D
PhD	E

3. How long have you been using computer?

Less than 3 months	A
3 months to 6 months	B
More than 6 months but less than 1 year	C
1 year or more	D

4. The purpose is to evaluate the software you have used, Please indicate which software package you are evaluating.

Atutor	A
Moodle	B
KEWLNextGen	C

5. How long have you worked with the software chosen in Question 3.

Less than 3 months	A
3 months to 6 months	B
More than 6 months but less than 1 year	C
1 year or more	D

The sorts of task you carry out with the software (Choose one or more from these).

I have used the software:

		A	B
6.	as a communication tool e.g. forum, chat, e-mail	YES	NO
7.	for learning activities e.g. tests, assignment, quiz, examination and reading course material	YES	NO
8.	for technical administration e.g. installation, maintaining and security settings	YES	NO
9.	for managing course e.g. uploading course material	YES	NO

SECTION B: EVALUATION OF THE QUALITY OF THE E-LEARNING SYSTEM

TICKED IN QUESTION 3

In this questionnaire usability is defined as follows:

<p>“The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”.</p> <p style="text-align: right;">ISO 9241-11</p>

In the next part, you are given a number of usability attributes and you are required to give your opinion by grading each sub characteristic.

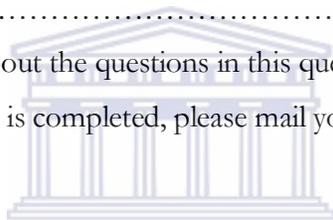
NO		Strongly Agree	Agree	Disagree	Strongly Disagree	I do not have enough information to answer
	Usability characteristic					
	Operability sub-characteristic					
10.	This software responds fast enough to inputs (effort to operate)	A	B	C	D	E
11.	It is relatively easy to move from one part of a task to another (tailorability)	A	B	C	D	E
12.	I feel in command of this software when I am using it (administrability)	A	B	C	D	E
	Attractiveness sub characteristic					
13.	I enjoy my sessions with this software (confidence)	A	B	C	D	E
14.	The software allows me to explore different menus tabs without fearing errors (error correction and prevention)	A	B	C	D	E
15.	This software seems not disrupt the way I normally like to arrange my work (user control)	A	B	C	D	E
16.	Working with this software is satisfying (satisfaction)	A	B	C	D	E
17.	There is enough information on the screen when it's needed (general user support)	A	B	C	D	E
18.	Software gives adequate messages showing the status of operation or process in progress (informative feedback to user)	A	B	C	D	E
19.	The software has always done what I was expecting (compatibility with user conventions and expectations)	A	B	C	D	E
20.	The software has a very attractive presentation (visual clarity on screen / consistency of screen presentation)	A	B	C	D	E
	Usability compliance sub characteristic					
21.	The interface of the software adhere well to standards, conventions, style guides and regulations (standardization)	A	B	C	D	E
	Complexity sub characteristic					
22.	The user interface is very descriptive in such a way that it does not need too much reading of user manual before you can use the software (complexity of provided interface)	A	B	C	D	E
	Understandability sub characteristic					
23.	The organization of the menus or information lists seems logical (assistance)	A	B	C	D	E
24.	The software documentation is informative (user documentation)	A	B	C	D	E
25.	The way that system information is presented is clear and	A	B	C	D	E

	understandable (demonstration coverage)					
26.	The instructions and prompts are helpful (help system / support)	A	B	C	D	E
	Learnability sub characteristic					
27.	Learning how to use new functions is not difficult (time to configure)	A	B	C	D	E
28.	After mastering how to use the software, I don't often have to go back to look at the guides in documentation (time to expertise)	A	B	C	D	E
29.	It does not take long to learn the software commands / instructions (time to use)	A	B	C	D	E

30. Please, can you give your comments for the general quality of the software?

.....

If you have any general comments about the questions in this questionnaire⁴ and if you would like to receive feedback on this study after it is completed, please mail your request to the researcher at csanga@uwc.ac.za



THANK YOU FOR YOUR TIME
UNIVERSITY of the WESTERN CAPE

⁴ Some questions has been adapted from **Software Usability Measurement Inventory (SUMI)** questionnaire available at <http://sumi.ucc.ie/>

APPENDIX G

STEPWISE PROCEDURES FOR CHECKING CONSISTENCY

Inconsistency is viewed as the difference or conflict in evaluation judgements from users (Teknomo, 2006).

- i. First procedure

Let A be a pairwise comparison matrix for n criteria and w be the weight vector computed for A . Let n be the number of criteria compared in the matrix.

$$A^T = Aw^T \quad (1)$$

- ii. Second procedure

Let λ_{\max} be a maximum or principal eigenvalue and n be the number of criteria. λ_{\max} , given that the number of criteria is n , is:

$$\lambda_{\max} = \frac{1n(A)_{ij}}{n-1} \sum_{i=1}^n (w_i) \quad (2)$$

- iii. Third procedure

Let CI be the consistency index. Compute CI

$$CI = \frac{\lambda_{\max} - n}{n-1} \quad (3)$$

- iv. Fourth procedure

Let CR be the consistency ratio. Let RI be the random index (see Table 48) then.

$$CR = \frac{CI}{RI} \quad (4)$$

If $CI = 0$ then A is consistent; otherwise

If $CR = \frac{CI}{RI} \leq 0.1$ then A is consistent enough

If $CR = \frac{CI}{RI} \geq 0.1$ then A is seriously inconsistent

The random index (RI_n) for any $n \times n$ square matrix is given as a constant value, as shown in Table 26

From the above procedures (3) and (4) a consistency index (CI) and consistency ratio (CR) are derived (Teknomo, 2006). Each pairwise comparison is checked against a predetermined consistency index. Even though it is impossible to come up with perfect consistency, Saaty (1980) established a threshold value of 0.10, which he determined to be an acceptable consistency ratio.

n	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49

Table 26: Random Index (adopted from Saaty, 1980)



APPENDIX H

FUZZY AHP AND PROPOSED ALGORITHM DEVELOPMENT

FUZZY SET THEORY

Fuzzy set theory has proven advantages within fuzzy, imprecise and uncertain decision situations and looks like human reasoning in its use of approximate information and uncertainty to generate decisions (Zadeh, 1965). Fuzzy set theory implements grouping of data with boundaries that are not sharply defined (i.e. fuzzy). In conventional, AHP, the pairwise comparison is established using a nine-point scale which converts the human preferences between available alternatives as equally, moderately, strongly, very strongly or extremely preferred (Cheng, Yang, & Hwang, 1999). The discrete scale of AHP has the advantage ease of use but, it can not handle (i.e is not sufficient to take into account) the uncertainty associated with the mapping of evaluator's perception to a number (Kwong & Bai, 2002). The evaluation judgements of users are normally vague and it is difficulty to represent it in terms of exact or precise numbers. It could best be given as interval judgements than fixed value judgements. Therefore, different types of fuzzy numbers (such as triangular fuzzy numbers) are used to decide the priority of one decision variable over other (Buckely and Pedrycz, 1985). Synthetic extent analysis method is used to decide the final priority weights based on the chosen fuzzy numbers (e.g. triangular fuzzy numbers) and in the thesis, it is referred as fuzzy AHP (with extent analysis). The fuzzy AHP (with extent analysis) is the fuzzy extension of AHP to handle the fuzziness of the decision during the evaluation judgements (Mikhailov, 2003; Wang & Fu, 1997). It handles both qualitative and quantitative data in the multi-attribute decision making problems.

PROPOSED ALGORITHM

The modified Fuzzy AHP (with extent analysis) algorithm introduces the fuzzy concept in each step of the conventional AHP. The purpose of introducing fuzzy set theory in AHP is to address the drawbacks of conventional AHP (Cheng, Yang, & Hwang, 1999). The classical AHP uses unbalanced scale assumes user evaluation judgement can be mapped into exact number (crisp value). But in reality human judgements span in a range of numbers (i.e. imprecise or fuzzy). This is what is called fuzzy number. The fuzzy number can be either triangular fuzzy number or trapezoidal fuzzy number etc. Thus, the modified step of the AHP becomes:

a. Fuzzy problem structuring

The problem structuring is similar to the way it is done in traditional AHP. This is the stage the unstructured problem and clearly state the goal/objectives and the intended outcomes/ results are defined. After the problem has been structured into hierarchy of characteristics, sub-characteristics, attributes and alternatives; the next step is to derive pairwise comparison in each level. AHP does this by crisp pairwise comparisons (i.e. by giving exact value for comparisons). But since human reasoning while giving evaluation judgements is “fuzzy or uncertain”. In many cases the evaluator faces the problem in comparing and might end up giving judgements in vague patterns. This makes the application of AHP not reliable to capture the vagueness and uncertainty of human (e.g. user/customer, software developer) evaluations judgement. Thus, the use of fuzzy comparison matrix is recommended instead of the pairwise comparison matrix.

b. Deriving priorities from Fuzzy Comparison matrices

The computation for local priorities must take into account the vagueness nature of human thinking (Mikhailov & Tsvetinov, 2004). To express the comparison between any two elements at the same level of hierarchy the fuzzy sets theory or fuzzy numbers are introduced to handle the fuzziness/imprecision/uncertainty of evaluation comparison judgements (Zadeh, 1965).

Fuzzy Logic

Example: The triangular fuzzy number (TFN) $\tilde{N} = a \leq b \leq c$ or can also be expressed as (a, b, c) where b, a, and c are the mean, the lower bounds and upper bounds, respectively (Buckely and Pedrycz, 1985).

The triangular fuzzy numbers (TFNs) \tilde{N} can be shown as a linear piecewise continuous membership function $\mu_{\tilde{N}}(x)$ of the type;

$$\mu_{\tilde{N}}(x) = \begin{cases} (x-a)/b-a & , a \leq x \leq b \\ (c-x)/c-b & , b \leq x \leq c \\ 0 & , o t h e r w i s e \end{cases}$$

where $-\infty < a \leq b \leq c < \infty$.

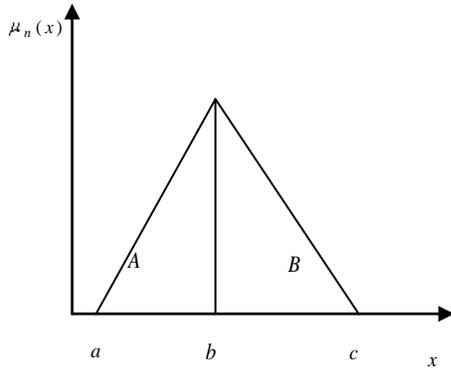


Figure 24: Fuzzy triangular membership function

The above figure shows the fuzzy set which is characterized by a membership function which assign to each object x a value ranging from 0 and 1 (Zadeh, 1965). It differs from traditional set which defines an element either belongs or does not belong to a set (i.e. 0 and 1). The fuzzy triangular membership function is define the sets in terms of the centre (b which is a point of maximum membership equivalent to 1) of the triangle and the width of the set. It is formed by two gradients $(\frac{\Delta y}{\Delta x})$, A and B , which are equivalents to $(x-a)/(b-a)$ and $(c-x)/(c-b)$ respectively. The fuzzy triangular membership function gives the foundation for defining other types of membership functions such as general triangular function, right-angled triangular function and trapezoidal function (*ibid*). For example when $a=b$ for a right-angled triangular membership function such as (1, 1, 3) (Csutora & Buckley, 2001). The user compare the characteristics/or sub-characteristics or attributes which are in the same level of the hierarchy using crisp value. Because of the inherent subjectivity of the evaluation, the algorithm must change the crisp / exact value input from user to triangular fuzzy numbers (TFNs). By doing so, the uncertainty of the evaluation judgement from the user (i.e. evaluator) is taken into account (Zadeh, 1983). The normal arithmetic operations (like addition, multiplication, division and inverse) can be done on triangular fuzzy numbers (Dubois & Prade, 1979; Zhu, Jing & Chang, 1999; Mikhailov, 2002; Mikhailov, 2003; Mikhailov & Tsvetinov, 2004; Bozdag, Kahraman & Ruan, 2003; Srdjevic, 2005).

The following tables give definitions of triangular fuzzy numbers on Saaty scale (i.e. 1 to 9). They are used to make fuzzy judgement matrix in order to solve the problem of unbalanced scale used in AHP (Kwong & Bai, 2002).

The first way to get the membership function of the scale for triangular fuzzy number is to assume that the difference between two consecutive mid-value element is two (see Table 27 and Table 28). Also, another way is to assume that the difference is 1 (see Table 27).

Fuzzy number	Membership function
$\tilde{1}$	(1,1,3)
\tilde{x}	(x-2,x,x+2) for x = 3, 5, 7
$\tilde{9}$	(7,9,9)

Table 27: Identification of membership function

Linguistic term	Fuzzy number	Membership function
Very poor	$\tilde{1}$	(1,1,3)
poor	$\tilde{3}$	(1,3,5)
ordinary	$\tilde{5}$	(3,5,7)
excellent	$\tilde{7}$	(5,7,9)
Very excellent	$\tilde{9}$	(7,9,9)

Table 28: Example of the membership function

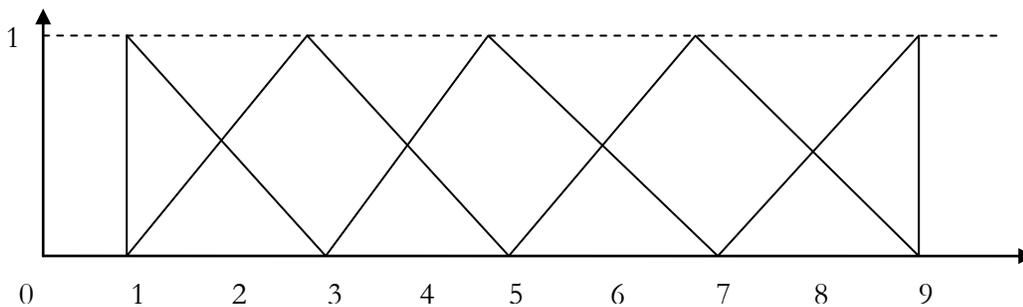


Figure 25: Membership function plot for Table 28

Linguistic term	Fuzzy number	Membership function
Equally important	$\tilde{1}$	(1,1,2)
Intermediate	$\tilde{2}$	(1,2,3)
Moderately important	$\tilde{3}$	(2,3,4)
Intermediate	$\tilde{4}$	(3,4,5)
Important	$\tilde{5}$	(4,5,6)
Intermediate	$\tilde{6}$	(5,6,7)
Very important	$\tilde{7}$	(6,7,8)
Intermediate	$\tilde{8}$	(7,8,9)
Extremely important	$\tilde{9}$	(8,9,9)

Table 29: Another example of membership function

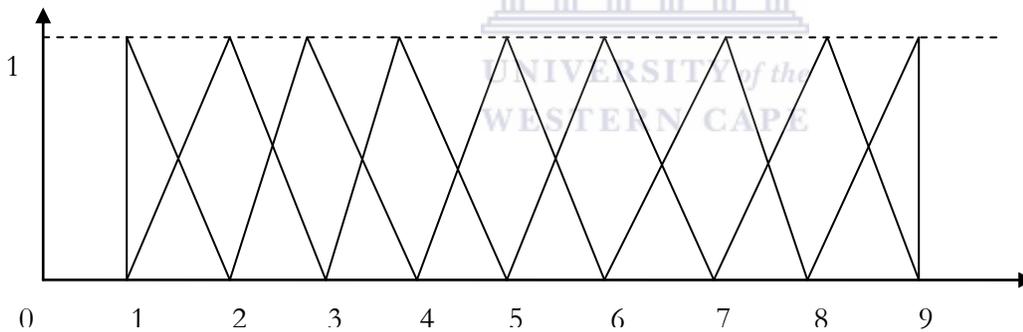


Figure 26: Membership function plot for Table 29

The following table shows the conversion of crisp pairwise comparison to fuzzy pairwise comparison

Crisp (fuzzy singleton) pairwise comparison	Fuzzy pairwise comparison	Crisp (fuzzy singleton) pairwise comparison	Fuzzy pairwise comparison
1	(1,1,3)	1	(1/3,1,1)
3	(1,3,5)	1/3	(1/5,1/3,1)
5	(3,5,7)	1/5	(1/7,1/5,1/3)
7	(5,7,9)	1/7	(1/9,1/7,1/5)
9	(7,9,9)	1/9	(1/9,1/9,1/7)

Table 30: Membership function plot for conversion of Crisp to Fuzzy pairwise comparison

One scale conversion must be adopted from the above-described scales depending on the characteristics of the data then fuzzy prioritization is done.

c. Computing Fuzzy Prioritization

Different researchers have proposed different methods for prioritizing fuzzy judgement (Wang & Chin, 2008; Zhu, Jing & Chang, 1999; Chang, 1996; Bozdog, Kahraman & Ruan, 2003). The methods includes: fuzzy AHP (with extent analysis), Fuzzy goal programming, fuzzy preference programming (Mikhailov, 2003; Wang & Fu, 1997). This study proposes a modification of extent analysis and also a new group Fuzzy AHP.

d. Computing global priorities

To rank the alternatives, the prioritization of the aggregated assessments is required. The common used methods for aggregation are the mean, max, min, median, geometric mean and mixed operators of the fuzzy numbers (Zadeh, 1965).

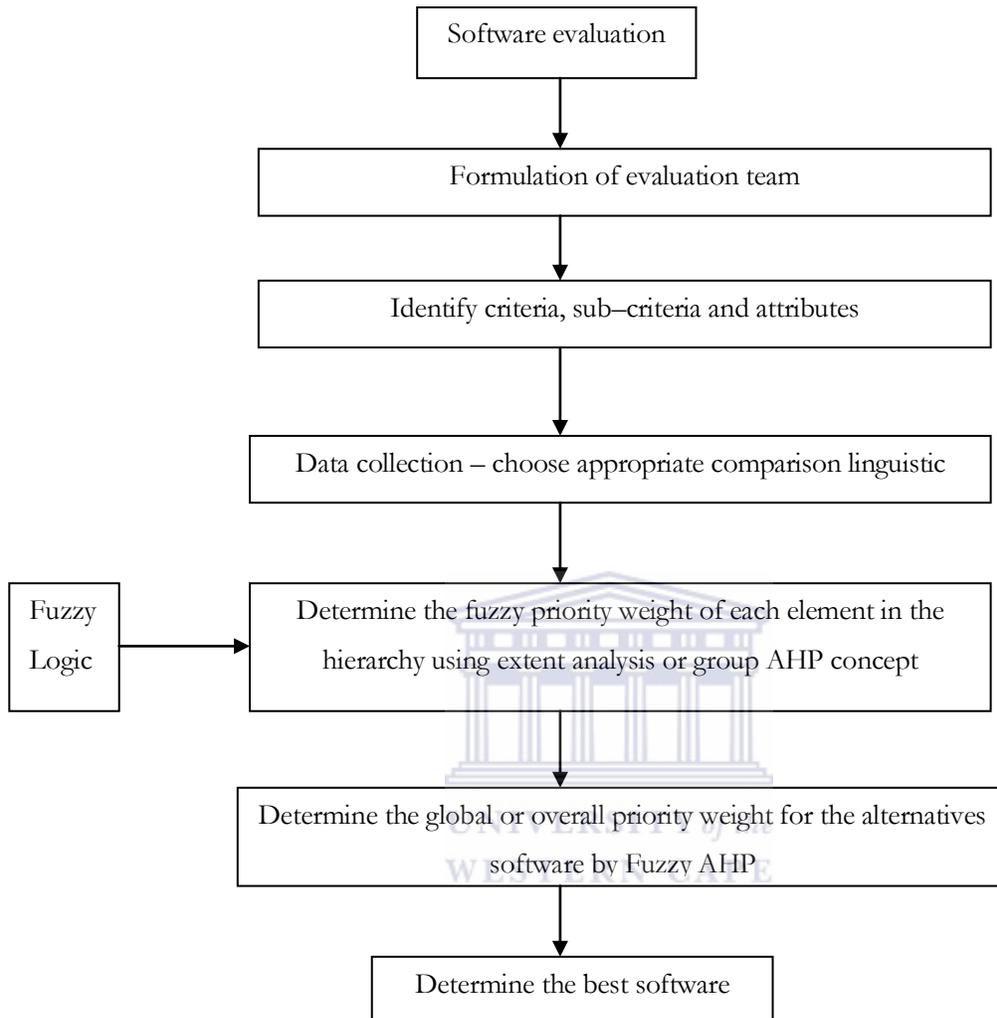


Figure 27: Fuzzy AHP

Furthermore, the following methods can be used for computing the aggregation: (i) centre of gravity method (ii) the dominance measure method (iii) the $\alpha - cut$ interval synthesis method, and (iv) the total integral value (Chen & Klein, 1997).

Proposed algorithm

Fuzzy AHP (with extent analysis)

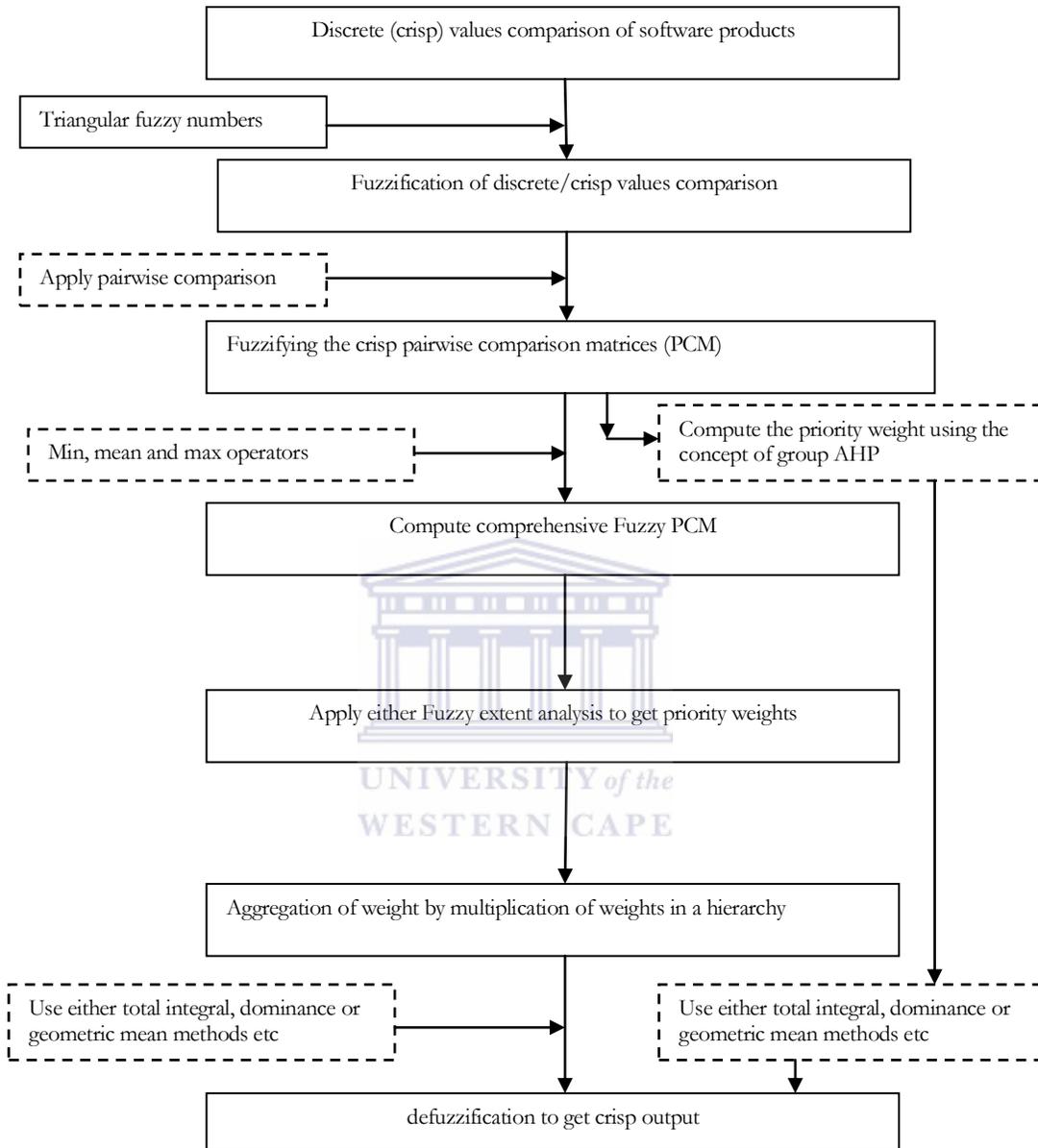


Figure 28: Fuzzy AHP (with extent analysis) and Group Fuzzy AHP

Why propose a new algorithm?

Because of the weakness of the AHP algorithm, a novel algorithm is proposed.

The new algorithm (called Group Fuzzy AHP) is introduced — it is a modified version of the Fuzzy AHP (with extent analysis) algorithm. Group Fuzzy AHP is an extension of the AHP, (the multi-criteria evaluation method under certain judgements), and it imitates the way humans reason (i.e. it can deal with uncertain judgement).

The other proposed algorithm incorporates the concept of fuzzy extent analysis in AHP.

The proposed Fuzzy AHP algorithm has three steps, which is similar to conventional AHP except that in each step, fuzzy set theory is introduced. Fuzzy extent analysis is used to obtain criteria importance and alternative performances (Zhu, Jing, & Chang, 1999). Thus, the computation of fuzzy extent analysis results in fuzzy weights. The last step is the computing of global priorities. It is done by computing the integrated fuzzy weight. This is converted into a crisp output by means of defuzzification.



APPENDIX I

GROUP FUZZY AHP

The new algorithm is described as follows: First, the evaluators give their evaluation comparison judgements of the different alternative software products in crisp values. Then the crisp values of the evaluation judgement about attribute or characteristics are fuzzified depending on the type of fuzzy number chosen. Then the fuzzy comparison matrix is split into crisp pairwise comparison matrix. This is done to get the ranking of each evaluator before the overall ranking results which is obtained by using a geometric mean technique. From the literature review we found that there is no any algorithm which resembles this algorithm, Group Fuzzy AHP (Ota, Taira, & Miyagi, 2008; Tang & Zhang, 2007). Thus, this forms part of contribution to knowledge in the area of computational intelligence (Pedrycz, 2002).

The steps of Group Fuzzy AHP are depicted in the following flowchart:

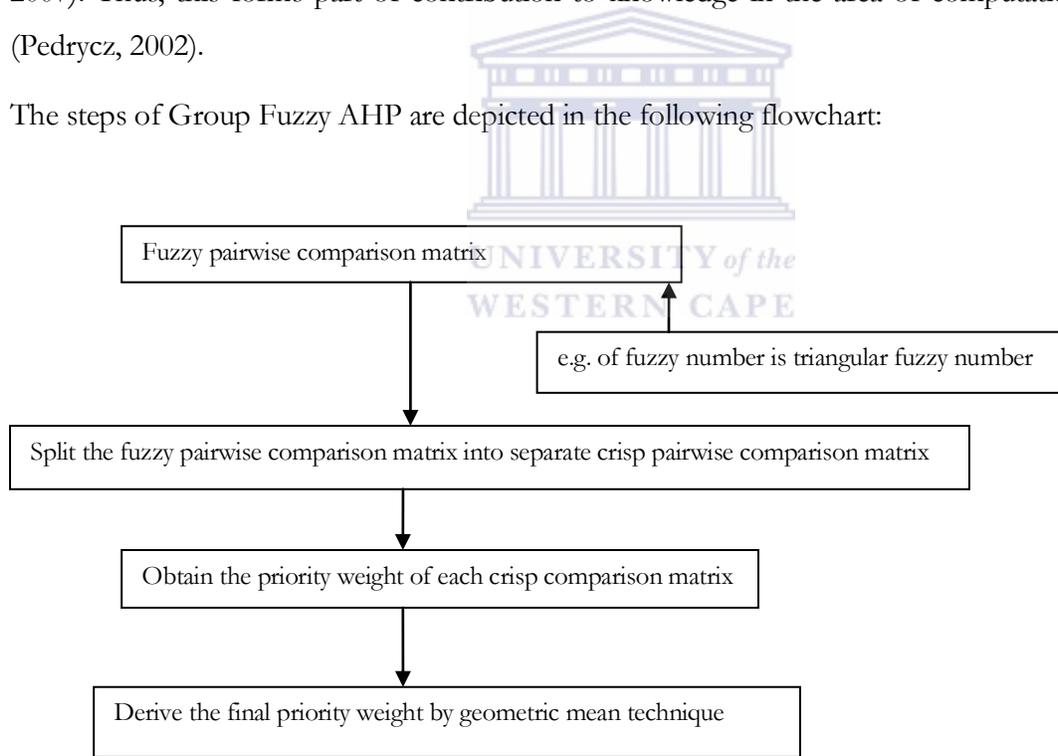


Figure 29: Group Fuzzy AHP

STEPS OF THE PROPOSED FUZZY AHP (WITH EXTENT ANALYSIS) ALGORITHM

1. Each participant evaluates the software (by either filling the questionnaire, focus group interview or answering the open-ended questions).
2. The questionnaire was filled by choosing the linguistic attributes
E.g. the question “Software gives adequate messages showing the status of operation or process in progress (informative feedback to user)” needs a user to choose one answer from (strongly agree, agree, disagree, strongly disagree and I do not have enough information to answer)
3. The qualitative chosen parameters (i.e. linguistic attributes) are converted into numerical values using triangular fuzzy numbers (a fuzzy scale is allowable from the measuring theory).

The following are examples of crisp and fuzzy scale

Symbol	A	B	C	D	E
Nominal	Strongly agree	Agree	Disagree	Strongly disagree	I don't have enough information to answer
Ordinal scale	5	4	3	2	1
Ratio scale	10	8	6	4	1
Saaty scale	9	7	5	3	1

Table 31: Crisp scale

Symbol	A	B	C	D	E
Nominal	Strongly agree	Agree	Disagree	Strongly disagree	I don't have enough information to answer
Ordinal scale	$\tilde{5}$	$\tilde{4}$	$\tilde{3}$	$\tilde{2}$	$\tilde{1}$
Ratio scale	$\tilde{10}$	$\tilde{8}$	$\tilde{6}$	$\tilde{4}$	$\tilde{2}$
Saaty scale	$\tilde{9}$	$\tilde{7}$	$\tilde{5}$	$\tilde{3}$	$\tilde{1}$

Table 32: Fuzzy scale

There are many conversion of crisp scale to fuzzy scale. The following is an example of a scale for each value of a fuzzy scale:

Fuzzy number	$\tilde{1}$	$\tilde{2}$	$\tilde{3}$	$\tilde{4}$	$\tilde{5}$	$\tilde{6}$	$\tilde{7}$	$\tilde{8}$	$\tilde{9}$
Membership function	(1,1,2)	(1,2,3)	(2,3,4)	(3,4,5)	(4,5,6)	(5,6,7)	(6,7,8)	(7,8,9)	(8,9,9)

Table 33: Membership function for conversion of crisp to fuzzy scale

The Figure 30 below was formed from the membership function shown in Table 33.

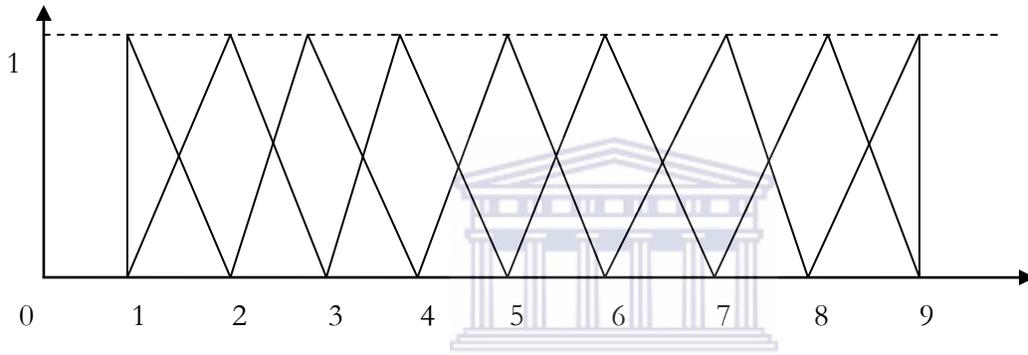


Figure 30: Membership function plot for Table 33

a. For this study, a conversion of linguistic attributes to numerical values was done for all lower level of the hierarchy (i.e. sub criteria and attributes – see Table 25 25). Table 33 was used in the conversion.

4. Compute the mean value of all opinions from evaluators (refer to step 3) (in this case it resulted to triangular fuzzy number for Moodle and ATutor). The use of weight average operator helps to get the collective opinion of all participants (any other operator can be used).

$$v_j = \frac{\sum_{r=1}^{r=q} \tilde{p}_r}{q} \quad (1)$$

where $\tilde{p}_r = (\tilde{v}_{l_r}, \tilde{v}_{m_r}, \tilde{v}_{u_r})$

$$\tilde{v}_{l_r} = \frac{\sum_{r=1}^{r=q} l_r}{q}$$

$$\tilde{v}_{m_r} = \frac{\sum_{r=1}^{r=q} m_r}{q}$$

$$\tilde{v}_{u_r} = \frac{\sum_{r=1}^{r=q} u_r}{q}$$

q is the total number of either characteristics or sub-characteristics or attributes and $r=1,2,3,\dots,q$

5. Then compute the pairwise comparisons matrices in order to obtain the preference weights of each element. This step gives the Fuzzy pairwise comparison matrix.

- a. i.e. Finding weight using the pairwise comparison for each matrix which is in form of triangular fuzzy number (l, m, u)

The pairwise comparison judgement matrix, which gives the preference of one criteria (c_i) over the other (c_j), is given by $c_{ij} = \frac{c_i}{c_j}$ for $i,j=1,2,3,\dots,n$. Similarly, it was done for sub-criteria and attributes.

Thus for four criteria the following will be the pairwise comparison matrices:

- b. There will be 12 pairwise comparison matrices since each element will have three triangular fuzzy scale.

	c_1	c_2	c_3	c_4		c_1	c_2	c_3	c_4	
$w_1 =$	c_1	1	x	y	$w_2 =$	c_1	1	x	y	
	c_2	$1/x$	1	z		c_2	$1/x$	1	z	
	c_3	$1/y$	$1/z$	1		c_3	$1/y$	$1/z$	1	
	c_4	.	.	.	1	c_4	.	.	.	
	c_1	c_2	c_3	c_4		c_1	c_2	c_3	c_4	
$w_3 =$	c_1	1	x	y	.	$w_4 =$	c_1	1	x	y
	c_2	$1/x$	1	z	.		c_2	$1/x$	1	z
	c_3	$1/y$	$1/z$	1	.		c_3	$1/y$	$1/z$	1
	c_4	.	.	.	1		c_4	.	.	.

	c_1	c_2	c_3	c_4		c_1	c_2	c_3	c_4	
$w_5 =$	c_1	1	x	y	.	$w_6 =$	c_1	1	x	y
	c_2	$1/x$	1	z	.		c_2	$1/x$	1	z
	c_3	$1/y$	$1/z$	1	.		c_3	$1/y$	$1/z$	1
	c_4	.	.	.	1		c_4	.	.	.

$$\begin{array}{cccc}
 & c_1 & c_2 & c_3 & \cdot \\
 c_1 & 1 & x & y & \cdot \\
 w_7 = c_2 & 1/x & 1 & z & \cdot \\
 c_3 & 1/y & 1/z & 1 & \cdot \\
 c_4 & \cdot & \cdot & \cdot & 1
 \end{array}
 \quad
 \begin{array}{cccc}
 & c_1 & c_2 & c_3 & c_4 \\
 c_1 & 1 & x & y & \cdot \\
 w_8 = c_2 & 1/x & 1 & z & \cdot \\
 c_3 & 1/y & 1/z & 1 & \cdot \\
 c_4 & \cdot & \cdot & \cdot & 1
 \end{array}$$

$$\begin{array}{cccc}
 & c_1 & c_2 & c_3 & c_4 \\
 c_1 & 1 & x & y & \cdot \\
 w_9 = c_2 & 1/x & 1 & z & \cdot \\
 c_3 & 1/y & 1/z & 1 & \cdot \\
 c_4 & \cdot & \cdot & \cdot & 1
 \end{array}
 \quad
 \begin{array}{cccc}
 & c_1 & c_2 & c_3 & c_4 \\
 c_1 & 1 & x & y & \cdot \\
 w_{10} = c_2 & 1/x & 1 & z & \cdot \\
 c_3 & 1/y & 1/z & 1 & \cdot \\
 c_4 & \cdot & \cdot & \cdot & 1
 \end{array}$$

$$\begin{array}{cccc}
 & c_1 & c_2 & c_3 & c_4 \\
 c_1 & 1 & x & y & \cdot \\
 w_{11} = c_2 & 1/x & 1 & z & \cdot \\
 c_3 & 1/y & 1/z & 1 & \cdot \\
 c_4 & \cdot & \cdot & \cdot & 1
 \end{array}
 \quad
 \begin{array}{cccc}
 & c_1 & c_2 & c_3 & c_4 \\
 c_1 & 1 & x & y & \cdot \\
 w_{12} = c_2 & 1/x & 1 & z & \cdot \\
 c_3 & 1/y & 1/z & 1 & \cdot \\
 c_4 & \cdot & \cdot & \cdot & 1
 \end{array}$$



(2-4)

6. Compute the comprehensive pairwise comparison

The aim of this computation is to get a single large matrix (\tilde{W}) which combines all weight using fuzzy quantifiers (Yager, 1988).

Comprehensive pairwise comparison is obtained by first grouping each element of pairwise comparison matrices, which must be in form of a fuzzy triangular number.

$$\tilde{w}_{ijk} = (w_{l_{ijk}}, w_{m_{ijk}}, w_{u_{ijk}}) \tag{5}$$

Where l , m and u represent the following:

$$\begin{aligned}
 w_{l_{ijk}} &= \text{minimum}(l_{ijk}) \\
 w_{m_{ijk}} &= \text{average}(m_{ijk}) \\
 w_{u_{ijk}} &= \text{maximum}(u_{ijk})
 \end{aligned}$$

Then the single formed large matrix is reduced to a small matrix by finding minimum of its lower values, mean of its middle values and maximum of its upper bound values.

This is possible using the concept of fuzzy triangular number. The triangular fuzzy number (TFN) $\tilde{N} = l \leq m \leq u$ or can also be expressed as (l, m, u) where m , l , and u are the mean, the lower bounds and upper bounds.

The outcome of this step computation is a fuzzy comprehensive pairwise comparison matrix

$$\tilde{W} = [w_1, w_2, w_3, w_4, \dots, w_n] \quad (6)$$

$$\tilde{A} = \tilde{w}_{ijk} = \begin{bmatrix} (1,1,1)(a_{12l}a_{12m}a_{12u}) & \cdots & (a_{1nl}a_{1nm}a_{1nu}) \\ \vdots & \ddots & \vdots \\ (a_{m1l}a_{m1m}a_{m1u})(a_{12l}a_{11m}a_{12u}) & \cdots & (1,1,1) \end{bmatrix} \quad (7)$$

Equation 7 can be expressed in form of equation 8

$$\tilde{w}_{ijk} = \begin{bmatrix} (1,1,1) & \cdot & \cdot & \cdot \\ \cdot & (1,1,1) & \cdot & \cdot \\ \cdot & \cdot & (1,1,1) & \cdot \\ \cdot & \cdot & \cdot & (1,1,1) \end{bmatrix} \quad (8)$$

Still fuzzy and crisp values are present in the above matrix (8). This also has been observed by Yeh and Deng (2004) who argue that the multi-criteria problem usually contain both crisp and fuzzy values.

7. Apply the fuzzy extent analysis to the comprehensive pairwise comparison matrix. The reason for applying fuzzy extent analysis is because the comprehensive pairwise matrix resembles the fuzzy judgement matrix. It is from only fuzzy judgement matrix where fuzzy extent analysis can be applied (Zhu et al., 1999).

Find the priorities using fuzzy extent analysis $\tilde{w}_{ijk} = (l_{ijk}, m_{ijk}, u_{ijk})$

$\tilde{A} = \tilde{w}_{ijk}$ is the fuzzy comprehensive pairwise comparison matrix which the formula for fuzzy extent analysis was applied (9).

$$\tilde{x}_i = \tilde{w}_j = \frac{\sum_{j=1}^k \tilde{a}_i}{\sum_{i=1}^k \sum_{j=1}^k \tilde{a}_{ij}} \quad (9)$$

where $i=1,2,3,\dots,k$; $p=1, 2, 3,\dots, q$ and $k=q$

$$\tilde{x}_i = \begin{bmatrix} (a_{11l}a_{11m}a_{11u}) \\ \vdots \\ (a_{i1l}a_{ijm}a_{iju}) \end{bmatrix} \quad (10)$$

where j = number of sub-criteria and number of criteria in the other upper level

$\tilde{x}_i = [(w_{1l}w_{1m}w_{1u})(w_{2l}w_{2m}w_{2u}) \cdots (w_{nl}w_{nm}w_{nu})]$ where n = number of sub-criteria or sub-criteria.

Thus, a fuzzy weighted performance matrix (p) can be obtained by multiplying the weight vector with the decision matrix.

$$p_i = \tilde{x}_i \times \tilde{w}_{ij} \text{ where } \tilde{x}_i = s_i \text{ (see equation 16)}$$

7.1 Computing Fuzzy Extent

The basic procedures for fuzzy extent are adopted from Zhu et al. (1999).

Let $x = \{x_1, x_2, \dots, x_n\}$ an object set

$G = \{g_1, g_2, \dots, g_n\}$ be a goal defined in the hierarchical structure of the evaluation problem. The same can be done for second and third level. Thus, G can change depending on the level of the hierarchy (that is either at the level of criteria or sub-criteria or attributes).

M extent analysis on each object is taken

$$M_{gi}^1, M_{gi}^2, M_{gi}^3, \dots, M_{gi}^n \text{ where } i=1,2,3,\dots,n$$

Where \tilde{M}_{gi}^j ($j=1, 2, 3, \dots, m$) are triangular fuzzy numbers

7.1.1 First procedure: The value of fuzzy synthetic extent with respect to the i th object is defined as

$$S_i = \sum_{j=1}^m \tilde{M}_{gi}^j \otimes \left[\sum_{i=1}^n \sum_{j=1}^m \tilde{M}_{gi}^j \right]^{-1} \quad (11)$$

To obtain $\sum_{j=1}^m \tilde{M}_{gi}^j$, perform the fuzzy addition operation of m extent analysis values for a particular matrix such that:

$$\sum_{j=1}^m \tilde{M}_{gi}^j = \left[\sum_{j=1}^m l_j, \sum_{j=1}^m m_j, \sum_{j=1}^m u_j \right] \quad (12)$$

and to obtain $\left[\sum_{j=1}^n \sum_{g=1}^m \tilde{M}_{gi}^j \right]^{-1}$, perform the fuzzy addition operation of $M_{gi}^j (j = 1, 2, \dots, m)$ values

such that

$$\sum_{i=1}^n \sum_{j=1}^m \tilde{M}_{gi}^j = \left[\sum_{i=1}^n l_i, \sum_{i=1}^n m_i, \sum_{i=1}^n u_i \right] \quad (13)$$

and then compute the inverse of the vector above, such that:

$$\left[\sum_{i=1}^n \sum_{j=1}^m \tilde{M}_{gi}^j \right]^{-1} = \left[\frac{1}{\sum_{i=1}^n u_i}, \frac{1}{\sum_{i=1}^n m_i}, \frac{1}{\sum_{i=1}^n l_i} \right] \quad (14)$$

7.1.2 Second procedure: layer simple sequencing

Pair by pair comparison of each block towards the overall goal is done. This gives the sequencing weight vector for each block. The same procedure is done when finding the priority weights for second and third levels. This method has disadvantage of eliminating some useful judgement from users or evaluators. This will be advantageous if the aim is to isolate some data from a large database.

According to Bozdag et al. (2003) as $\tilde{M}_1 = (l_1, m_1, u_1)$ and $\tilde{M}_2 = (l_2, m_2, u_2)$ are two triangular fuzzy numbers, the degree of possibility of $\tilde{M}_2 = (l_2, m_2, u_2) \succeq \tilde{M}_1 = (l_1, m_1, u_1)$ defined as:

$$V(\tilde{M}_2 \succeq \tilde{M}_1) = \begin{cases} 1, & \text{if } m_2 \geq m_1 \\ 0, & \text{if } l_1 \geq u_2 \\ \frac{l_1 - u_2}{(m_2 - u_2) - (m_1 - l_1)}, & \text{otherwise} \end{cases} \quad (15)$$

This method has disadvantage of eliminating some useful judgement from user especially when

$$V(\tilde{M}_2 \succeq \tilde{M}_1) = 0$$

But this might be important feature for the case of the researchers who want to isolate some data from a chunk of data in database.

7.1.3 Third procedure:

Is to normalize the sequencing vector obtained in the above procedure. The result is non-fuzzy number (Zhu et al., 1999).

Therefore, fuzzy weight becomes

$$p_i = \sum_{j=1}^m \tilde{M}_{gi}^j \otimes \left[\sum_{i=1}^n \sum_{j=1}^m \tilde{M}_{gi}^j \right]^{-1} \otimes \tilde{w}_i \quad (16)$$

Then the value of p must be defuzzified for final ranking. Examples of methods for defuzzification are the centre of gravity method, the dominance measure method, the alpha cut (α –cut) with synthesis method and the total integral value method. In this study, we adopted the total integral value method by Liou and Wang (1992). The motivation for using this is because it can be used in computing a wide of range of defuzzification values between 0 and 1. This is similar to the fuzzy state of reasoning of the evaluators.

Thus, according to Liou and Wang method for any given matrix in form of triangular fuzzy numbers (TFNs) $A = (l_1, m_1, u_1)$, the total integral value is defined by:

$$I_T^\lambda(A) = (1/2) \left(u_1 + m_1 + (1-\lambda)l_1 \right), \lambda \in (0,1) \quad (17)$$

This expression indicates an optimism index (λ) which expresses the judgement subjectivity (i.e. it shows the views of evaluator in pessimistic, moderate or optimistic).

APPENDIX J

THE CHECKING OF CONSISTENCY: SSM CYCLE 1.

i. First procedure

The computation for A^T was done for each category (see E.q. (1) in Appendix G). In order to check consistency for the adaptation category, the pairwise comparison matrix is multiplied by its priority vector, as shown in Table 24.

The result of this computation for the adaptation category is equal to 1.0093, 0.9252, 1.0093, 1.0935, 1.0935, 1.1776, 1.0935, 0.6729, and 0.9252.

ii. Second procedure

The computation for the maximum eigenvalue was done for each category (see E.q. (2) in Appendix G) For the adaptation category,

$$\text{maximum eigenvalue} = 1/9 (1.0093/0.1121 + 0.9252/0.1028 + 1.0093/0.1121 + 1.0935/0.1215 + 1.0935/0.1215 + 1.1776/0.1308 + 1.0935/0.1215 + 0.6729/0.0748 + 0.9252/0.1028) = 9.0$$

iii. Third procedure

After that the computation for the consistency index (CI) was done for each category (see E.q. (3) in Appendix G). For example for the adaptation category, the consistency index (CI) = $(9-9) / (9-1) = 0$

iv. Fourth procedure

The level of consistency of the AHP method was checked which gave the results reported in Table 7 (see E.q. (4) in Appendix G). All CI were equal to zero, therefore we did not compute the CR .

APPENDIX K

APPLICATION OF AHP TO DATA COLLECTED FROM OUT

(This appendix presents the results of the second cycle of the SSM)

To obtain priority/ weight vector for level 1: respondents were asked to indicate the level of importance of three criteria usability, deployability and maintainability (c_1, c_2, c_3) respectively. Then the mean of all the respondents were computed to obtain the input values for AHP. We got ($c_1=9, c_2=7$ and $c_3=7$) using scale shown in Table 18. After that the equation (6) from Appendix A was applied and we obtained the following pairwise matrix:

1	1.285714286	1.285714286
0.777778	1	1
0.777778	1	1

Then we computed normalization matrix using equation (10) from Appendix A and we obtained the following matrix:

0.391304	0.391304348	0.391304348
0.304348	0.304347826	0.304347826
0.304348	0.304347826	0.304347826

After that the priority vector was computed using equation (13) from Appendix A and we obtained the following weights for usability, deployability and maintainability:

- 0.391304348
- 0.304347826
- 0.304347826

This answer is reflected in level 1 of Table 34.

Then we checked the consistency using the formulae shown in Appendix G.

Equation (1) in Appendix G gave the following result:

- 1.173913043
- 0.913043478
- 0.913043478

The equation (2) in appendix G was applied

$$\lambda_{\max} = \frac{1}{n} \sum_{i=1}^n \frac{A_{ii}}{w_i}$$

$$= 1/3 * (1.173913043 / 0.391304348 + 0.913043478 / 0.304347826 + 0.913043478 / 0.304347826) = 3$$

After this computation, we calculated the *CI* using equation (3) in Appendix G $CI = \frac{\lambda_{max} - n}{n - 1}$
 = (3-3)/2 = 0 (the number of criteria (*n*) was 3)

This indicated that the consistency index was equal to 0 (which means that the judgements were consistent).

These computations were done for all hierarchy levels and the results are shown in the following Table 34 (Priority weight obtained after using AHP).

LEVEL 1	LEVEL 2	LEVEL 3					
		ATutor	Moodle	ATutor	Moodle		
		Weight	Priority	Priority	Priority		
usability 0.3913	operability	0.0685	0.5000	0.5000	0.0114	0.0114	
			0.6250	0.3750	0.0143	0.0086	
	usability compliance	0.0685	0.2000	0.8000	0.0137	0.0548	
		complexity	0.0880	0.5833	0.4167	0.0514	0.0367
	understandability			0.5000	0.5000	0.0110	0.0110
			0.0880	0.5833	0.4167	0.0128	0.0092
				0.4167	0.5833	0.0092	0.0128
	learnability			0.5833	0.4167	0.0128	0.0092
			0.0098	0.5000	0.5000	0.0016	0.0016
				0.6923	0.3077	0.0023	0.0010
attractiveness			0.5000	0.5000	0.0016	0.0016	
		0.0685	0.5000	0.5000	0.0043	0.0043	
			0.5000	0.5000	0.0043	0.0043	
deployability 0.3043	portability		0.4167	0.5833	0.0036	0.0050	
			0.4667	0.5333	0.0040	0.0046	
			0.4667	0.5333	0.0040	0.0046	
			0.5000	0.5000	0.0043	0.0043	
			0.1429	0.8571	0.0012	0.0073	
	installability		0.5000	0.5000	0.0043	0.0043	
		0.0830	0.1667	0.8333	0.0069	0.0346	
	adaptability		0.6429	0.3571	0.0267	0.0148	
		0.0646	0.4375	0.5625	0.0282	0.0363	
			0.0461	0.5000	0.0115	0.0115	
			0.5556	0.4444	0.0128	0.0102	

maintainability 0.3043	configurability	0.0646	0.4375	0.5625	0.0282	0.0363
	distributability	0.0461	0.4375	0.5625	0.0202	0.0259
	stability	0.0609	0.4375	0.5625	0.0266	0.0342
	analyzability	0.0261	0.6364	0.3636	0.0166	0.0095
			0.4375	0.5625	0.0063	0.0082
	changeability	0.0435	0.5000	0.5000	0.0072	0.0072
			0.4667	0.5333	0.0068	0.0077
	testability	0.0435	0.5000	0.5000	0.0072	0.0072
			0.5000	0.5000	0.0072	0.0072
			0.5000	0.5000	0.0072	0.0072
	trackability	0.0261	0.5000	0.5000	0.0130	0.0130
	flexibility	0.0435	0.5000	0.5000	0.0217	0.0217
upgradeability	0.0609	0.5000	0.5000	0.0304	0.0304	
Total					0.4686	0.5314

Table 34: Priority weight



APPENDIX L

PRINCIPLES OF EVALUATING INTERPRETIVE FIELD STUDIES

The principles for evaluating interpretive field studies in information systems were adopted from Klein and Myers (1999) and Niehaves (2005). These were taken as a method to validate results. The following Table shows how the principles for conducting and evaluating the field studies were adhered to in our research.



Name of the principle	What is theory behind the principle	How the principle was applied in the research
1.The Fundamental Principle of the Hermeneutic Circle	This proposes that all human understanding is achieved by iterating between the interdependent meaning of parts under consideration and the whole that they form. This principle forms the basis for all the other principles.	This was done by positing the data collection methods to according to the hierarchical structure of the software quality characteristics, sub characteristics and attributes. This set up helped in the modeling of computation algorithm so that the results of the analysis were representation of the whole software products and not constituent individual/separate characteristics, sub characteristics and attributes.
2. The Principle of Contextualization	This principle deals with the critical reflection of the social and historical background of the research setting. This is done so that the intended respondents can understand how the current situation under examination emerged.	This was applied by contextualising the research study specific to the environment of developing countries
3. The Principle of Interaction between the Researchers and the Subjects	This principle deals with critical reflection on how data from the research were socially constructed through the interaction between the researchers and respondents.	This has been explained in the section of research methodology. At the end of the section the presentation about strategies taken to address the threat brought by the interaction between researcher and participants are explained.
4. The Principle of Abstraction and Generalization	This deals with associating the idiographic details exposed by the data interpretation through the application of the above principles (one and two) to theoretical, general concepts that describe the nature of human understanding and social action.	From the above 1 st and 2 nd principles, the results from this study can not be generalised because they were contextualized in specific environment. Evaluation involving human differs according to the type of the software system under evaluation, environment and experience/knowledge/expert/culture of users
5. The Principle of Dialogical Reasoning	This deals with identifying if there are any contradictions between the theoretical preconceptions directing the research design and actual research outcomes with subsequent cycles of adjustment.	There is no contradiction between the underlying theoretical assumptions involved in the research study which are from measurement theory and software quality. Measurement theory allows quantification of the software quality as perceived by users. In this case, the perceived depend on the software deployed in a certain environment. Systematic logical reasoning was derived in different cycles of SSM.
6. The Principle of Multiple Interpretations	This principles deal with identifying if there are differences in interpretations among the respondents. The idea is to find if there is multiple narratives of the same sequence of events under research study or similar stories of events from multiple narratives	This was taken care by the computational model which was having ability to consolidate results from different evaluators/ users/ participants.
7. The Principle of Suspicion	This finds if there are “biases” and systematic “distortions” in the story telling collected from the respondents.	This was done by comparing data from different instrument explaining similar theme

Table 35: Principles for conducting and evaluating interpretive field studies

APPENDIX M

PRIORITY WEIGHT OBTAINED AFTER USING FUZZY AHP (WITH EXTENT ANALYSIS) AND GROUP FUZZY AHP

(This section presents the outcome of the third and fourth cycle of SSM)

This section presents the results of the evaluation of two FOSS e-learning systems according to usability characteristics. For the sake of illustrating how the computation of the priority weights were done, the procedure for obtaining the priority weight for sub-characteristics of usability will be shown. First, the pairwise comparison was computed for the sub-characteristics of usability using equations 6 up to 9 (see Appendix A). This resulted in the following matrix:

<i>Usability</i>	learnability	undestandability	operability	attractiveness	usability compliance
learnability	1.000	7.000	0.333	9.000	5.000
undestandability	0.143	1.000	5.000	0.200	7.000
operability	3.000	0.200	1.000	3.000	0.143
attractiveness	0.111	5.000	0.333	1.000	0.111
usability compliance	0.200	0.143	7.000	9.000	1.000

After obtained the pairwise comparison matrix for sub-characteristics of usability, the normalization of the pairwise comparison matrix was done using equations 11 and 12 (see Appendix A). The resulting matrix was as follows:

<i>Usability</i>	learnability	undestandability	operability	attractiveness	usability compliance
learnability	0.225	0.525	0.024	0.405	0.377
undestandability	0.032	0.075	0.366	0.009	0.528
operability	0.674	0.015	0.073	0.135	0.011
attractiveness	0.025	0.375	0.024	0.045	0.008
usability compliance	0.045	0.011	0.512	0.405	0.075

Finding the priority vector is the final step of the AHP algorithm. The priority vector contains the local weights. The priority vector was computed using equation 13 (see Appendix A).

learnability	0.311
undestandability	0.202
operability	0.182
attractiveness	0.095
usability compliance	0.210

The above computation was done for each lower level element with respect to the higher-level element of the usability characteristic -as shown in the Table 25. This process gave the results for all the level of the usability characteristic in the hierarchy (see Table 36).

The priority vector of the sub-characteristics of usability were multiplied by the priority vector of their respective attributes (represented by the numbers 10-29 in Figure 13). This resulted to the priority vector containing global weight (column 5 in Table 36).

The last two columns of the priority vector (titled Moodle and ATutor) were obtained by comparing Moodle and ATutor with respect to each attribute.

The following is an explanation of how the first element of the last two columns respectively were calculated (in Table 34). As explained before, the priority vector was obtained by computing pairwise comparison matrix and then normalizing it. The pairwise comparison matrix was obtained using equation 6-9 (Appendix A). It gave the following matrix:

27	Moodle	ATutor
Moodle	1.000	2.000
ATutor	0.500	1.000

Then the above matrix was normalized and resulting vector, priority weights (using equation 13 in Appendix A) is:

Moodle	0.6666667
ATutor	0.3333333

The above steps were done for all elements in Table 25 and the final step was to compute the overall priority weight using equation 14.

$$\begin{bmatrix} 0.667 & \dots & 0.111 \\ 0.333 & \dots & 0.889 \end{bmatrix} \times \begin{bmatrix} 0.108 \\ \vdots \\ 0.210 \end{bmatrix} = \begin{bmatrix} 0.483 \\ 0.517 \end{bmatrix}$$

The Table 36 shows the priority vector for each level which was computed. In order to validate the algorithm the error was checked and it was found to be 0 and hence the evaluation judgement was consistency. That is why there was no need of testing consistency (equations 1- 4) (see Appendix G). The overall priority weight obtained for ATutor was 0.517 and that of Moodle was 0.483

Results

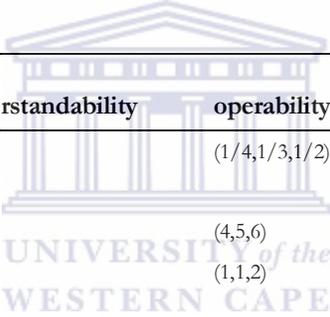
	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.311	27	0.348	0.108	0.667	0.333
		28	0.343	0.107	0.833	0.167
		29	0.309	0.096	0.857	0.143
understandability	0.202	23	0.245	0.049	0.750	0.250
		24	0.147	0.030	0.125	0.875
		21	0.199	0.040	0.250	0.750
		25	0.408	0.083	0.333	0.667
operability	0.182	10	0.655	0.119	0.667	0.333
		11	0.097	0.018	0.900	0.100
		12	0.248	0.045	0.100	0.900
attractiveness	0.095	13	0.194	0.019	0.167	0.833
		14	0.107	0.010	0.200	0.800
		15	0.187	0.018	0.667	0.333
		16	0.090	0.009	0.167	0.833
		17	0.066	0.006	0.900	0.100
		18	0.101	0.010	0.125	0.875
		19	0.165	0.016	0.333	0.667
usability compliance	0.210	20	0.090	0.009	0.833	0.167
		22	1.000	0.210	0.111	0.889
				Priority	0.483	0.517
				TOTAL	1	
				Error	0	

Table 36: Results from conventional AHP

Group Fuzzy AHP

This algorithm addresses the problem giving a crisp value when evaluating software. For example, a user might feel that an attribute for user interface is not good, nor it is bad, it might be something in between. To be able to use the data collected at OUT, it was decided to convert the discrete/ crisp comparison values into fuzzy/continuous values using some defined conversion method (those explained in Appendix H). The following pairwise comparison matrix depicts how this conversion was done. It was done for all level of hierarchy in Figure 13 but only one usability sub-characteristics is discussed below.

The fuzzy pairwise comparison matrix for the usability sub-characteristics is:



<i>Usability</i>	learnability	understandability	operability	attractiveness	usability compliance
learnability	(1,1,2)	(6,7,8)	(1/4,1/3,1/2)	(8,9,9)	(4,5,6)
understandability		(1,1,2)	(4,5,6)	(1/6,1/5,1/4)	(6,7,8)
operability			(1,1,2)	(2,3,4)	(1/8,1/7,1/6)
attractiveness				(1,1,2)	(1/9,1/9,1/8)
usability compliance					(1,1,2)

The lower elements from the diagonal of the pairwise comparison matrix is filled by computing the reciprocal (or inverse) of each corresponding element. This results to fuzzy pairwise comparison matrix.

Then the fuzzy pairwise comparison matrix is divided into three matrices consisting of lower, middle and upper bound elements. It is divided into different number of elements depending on the type of fuzzy number (e.g. triangular fuzzy number or trapezoidal fuzzy number etc).

If the above fuzzy pairwise comparison matrix is split into lower, middle and upper bound elements the following is obtained:

For the lower elements

<i>Usability</i>	learnability	understandability	operability	attractiveness	usability compliance
learnability	(1)	(6)	(1/4)	(8)	(4)
undestandability		(1)	(4)	(1/6)	(6)
operability			(1)	(2)	(1/8)
attractiveness				(1)	(1/9)
usability compliance					(1)

For the middle elements

<i>Usability</i>	learnability	understandability	operability	attractiveness	usability compliance
learnability	(1)	(7)	(1/3)	(9)	(5)
undestandability		(1)	(5)	(1/5)	(7)
operability			(1)	(3)	(1/7)
attractiveness				(1)	(1/9)
usability compliance					(1)

For the upper elements

<i>Usability</i>	learnability	understandability	operability	attractiveness	usability compliance
learnability	(2)	(8)	(1/2)	(9)	(6)
undestandability		(2)	(6)	(1/4)	(8)
operability			(2)	(4)	(1/6)
attractiveness				(2)	(1/8)
usability compliance					(2)

After that the priority weight of each pairwise comparison matrix is done like what has been shown in the previous example of the conventional AHP. This method is called group Fuzzy AHP.

The following Table 37 shows the priority weights for the lower element.

The results for the lower elements

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.280	27	0.333	0.093	0.500	0.500
		28	0.345	0.097	0.800	0.200
		29	0.321	0.090	0.833	0.167
understandability	0.188	23	0.212	0.040	0.667	0.333
		24	0.158	0.030	0.111	0.889
		21	0.183	0.034	0.200	0.800
		25	0.447	0.084	0.250	0.750
operability	0.185	10	0.603	0.111	0.500	0.500
		11	0.111	0.021	0.889	0.111
		12	0.285	0.053	0.100	0.900
attractiveness	0.113	13	0.181	0.021	0.143	0.857
		14	0.093	0.011	0.167	0.833
		15	0.168	0.019	0.500	0.500
		16	0.102	0.012	0.143	0.857
		17	0.079	0.009	0.889	0.111
		18	0.111	0.013	0.111	0.889
		19	0.167	0.019	0.250	0.750
20	0.098	0.011	0.800	0.200		
usability compliance	0.235	22	1.000	0.235	0.100	0.900
					Priority	0.398
				TOTAL	1	

Table 37: Results from Group Fuzzy AHP - for lower elements

Results for the middle elements

The following Table 38 shows the priority weights for the middle element.

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.311	27	0.348	0.108	0.667	0.333
		28	0.343	0.107	0.833	0.167
		29	0.309	0.096	0.857	0.143
understandability	0.202	23	0.245	0.049	0.750	0.250
		24	0.147	0.030	0.125	0.875
		21	0.199	0.040	0.250	0.750
		25	0.408	0.083	0.333	0.667
operability	0.182	10	0.655	0.119	0.667	0.333
		11	0.097	0.018	0.900	0.100
		12	0.248	0.045	0.100	0.900
attractiveness	0.095	13	0.194	0.019	0.167	0.833
		14	0.107	0.010	0.200	0.800
		15	0.187	0.018	0.667	0.333
		16	0.090	0.009	0.167	0.833
		17	0.066	0.006	0.900	0.100
		18	0.101	0.010	0.125	0.875
		19	0.165	0.016	0.333	0.667
		20	0.090	0.009	0.833	0.167
usability compliance	0.210	22	1.000	0.210	0.111	0.889
				Priority	0.483	0.517
				TOTAL	1	

Table 38: Results from Group Fuzzy AHP - for middle elements

Results for the upper elements

The following Table 39 shows the priority weights for the upper elements.

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.361	27	0.380	0.137	0.729	0.271
		28	0.339	0.122	0.837	0.163
		29	0.281	0.101	0.856	0.144
understandability	0.215	23	0.311	0.067	0.778	0.222
		24	0.154	0.033	0.163	0.837
		21	0.192	0.041	0.350	0.650
		25	0.343	0.074	0.500	0.500
operability	0.157	10	0.695	0.109	0.729	0.271
		11	0.090	0.014	0.883	0.117
		12	0.214	0.034	0.129	0.871
attractiveness	0.083	13	0.203	0.017	0.222	0.778
		14	0.124	0.010	0.271	0.729
		15	0.195	0.016	0.729	0.271
		16	0.085	0.007	0.222	0.778
		17	0.056	0.005	0.883	0.117
		18	0.093	0.008	0.163	0.837
		19	0.161	0.013	0.500	0.500
usability compliance	0.184	20	0.083	0.007	0.837	0.163
		22	1.000	0.184	0.144	0.856
				Priority	0.558	0.442
			TOTAL	1		

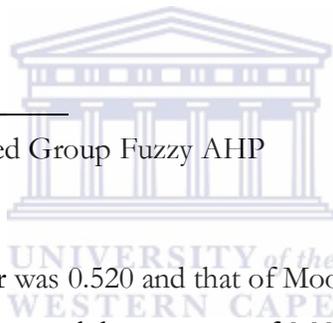
Table 39: Results from Group Fuzzy AHP - for upper elements

After obtaining the results for the priority weights of the lower, middle and upper elements then the final step is to combine three respective overall priority weights (for the lower, middle and upper element) in order to get the global weight for all. As mentioned in the literature review that there are several methods such as total integral, dominance or geometric mean methods for such purpose.

In this thesis, the geometric mean was used and the following results were obtained:

	Moodle	ATutor
1	0.398	0.602
2	0.483	0.517
3	0.558	0.442
geometric mean	0.475	0.516
average	0.480	0.520
Total	0.991	
Error	0.009	

Table 40: Results of the combined Group Fuzzy AHP



The overall priority weight of ATutor was 0.520 and that of Moodle was 0.48. However since the total was not 1 as expected it seems as if the result has an error of 0.009 (i.e. 1-0.991).

After the results for the new algorithm, Group Fuzzy AHP, which uses the concept of group AHP, was obtained; another algorithm was used to find if the results obtained are correct. In addition, the aim was to find if the fuzzy AHP can be modified to a better algorithm. This method was done as illustrated in Figure 16 and Figure 29.

The first step was to convert the discrete/ crisp comparison value into fuzzy / continuous value using some conversion (as explained in Appendix H). Then the elements in the matrix were sorted using equation 5 (see Appendix I) to get a comprehensive fuzzy pairwise comparison matrix. After that, the extent analysis was done on a comprehensive fuzzy pairwise comparison matrix to compute the fuzzy priority vector.

This is similar to the first procedure in the previous algorithm, Group Fuzzy AHP. It is from the previous example of group Fuzzy AHP were the lower, middle and upper bound values were combined (instead of splitting as in group Fuzzy AHP) to form a comprehensive fuzzy pairwise

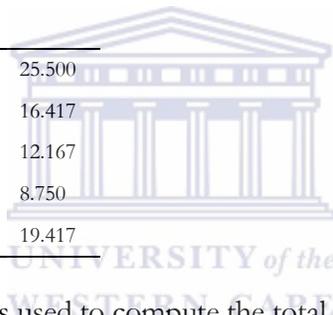
comparison matrix. This was done using equations 2-8 (see Appendix I). It resulted into the following matrix:

learnability			Understandability			operability			attractiveness			usability compliance		
1.000	1.333	2.000	6.000	7.000	8.000	0.250	0.361	0.500	8.000	8.667	9.000	4.000	5.000	6.000
0.125	0.145	0.167	1.000	1.333	2.000	4.000	5.000	6.000	0.167	0.206	0.250	6.000	7.000	8.000
2.000	3.000	4.000	0.200	0.817	2.000	1.000	1.333	2.000	2.000	3.000	4.000	0.125	0.145	0.167
0.111	0.116	0.125	4.000	5.000	6.000	0.250	0.361	0.500	1.000	1.333	2.000	0.111	0.116	0.125
0.167	0.206	0.250	0.125	0.145	0.167	6.000	7.000	8.000	8.000	8.667	9.000	1.000	1.333	2.000

Then the equations (9 – 14) (see Appendix I).were applied on the above matrix to form the following matrix:

Extent

learnability	19.250	22.361	25.500
understandability	11.292	13.684	16.417
operability	5.325	8.295	12.167
attractiveness	5.472	6.926	8.750
usability compliance	15.292	17.350	19.417



The equation 11 (see Appendix I) was used to compute the total of value of each column:

sum	56.631	68.616	82.250
-----	--------	--------	--------

Then we applied equation 14 (see Appendix I) to get the inverse of the above column vector:

0.012	0.015	0.018
-------	-------	-------

After that, the equation 10 (see Appendix I) is computed to get the priority weight. This means the above extent matrix is multiplied by the inverse vector

$$\begin{bmatrix} 19.250 & \dots & 25.500 \\ \vdots & \ddots & \vdots \\ 15.292 & \dots & 19.417 \end{bmatrix} \times \begin{bmatrix} 0.012 \\ 0.015 \\ 0.018 \end{bmatrix} = \begin{bmatrix} 0.234 & \dots & 0.450 \\ \vdots & \ddots & \vdots \\ 0.186 & \dots & 0.343 \end{bmatrix}$$

The result for the above expression is shown below:

Priority	0.234	0.326	0.450
weight	0.137	0.199	0.290
	0.065	0.121	0.215
	0.067	0.101	0.155
	0.186	0.253	0.343

This fuzzy priority vector is defuzzified by using either total integral, dominance or geometric mean methods to get the final crisp/discrete value. In this thesis, the geometric mean was used.

Geometric mean

	Local weight
learnability	0.325
understandability	0.199
operability	0.119
attractiveness	0.101
Usability compliance	0.253



Thus, this completes the computation for sub characteristics of usability. Then all the above procedures were done for other levels of the hierarchy and gave the results in Table 41.

	local weight	Question No.	local weight	global weight (overall)	Moodle	ATutor
learnability	0.325	27	0.340	0.111	0.619	0.381
		28	0.402	0.131	0.798	0.201
understandability	0.199	29	0.256	0.083	0.710	0.290
		23	0.298	0.059	0.824	0.175
		24	0.198	0.040	0.155	0.844
operability	0.119	21	0.130	0.026	0.290	0.710
		25	0.373	0.074	0.381	0.619
attractiveness	0.101	10	0.588	0.070	0.619	0.381
		11	0.080	0.010	0.867	0.131
		12	0.331	0.039	0.131	0.867
		13	0.187	0.019	0.201	0.798
		14	0.105	0.011	0.237	0.763
usability compliance	0.253	15	0.151	0.015	0.619	0.381
		16	0.107	0.011	0.201	0.798
		17	0.059	0.006	0.867	0.131
		18	0.136	0.014	0.155	0.844
		19	0.149	0.015	0.381	0.619
usability compliance	0.253	20	0.106	0.011	0.798	0.201
		22	1.000	0.253	0.139	0.860
		Priority			0.454	0.541
			Total		0.996	
			error		0.004	

Table 41: Results from Fuzzy AHP (with extent analysis)

After obtaining the above priority weights the remaining task was to merge the priority vector from different levels. The local weight (or priority vector) of first level were multiplied by the priority vector of the second level to get the global weight of each attributes. This was then multiplied to the respective local weight of each e-learning system to get the overall weight (or global priority vector).

This is the principle of equation 16 (see Appendix I) which gave the final priority vector used in ranking after defuzzification.

$$p_i = \sum_{j=1}^m \tilde{M}_{gi}^j \otimes \left[\sum_{i=1}^n \sum_{j=1}^m \tilde{M}_{gi}^j \right]^{-1} \otimes \tilde{w}_i$$

The Table 41 gives the summary of the priority weight of each element in Figure 13 using Fuzzy extent analysis.

The overall weight of ATutor was 0.541 and that of Moodle was 0.454. We got an error of 0.004

Comparison of the results

Method	Priority weight		
	Moodle	ATutor	Error
Conventional AHP	0.483	0.517	0
Group Fuzzy AHP	0.475	0.516	0.009
Fuzzy AHP (with extent analysis)	0.454	0.541	0.004

Table 42: Comparison of outcome



APPENDIX N

CASE STUDY SUMMARY

1. The purpose of the case study

The main objective of this thesis is to develop a framework for the selection and evaluation of software products in terms of software quality when the evaluators' judgements are uncertain. The development of such framework will be useful for universities (e.g. Open University of Tanzania, OUT) in developing countries when evaluating software according to specific software quality characteristics, sub-characteristics and attributes of interest. Universities in developing countries have a challenge of evaluating quality Free and Open Source e-learning system for their implementation to enhance learning. There are many Open Source e-learning systems which are available on the Internet and can be adopted by universities without having to purchase the system, but how does a university decide which quality Open Source e-learning system to adopt? The research aim at working toward this end.

2. Description of the field of research

OUT was established in 1992 under an Act of Parliament No. 17 which permitted them to offer Open and Distance Education. OUT offered the following degree courses: Bachelor of Arts (B.A), Bachelor of Arts with Education (B.A (Ed.)), Bachelor of Commerce (B.Com.), Bachelor of Commerce with Education (B.Com. (Ed.)), Bachelor of Laws (LLB), Bachelor of Science (BSc.), BSc. Information Communication Technology and Bachelor of Science with Education (BSc.) (Ed.)

OUT serves an area of 945,000 square km. Its teaching and learning system is based on print learning materials and two written assignments, but also includes an orientation, two face-to-face tutoring sessions, science laboratory sessions or teaching practical, two timed tests, and an annual and supplementary examination (if required) in each course (OUT, 2010). These activities occur in the 26 regional centers, which also provide limited library services; to reach them, often across long distances, students need enough money to pay for transportation, accommodation, food, and any medical care. There are also 69 local centers or study centres, some in each region, where students meet subject peer groups and their tutors (Bakari et al., 2008). The printed course materials are delivered to students

mainly by postal services, but also through public carriers and via the OUT staff during their visits to the regional centers. Plans are underway to use the regional centers as the main venues for storing and distributing study materials (Mmari, 1997). OUT buys study materials from other distance learning universities in Nairobi, Zimbabwe, South Africa, Abuja, and India and has published 84 of its own courses, written by faculty from Tanzanian universities and other post secondary institutions. OUT has four faculties and two Institutes with a total of 220 full time staff and about 240 part-time staff (Bakari et. al., 2008).

OUT has already enacted ICT policy and strategic plan known as Synopsis OUT Rolling Strategic Plan (Bakari et. al., 2008). Under plan, OUT envisages to be a leading world-class university in delivery affordable quality education through Open and Distance Learning. Its institutional ICT Policy and Guidelines provide the framework on how ICT can enhance provision of education system. Furthermore, OUT ICT policy and Guidelines plus that of National Information and Communications Technologies Policy act as catalyst to the application of ICT in education. In summary, the ICT policy provides the regulations and guidelines how OUT can implement ICT to fulfill the university functions (i.e. teaching, learning, research, outreach and consultancy).

Problems facing OUT include: lack of enough fund to buy equipments/personal computer and build ICT infrastructure, inadequately lecturers / tutors, lack of enough staff for planning of the curriculum, lack of enough learning materials, inadequate buildings, lack of enough bandwidth for Internet connectivity, interruption of electricity and lack of enough computer laboratory.

The challenge facing OUT is to implement a quality ICT based solution to improve learning. The solution must be catered with the limited available resources and taking into consideration the level of ICT attained by the country.

It is from the mentioned challenge that in 2006, OUT's system analyst and administrators were tasked to choose one free and open source e-learning system and install. The aim was to implement an e-learning system to support learning. It was by trial and error that two FOSS e-learning systems were considered. The choice was made based on which is the most widely used FOSS e-learning systems by universities in Tanzania and in the world in general. In addition, cost was another factor considered. The software which was free and which won't force the university to purchase additional accessories

to install to the available resources were given higher preference. In 2006, ATutor was implemented but few lecturers and students used. In 2007, ATutor was phased out and Moodle was implemented. Even though both systems are from FOSS but their quality attributes differ. Since the selection and evaluation of the FOSS e-learning were done in ad-hoc fashion the need for developing the framework which will help different stakeholders (i.e. university top leaders, staffs, ICT experts and students) at OUT was important.

3. Research plan and methodology

3.1. Study location

The main campus of OUT in Dar es Salaam, Tanzania was used as study area.

3.2. Methodology

Data were collected between November 2007 to February 2008. The methodologies chosen for data collection for this study were: the survey, case study and ethnography. Soft Systems Methodology (SSM) was used to manage the analysis of the collected data.

3.3. Methods

Qualitative methods and quantitative methods were used in both data collection and data analysis. The qualitative method entailed a case study where participant observation was used and focus group interviews were conducted. The quantitative methods consisted of the use of questionnaires to collect data and the different algorithms such as AHP, Fuzzy AHP and Group Fuzzy AHP to analyse the data.

3.4. Research ethics

Before visiting the study area in November 2007, the ethical clearance (as shown in appendix C) was sought from the University of the Western Cape research Ethics Committee.

3.5. Data analysis

3.5.1. Survey

Some of the answers given by different respondents during the interview session are as given below:

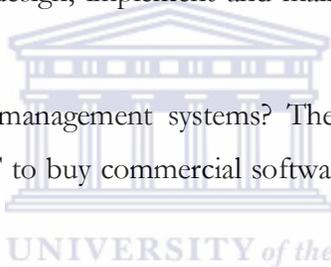
- (a) top leaders group comprised of Deputy Vice Chancellor (Academic), Director of Research, Postgraduate and Publications, and Director of Institute of Education Technology, Director of Library, Coordinator of Open Distance Electronic Learning (ODEL) project.

Do you have university ICT policy? They said that OUT has enacted ICT policy.

Do you have infrastructure for implementation of e-learning? They said that at main campus there is a Local Area Network connecting computers of staff and students. Also, there is Wide Area Network (WAN) connecting main campus in Dar es Salaam with some other regional centres located in different geographical areas in Tanzania. In addition, there is Internet connectivity in staff offices and student computer laboratories.

Do you have technical personal to design, implement and maintain e-learning? They said that there were few IT experts.

Can you buy commercial learning management systems? They said that with limited budget the university has, it is difficult for OUT to buy commercial software (in particular learning management system or e-learning system).



Can you adopt Free and Open Source learning management systems? They said adoption from Free and Open Source learning management systems is the best option for OUT. Already they said there are several initiatives toward this aim.

What are the benefits of using FOSS in provision of online learning (i.e. e-learning)? They mentioned the following advantages: (i) being free thus cost effective for the university with limited budget. (ii) provides a total ownership. (iii) It reduces piracy.

- (b) Staff comprised two tutors

Can you change the way learning and teaching is conducted? The staff agreed that it is possible to adopt ICT in teaching provide there is change of mindset between the stakeholders.

What are the benefits of using FOSS in provision of online learning (i.e. e-learning)? Staff mentioned the following benefits: (i) lowers the cost to purchase commercial software (ii) good for providing free source codes and software for testing, learning and teaching.

(c) Students comprised of 20 BSc. ICT (second year).

Are student motivated and satisfied with the way teaching is offered by OUT? The students mentioned the following as means of teaching and learning: broadcasting, telecasting, Information and Communication Technologies (ICT), correspondence, enhanced face to face, seminars, contact programmes or the combination of any two or more of such means.

Students were motivated with the means of teaching but they were not satisfied. The blended way of learning (i.e. combination of face to face and ICT based learning) was required by students.

What are the benefits of using FOSS in provision of online learning (i.e. e-learning)? Benefits which were mentioned by students are: (i) enhances communication with their lecturers and between students (ii) encourage independent learning (iii) provides the opportunity for students to contribute to the development of the FOSS

(d) IT experts comprised of e-learning coordinator and two system administrators.

Can the team evaluate and implement a quality FOSS e-learning system? The IT experts said they have an initiative to select and evaluate FOSS e-learning system for their implementation.

What are the benefits of using FOSS in provision of online learning (i.e. e-learning)? IT experts prefer FOSS because (i) there is total control of the software (not locked). This means the software is given free with its source codes (ii) it forces them to learn thus ignites creativity

Apart from the above questions, the questionnaires were given to the sampled students, lecturers and IT experts. Furthermore, the focus group was conducted to probe some of the issues pertaining to software quality as indicated in Appendix F.

Before the respondents filled the questionnaire and attended the focus group, they were introduced to the terms concerning the software quality. The definitions from the ISO 9126 software quality were given to the participants (ISO, 2001). This helped the participants to be familiar with the exercise of filling the questionnaires and answering questions from the focus group. The task of clarifying each software quality characteristics, sub-characteristics and attributes used in the study was similar to the following as illustrated by Padayachee et. al (2010).

Sub-characteristics of usability characteristic:

- Understandability - 'Does the user comprehend how to use the system easily?'
- Learnability - 'Can the user learn to use the system easily?'
- Operability - 'Can the user use the system without much effort?'
- Attractiveness - 'Does the interface look good?'

Sub-characteristics of maintainability characteristic:

- Analyzability - 'Can faults be easily diagnosed?'
- Changeability - 'Can the software be easily modified?'
- Stability - 'Can the software continue functioning if changes are made?'
- Testability - 'Can the software be tested easily?'

Sub-characteristics of deployability characteristic:

- Adaptability - 'Can the software be moved to other environments?'
- Installability - 'Can the software be installed easily?'
- Conformance - 'Does the software comply with portability standards?'
- Replaceability - 'Can the software easily replace other software?'

Usability characteristic

Understandability sub-characteristic

Participants' perceptions of the understandability sub-characteristic of e-learning system with respect to:

- Consistent use of terms throughout the system
- Consistency of layout
- Functions of buttons
- System terminology related to pedagogic tasks
- Consistent positioning of error messages on the screen
- Clear prompts for input

- Informing users of systems progress
- Match between task in interface and task as understood by user and supported by system
- Ease of understanding information (and documentation) provided by the system

Learnability sub-characteristic

Participants' perceptions of learnability sub-characteristic of FOSS e-learning system with respect to:

- The level of difficulty when learning to operate the system
- The level of difficulty when exploring new features by trial and error
- The level of difficulty when remembering names and use of commands
- The ease and straightforwardness of performing tasks
- The usefulness of help messages on the screen
- The clarity of supplemental reference materials (online help, onscreen messages and other documentation)



Operability sub-characteristic

Participants' perceptions of operability sub-characteristic of FOSS e-learning system:

- The level of ease with which tasks can be performed such as uploading resources; organizing students into groups etc.
- The clarity with which information is organized
- The logic and clarity in which sequence of screens presented
- The level of ease when correcting errors
- Effectiveness of help systems in use
- Ease of finding required information

Attractiveness sub-characteristic

Participants' perceptions with regards to:

- Pleasantness of systems interface
- Attractiveness of systems interface

3.5.1. Ethnography

Participant observation was done to fill the gaps in data collected with other data collection methods. The missed data were some issues corresponding to how IT experts, students and lecturers involved in different functions executed by the FOSS e-learning system. The functions which lecturers used are (i) uploading the course materials (ii) communicating with the students through e-mail, wiki etc. The functions which students used are (i) assessing course materials (ii) communication with lecturers – through discussion forum, wiki, e-mail etc.

The observation was that most of students did not regularly access the course materials and communicate with their lecturers by using FOSS e-learning system. This was contributed by the challenges which will be mentioned in subsequent sections.

4. Outline of the issues and findings of the case study

From the qualitative analysis, it was found that OUT was in preliminary stage in using either ATutor or Moodle for e-learning. Also, there was having some couple of projects related to ICT. These were hosting a mirror of MIT Open Courseware and initiative to host a branch for Africa Virtual University (AVU). AVU is a consortium dealing with provision of teaching and learning online in African Universities. Furthermore, there was an initiative for creating course contents through ODEL. The Director of Institute of Education Technology was the custodian of all ICT activities.

Challenges for using MIT Open Courseware were: (i) low bandwidth (ii) most students being distance learners. This means only few face to face teaching and learning were done in the main campus. Most of the students access Internet from Internet café. Few of students access from home, own office and OUT computer laboratories. Thus, inadequate access to Internet forced the students and staff to have many questions: for example, (i) how to access Open Course ware? (ii) how to use it? (iii) how to adopt course materials from Open Courseware? (iv) how to sensitize the use of e-content so that the IT phobia can be reduced. (v) how to conduct training for basic computer skills to reduce the computer illiteracy among the students.

The lead respondent asked the researcher: how to get the students and staff use the MIT Open Courseware and the FOSS e-learning system regardless of the challenges in adoption?

The challenges in adoption of FOSS e-learning system identified by respondents were:

- a. Low bandwidth – which was the major constraint for the implementation
- b. Lack of enough infrastructure available to the students to access Internet
- c. Lack of enough technical staff to provide help and guidance for lecturers to create material to upload on a system
- d. Some students lack the expertise or knowledge on how to access the Internet and e-learning system
- e. Course materials being not uploaded to the system
- f. Lack of framework to select and evaluate a system according to software quality characteristics, sub-characteristics and attributes preferred by stakeholders (i.e. students, lecturers)

Besides the above challenges, the respondents jot down the benefits of using ICT to improve the learning experience of students. The benefits mentioned identified by respondents were:

- a. Widen access to learning and teaching
- b. Improve the communication between the students / learners and lecturers / instructors
- c. Overcome the shortage of instructors / lecturers
- d. Encourage student independent student learning
- e. Reduce the need of instructors to travel between regional centres and learning centres
- f. Provide the means to reach out to students who are distributed over the country and outside the country
- g. Help the students to progress faster through their degrees
- h. Facilitate learning in spaces and times where learning does not normally happen
- i. Integrate OUP with a global learning and teaching community and help improve its practices

5. Recommendations

To use the Soft Systems Methodology (SSM) to analyse the data collected from qualitative and quantitative research methods. This is presented in the thesis. The outcome of the thesis lead into the development of the framework for the selection and evaluation of FOSS e-learning system according to software quality characteristics, sub – characteristics and software attributes.



INDEX

A

AHP · xiv
AVOIR · xiv

C

CVS · xiv

D

Distance education · xvi
DVD · xv

E

EFL · xiv
E-learning · xvi
E-learning system · xvi
ESL · xiv

F

FOSS · xiv

G

GDP · xiv, xv
GQM · xiv

I

ICT · xiv
ICT4D · xv
INCAMI · xv
ISO · xv
IT · xiv, xv, xvi

K

KEWL.NextGen · xv

M

MCDA · xv

O

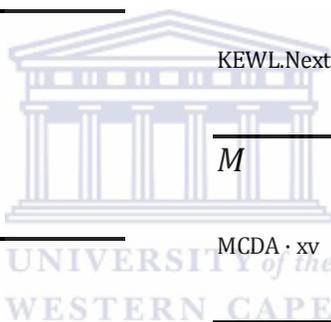
OTSO · xv
OUT · xv

P

Pedagogy · xvi
PORE · xv

Q

QWS · xv



S

SSEF · xv
STACE · xv
SUA · xv

UWC · xv

W

WebQEM · xv

U

UDSM · xv

