REMOTE EVALUATION OF AN INTERFACE USABILITY USING ASYNCHRONOUS AUTO-LOGGING METHODOLOGY

TARUS NICHOLAS KIPROTICH

MASTER OF SCIENCE

(Computer Systems)

JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY

2017

Remote evaluation of an interface usability using asynchronous auto-logging methodology

Tarus Nicholas Kiprotich

A thesis submitted in partial fulfilment for the degree of Master of Science in Computer Systems in the Jomo Kenyatta University of Agriculture and Technology.

2017

DECLARATION

This thesis is my original work and has not been presented for a degree in any other University.

Signature..... Date.....

Tarus Nicholas Kiprotich

This thesis has been submitted for examination with my approval as University Supervisors.

Signature.....Date.....

Dr. Stephen Kimani, PhD

JKUAT, Kenya

Signature...  .....Date..31st May 2017.....

Dr. Wilson Cheruiyot, PhD

JKUAT, Kenya

DEDICATION

I wish to dedicate this work to my late wife Tabitha Jepchirchir Lessan, my sons Bramwel Kiptoo Tarus and Brian Kirwa Tarus to whom I owe a lot.

ACKNOWLEDGEMENT

It is difficult to overstate my gratitude to my advisors, Dr Stephen Kimani and Dr Wilson Cheruiyot. Their enthusiasm, inspiration, and efforts in explaining ideas clearly and simply helped make this project interesting. Moreover, they provided continuous encouragement, sound advice and many good ideas. I would have been lost without them.

I am also very thankful to Henry Ohanga for helping me during the various stages of the system development design and Ms. Wamukekhe Everlyne for providing input at different phases of this research. I am extremely grateful to Mr Raphael Murey for his valuable input in correcting this thesis and for teaching me how to write it.

I wish to thank Elias Kipyego and Ainabkoi CDF committee for helping me with the various infrastructure-related issues and lastly, many thanks to Visions Interactive and InterPay companies for the invaluable time they accorded me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

ACRONYM

AIDE	Automated Interface Design and Evaluator
AUE	Automated Usability Evaluation
CTTE	Concur Task Tree Environment
DRUM	Diagnostic Recorder for Usability Measurements
EMA	Automatic Analysis Mechanism for the Ergonomic Evaluation of User Interfaces
FLUD	Framework for Logging Usability Data
FLUDViz	Framework for Logging Usability Data Visualization
GOM	Goal, Operators, Methods and Selection
HCI	Human Computer Interaction
KRI/AG	Knowledge Based Review of User Interface.
MAPI	MUSiC Assisted Process Improvement
MUSiC	Measuring the Usability of System in Context
MHP	Model Human Processor
MRP	Maximum repeating patterns
NIST	National Institute of Standards and Technology
QUIP	Quantitative User Interface Profiling
RAD	Rapid Development
SE	Software Engineering
SEO	Search Engine Optimization

STN	State Transition Network
SUMI	Software Usability Measurement Inventory
TLX	Task Load Index
UEM	Usability Evaluation Methods
UI	User Interface
UIDE	User Interface Design Environment
USINE	Usability Evaluator
UX	User Experience
WAMMI	Website Analysis and Measurement Inventory
WebCAT	Web Category Analysis Tool
WebRemUsine	Web Remote Usability Evaluator
WebSAT	Web Static Analyser Tool
WebVIP	Web Variable Instrumenter Program
WIMP	Windows, Icons, Menus and Pointers
WUP	Web Usability Probe

DEFINITION OF TERMS

The following definitions are provided to ensure uniformity and understanding of these terms throughout the study. The researcher developed some definitions not accompanied by a citation.

Learnability: - How easy it is for the user to accomplish bask task for the first time they encounter the design (Neilson, 2003).

Violation: - User click stream that do not follow predefined clickstream paths.

Clickstream pattern: - The path a user takes while navigating from site to site, and from page to page within a site. It is a comprehensive body of data describing the sequence of activity between a user's browser and any other Internet resource, such as a web site or third party ad server (Runkler & Bezdek, 2003).

Subsessioning:- Dividing clickstreams into session

ABSTRACT

The need of developing more usable web applications has motivated the development of a number of techniques, methods and tools to address web application usability issues. In recent years many automated remote usability evaluation methods have been employed. Despite the advantages offered by these methods, automated remote usability evaluation is still a difficult and a time consuming task to a developer partly to due to complexity of the tools used and partly due to expertise required. This thesis presents an automated remote usability evaluation application which employs the use of dialogue designs to determine usability of a web application. Sate transition network was preferred in this work because they are the most intuitive among dialogue designs. A sequence of system states for carrying out a specific task in a web application can be identified and subsequently, clickstream path for this task can be developed. The format of information presentation on a dashboard is also important as it should be easily read and understood by developers. Graphical and tabular presentation were identified as suitable formats. Localhost of the developed application was evaluated for usability using the pre-defined paths and the results observed. The level of usability of the developed application was determined using metrics of usability elements. A task completion rate of 81.67 % and errors percentage of 80 % were realized, these indicates a relatively effective system. Time-based efficiency of 0.04 goals/min with an overall relative efficiency of 60.67 % was achieved. On satisfaction level, NASA-TLX indices suggest that the total workload experienced was lower and a good SUS score of 70/100 was also achieved. Analysing data obtained from proxy server logs file can reveal usage pattern of users and this gives a highly improved understanding of users' behaviour. This information can then be utilized in improving a web application interface.

Keywords: *Asynchronous, Auto-logging, Click-stream, Dialogue design, Paths, Usability, Violations.*

CHAPTER ONE

INTRODUCTION

1.1 Background

Usability is considered to be one of the most important quality factors for web applications, along with others such as reliability and security (Offutt, 2002). Web applications are currently the backbone of business and information exchange, and are therefore the initial means to present products and services to customers. They are also employed by governments' agencies to disseminate relevant information to her citizens. (Adrian et al., 2010).

Petrie and Bevan (2009) observed that when people use these web applications, they may encounter usability problems and concluded that this is due to the fact that applications are becoming more and more interactive, complex and packed with features. The ease or difficulty (usability) that users experience with these web applications determines their success or failure. Many web-based interactive systems have been developed in the past few years and authors of these applications usually create them with content and structure from their own perspective rather than user perspective. Therefore, usability evaluation techniques which are specifically crafted for the web applications have therefore become critical.

According to Ivory and Hearst (1999), usability is a multidimensional concept and a commonly accepted definition is still lacking. Nearly every field of research has its' own definition. These differences in the definition of usability directly affect how it is

evaluated, since methods or techniques employed in evaluation may focus on different aspects of the term usability. A wide range of web interface application usability evaluation techniques have been proposed, and a subset of these are currently in common use. Some evaluation techniques, such as automated testing, can only be applied after the interface design or prototype has been implemented. Others, such as formal evaluation, can be applied in the early stages of design. Each technique has its own requirements, strengths and weakness. Generally different techniques uncover different usability problem.

For many years research has been focused on automated remote usability evaluation. Perhaps the reasons are reduction of cost (hiring and travels expenses) and getting real-time information. While numerous automated remote usability evaluation methods have been proposed, a survey of these methods carried out by Emilio and Adrian (2009) revealed that there are four types. Among these methods, the approaches based on user testing have attracted much attention recently and have demonstrated excellent results since these methods rely heavily on the increasing internet connections in working environments. This thesis was developed from these efforts.

Despite the potential advantages that automation of usability evaluation have, automatic generation of information by tools employed is quite underexplored. Nearly all automatic methods and their approaches have been limited to providing information in formats that require expertise to review and analyse. Therefore, there is a widely recognised need for a tool that provides specific usability issues to developer remotely. This thesis presents

the development of an application which provides practical support in evaluating user interface remotely, and can reduce the cost of employing the services offered by a usability expert as well as providing information to developers in time. This approach offers considerable advantages for usability evaluation.

Dialogue designs provides an excellent approach for development of clickstreams pattern. STN was preferred in this work because they are the most intuitive among all formalisms. It assumes that a dialogue essentially refers to a progression from one state of the system to the next in the system state space. Each pattern make a path and has a specific usability issue to identify. Clickstream deviation from predefined STN pattern signifies a potential usability problem and therefore analysis is required so as to identify specific usability problem. Subsessioning clickstreams provides a more comprehensive analysis and hence accurate information can be generated but optimisation of subsessions make it make tedious (Sadagopan & Li, 2008).

Remote evaluation of an interface usability using asynchronous auto-logging methodology is outlined in this thesis, and the development of an automatic general application that is used for identification of interface usability issues as well as user behaviour analysis is provided. This will improve the current practices of web applications usability approaches. The thesis is organized as follows.

1.2 Statement of the problem

Despite growing interest and rapid growth of automated remote usability evaluation, analysis of the methods and approaches used reveals that evaluation is still a complex and

a time-consuming process and often carried out by an expert. This could be due to large amount of data which is difficult to analyse and partly due to complexity of tools used. For many years research has been focused on automated remote usability evaluation. Perhaps the reasons are reduction of cost (hiring and travels expenses) and getting real-time information.

While there has been extensive literature examining efficiency of approaches used, documented work reveals that information is generated in formats which developers cannot interpret easily. Extensive research has been carried out on automating usability evaluation. Many approaches and tools have been developed from these efforts but relies heavily on evaluator's (expert) judgement as a source of information. This service is expensive to a developer and therefore, there is a need to develop an application that automatically generate information to developers.

This thesis presents asynchronous auto-logging methodology application. The application carries out web interface evaluation remotely by following pre-defined clickstream pattern which are derived from dialogue designs (STN). Logging-on the user clickstream in order to capture user generated events is done followed by analysis of this clickstreams which can provide useful hints regarding possible usability problem. This application allows developers to directly get information regarding usability of their web applications.

1.3 Objectives of the study

1.3.1 Broad objective

To explore various automated remote usability evaluation approaches with a view of developing an application that will collect and provide usability information to developers remotely.

1.3.2 Specific objectives

i. To identify automated remote user session evaluation approaches.
ii. To examine determination of usability problems by the approaches identified in (i) above and the format of information presentation.
iii. To identify gaps in the literature concerning use of the identified approaches by developers.
iv. To develop asynchronous auto-logging application, use it in determination of usability problem and test its usability.

1.4 Research questions

This thesis was designed to answer the following questions.

i. What are the approaches used in automated remote usability evaluation of web applications

   interface?

ii. How do the approaches in (i) above determine usability problem?

iii. What are the challenges of using the approaches by a developer?

iv. How can an application that can identify a potential usability problem and provide information remotely be developed?

1.5 Justification

An automated user session evaluation is an effective approach since developers are interested in information of usability of their applications. Asynchronous auto logging methodology would therefore be the best approach to capture such valuable information.

1.6 Scope of the study

This thesis work encompassed the following:

- Review of automated remote user session evaluation approaches.
- Determining of the effectiveness and adequacy of these approaches in information generation to developers.
- Developing automatic remote usability application and testing the effectiveness of the application in determination of usability issues.

Chapter 1 has presented the introduction, statement of the problem, research questions, significance of the study, definition of terms, and scope of the study. Chapter 2 begins with a brief overview of automatic usability evaluation process. It presents and discusses a background of automated remote usability evaluation, existing methods, current focus of research and limitation for future research, approaches in each method and tools in each of these approaches.

Chapter 3 describes methodology used to achieve the research objectives. It also describe methods used to develop and test the application. Chapter 4 presents the results obtained from the developed application and discussions. Chapter 5 discusses the implication of research findings and the limitations of this approach. Finally, Chapter 6 presents conclusions and suggests promising ways to expand this approach to better support usability evaluation.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

Since early years of existence, internet has made massive advances in transformation into a form we know of today (Alexiei & Sarah, 2014). Web applications have come to play a central role in the world of modern business and information dissemination. In turn, this shift has imposed a need of professionals to develop and maintain web applications. Developing and maintaining web applications is not enough for it is the level of its usability that determines effectiveness and efficiency to achieve a desired goal as usability of an interactive software application is seated within the design of the user interaction components.

It is therefore a crucial exercise to do usability evaluation of a web application interface. Evaluation of users in their real work environment is very costly, and hence an obstacle to a sound evaluation. Increased internet connection in working environments makes it possible to collect users' session data and automating analysis is possible for identification of usability issues. Automated remote usability evaluation of an interface is therefore a promising area to pursue as it is equally efficient and effective for application usability evaluation as traditional usability evaluation (Betra & Bishu, 2007).

There has been extensive work relating to automating usability evaluation and this has resulted to development of a number of tools which perform website usability analysis. Dingli and Misfud (2011) suggested that despite advantages that automatic usability

evaluation provides, tools for evaluation should be in conjunction with standard usability evaluation techniques and information should also be presented in a format that is easily understood by developers.

It has also been observed that, with advances in technology automation has never been effective enough to enable tools to give/display specific usability information and even further critique a web application, it has also been noted there is inconsistency in results generated as each tool gives different information for same application interface. Past attempts of automation have also resulted in tools that are too specific and application related, very costly and oversimplified (Dingli & Misfud, 2011). One of the reasons of automating usability evaluation, is to reduce the cost of hiring an expert.

As a means of addressing these problems, an automatic application based on user testing method of usability evaluation was developed. It analyses clickstream patterns of a remote user for usability issues using a set of predefined clickstream pattern. Experimental results shows that clickstream pattern for usability can be developed and hence used for identifying usability issues.

Andrian, Emilio and Silvia, (nd); Ivory and Hearst (1999); Tonio, Fabio and Vagner, (2011); Fidas et al., (2007); Ivory and Hearst (2001) and Jesper, Anders, and Janne (2000) developed various academic papers and prototypes to automate web application usability evaluation remotely all with various degrees of success. Automation provides excellent solution towards overcoming limitation of having expertise in usability evaluation and

hence, developers will concentrate in development and improving existing applications using information from remote tools.

With a wider sampling of click-streams, pattern-matching analysis is a promising research area to pursue. Pattern-matching approaches compare click-streams of a developer with the click-streams of a user. Some task-based approaches reports substantial analysis data, this data could be compared to usability guidelines in order to support automated analysis. Chapter 2 provides an extensive review of the literature and research related to automatic remote interface evaluation. The chapter is divided into sections that include 2.1 Introduction 2.2 Automation of usability evaluation 2.3 Critique of the existing literature, Summary, and Research gaps.

2.2 Automation of usability evaluation

Automation is of usability extremely beneficial. It is not only assisting designers in creating more usable websites, but also enhances the internet users' experience on the web and increases their level of satisfaction (Alexiei & Sarah, 2014). The increasing diffusion of interactive software based applications in an increasing number of context, purposes and types of users require better methods for evaluating designs and obtaining more reliable information.

Usability evaluation with users in their work environment is one of the most fundamental, as it provides developers with direct information about how users interact with their applications. Ivory and Hearst (2001) reported that automation has been used

predominantly in capturing user session data and analysing the data according to some metrics or some models.

Although server logs i.e. entries of requesters' IP address, time of request, name of page requested and URL referencing the requested page does not only provide a record of user interaction that occur on the client side, they also provide more comprehensive usage data which can be used to determine usability of an interface (Fidas et al., 2007)

According to Madan and Dubey (2012), several taxonomies for classifying automated usability evaluation have been proposed. These methods can be broadly categorized into two groups namely, evaluation through user participation, also referred to as empirical methods and evaluation through expert analysis, also referred to as inspection methods.

Another overview of approaches to evaluation is provided by Ivory and Hearst (1999) in their work on *automated usability evaluation of WIMP and Web interfaces*. They revealed that there are four general methods for evaluation. These are user testing, inspection method, analytical modelling method and simulation method.

This was suggested to be one of the most complete studies in the *systematic review of usability evaluation in web development*, work done by Emilio and Adrian (2009). This taxonomy was used in this work and user testing method was pursued. User testing involves logging client requests automatically, log file analysis is a heavily used methodology for evaluating web interfaces.

According to Ivory and Hearst (1999), there are four general approaches for analysing web log files. These are metric based approach, pattern-matching approach, task-based approach, and inferential approach. These approaches focuses on user experiences and web applications' capacity to meet intended purpose. Usability evaluation is done through use of developed tools. Numerous tools (like) are available in the five approaches outlined in this work and these tools have become more and more advanced in recent years.

2.2.1 Metric based approach.

In this approach valuable information is obtained through observation of users' interaction with a web application. Analysis of what is observed with help of video recording provides highly effective means of evaluating usability. This approach gives reliable measures of effectiveness and efficiency of a web application use, by evaluating the extent to which specific task goals are achieved and the time taken to achieve these goals (Kirakowski & Bevan, 1999)

A measures of time spent unproductively can also be determined and valuable data about the source of such difficulties is obtained. These measures enables comparison of users' working on an application and alternative designs with earlier versions of systems design. According to Kirakowski and Bevan (1999), diagnostic information helps identify where specific problems are encountered and where improvement need to be made.

2.2.1.1 DRUM

DRUM is an example of metric based tool. It a software tool for video- assisted usability evaluation. It was developed by Macleods and Rengger in 1993. It provides a practical support for observational evaluation of usability thus reducing video protocol analysis (Macleod & Rengger, 1993). A lot of data is collected when using DRUM. Video recording provide considerable advantage for usability evaluation as video clips of end user working with a web application can provide persuasive evidence. It has been developed for over two decades and several versions have been released.

DRUM provides support for the usability analyst throughout the process of observation-based usability evaluation. It assists in many aspects of usability evaluation e.g. management of data, task analysis, video control and analysis of data. It speeds up the analysis of video recordings and automates some activities. It also build up time-stamped log evaluation sessions and calculates performance measures and metrics. Reports of evaluation findings are in text files formats for data storage, this allowing flexible compatibility with word processors, spreadsheets and statistics packages.

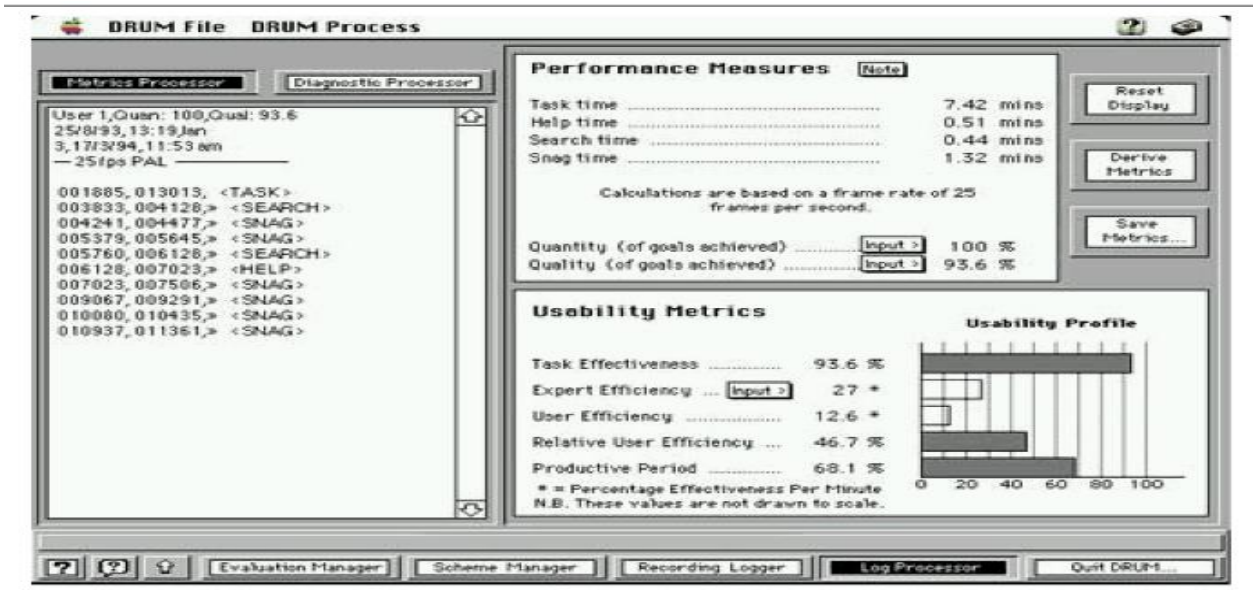


Figure 2.1: Usability evaluation using DRUM

2.2.1.2 MAPI

MAPI was later in developed 1995. The tool is supported by a software tool (DRUM) which greatly speeds up analysis of the video, and helps manage the evaluation. It ensured greater effectiveness and cost benefits. It provides facilities for editing and browsing task analysis. It has a graphical user interface, on-line context-sensitive and hyper-text help (Alty & Diaper, 1993). It runs on apple Macintosh, drives a variety of video machines and supports the management of data at different stages of observational evaluation from selected users. It is divided into modules each supporting a different aspect of usability evaluation.

The service providers of MAPI have identified different configurations in which MUSiC methods can satisfy the objectives of MAPI partners. According to Kirakowski and Bevan

(1999), the objectives of the MAPI projects are to show that MUSiC methods can assist companies to improve their processes, and to highlight ways in which the current MUSiC tools need to be developed and delivered in order to be cost-effective. Bevan (2009) pointed out that this tool supports quantitative usability measurements by organising and analysing user-based evaluation and delivers and measures diagnostic data. This work cannot be done by a developer since expertise is required.

2.2.1.3 Other video assisted usability evaluation methods

Camtasia Studio and Camtasia for Mac are software suites, created and published by TechSmith, for creating video tutorials and presentations directly via screencast, or via a direct recording plug-in to Microsoft PowerPoint. The screen area to be recorded can be chosen freely, and audio or other multimedia recordings may be recorded at the same time or added separately from any other source and integrated in the Camtasia Studio component of the product. Both versions of Camtasia started as enhanced screen capture programs and have evolved to integrate screen capture and post-processing tools targeted at the educational and information multimedia development marketplace (Nagaraju & Trivikram, 2013).

Figure 2.2: Camtasia screen recorder

Morae is software also developed by TechSmith for market research. It gives businesses, market research firms and academics a chance to do usability, product design, hardware and prototype testing, and focus groups as well as qualitative market and educational research. This software is meant to help in identifying site and application design problems. Some new features include highlight reels in high-quality flash video, sharing and viewing across platforms, and the ability to share findings on Screencast.

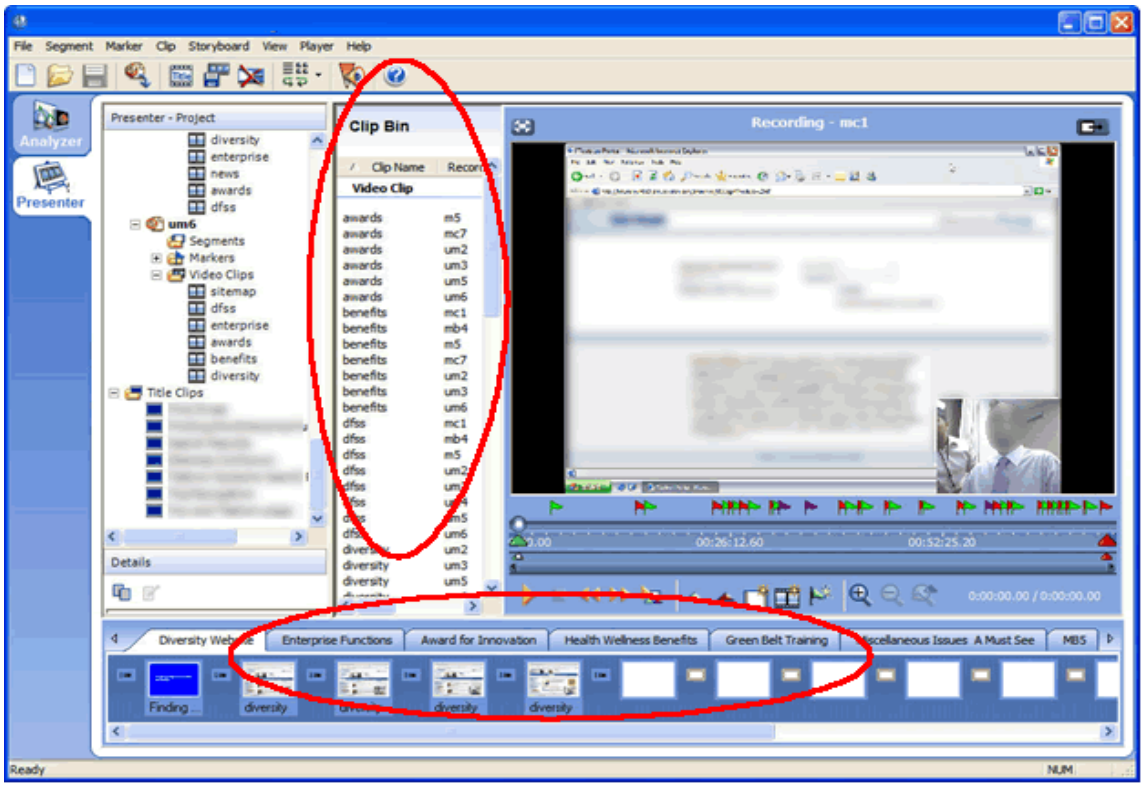


Figure 2.3: Morae screen cast

Silverback is another usability tool used for creating projects, recording the videos, and exporting the videos. In creating projects, a user selects new project followed by new session. Once a new session is started, a user will be tasked to choose a specific session subsequently a preview is presented. This will be followed by selection of record button and the screen will prompt the user to hit the space bar to get started. Once that happens, automatic capturing of the screen, it's associated clicks and recording a user's reactions to questions occurs. After a session is finished, exporting the video for further analysis is carried out (Andrew, 2009).

Snagit is another video assisting usability evaluation tool. It is a screen capture and editing tool. It has a print screen feature that captures animation, graphics and text in the form of images as well as video. This tool also allows a user to grab the entire desktop, a region, a window, or a scrolling window. Snagit has ability to combine images, blur and magnifying effects, arrows, callouts, torn edges, etc. The final capture can be saved in various formats, a URL or embed code can be produced, or it can be added to Microsoft Word, PowerPoint, MindManager, or Microsoft OneNote directly from the product interface (Tom, 2013). Images captured are automatically saved to the Snagit Library in order to locate past captures.

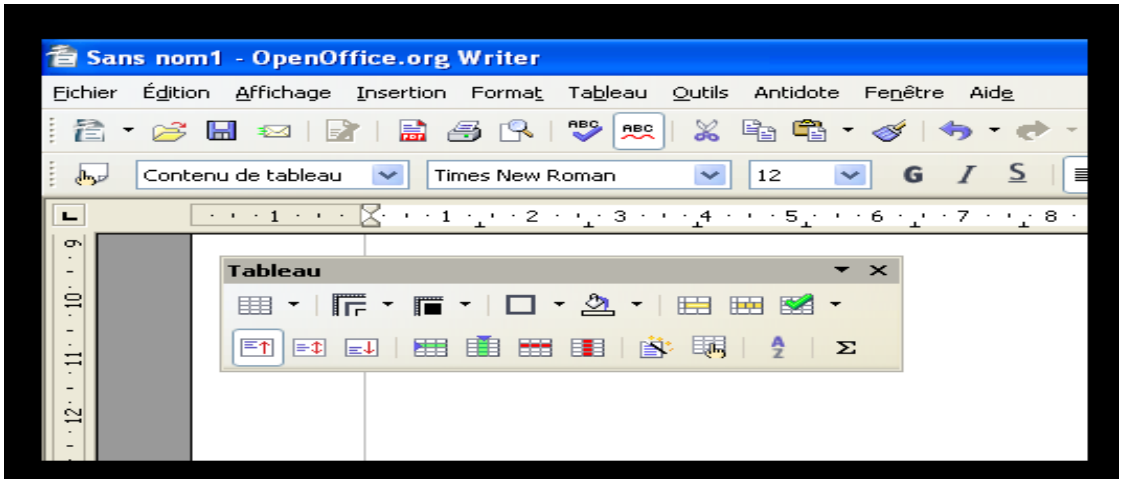


Figure 2.4: Snagit video recorder

2.2.1.4 NIST Web Metrics

The objectives of the NIST Web Metrics were to explore the feasibility of a range of tools and techniques that support rapid, remote, and automated testing of website usability. Web metrics consists of, among others, the following prototypes:

2.2.1.4.1 WebSAT

Jovanovic, Kruegel and Kirda (2006) reported that WebSAT checks the HTML codes of web pages against usability guidelines, either its own, or a set of guidelines. It can also check individual web pages or an entire website. According to Chiew and Salim (2003), WebSAT uses a set of heuristic rules to evaluate the use of tags in web page design. These rules do not form a comprehensive set of guidelines. However, they are a sample set of typical rules to demonstrate the feasibility (and limitations) of an automatic checker. The rules are grouped into six categories as follows: Accessibility, form use, performance, maintainability, navigation and readability.

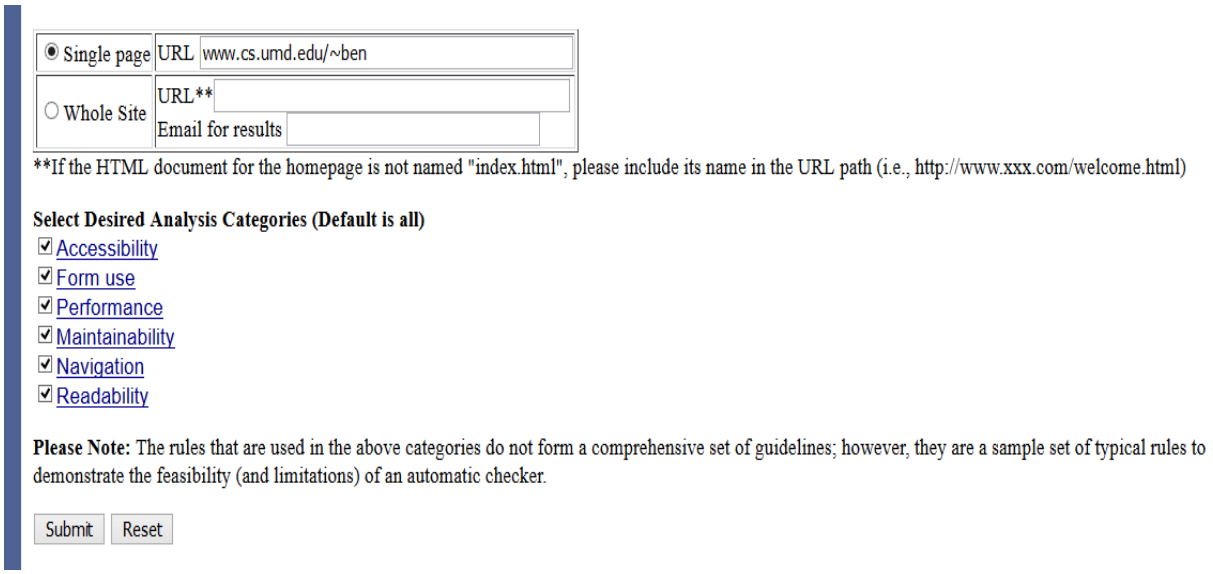
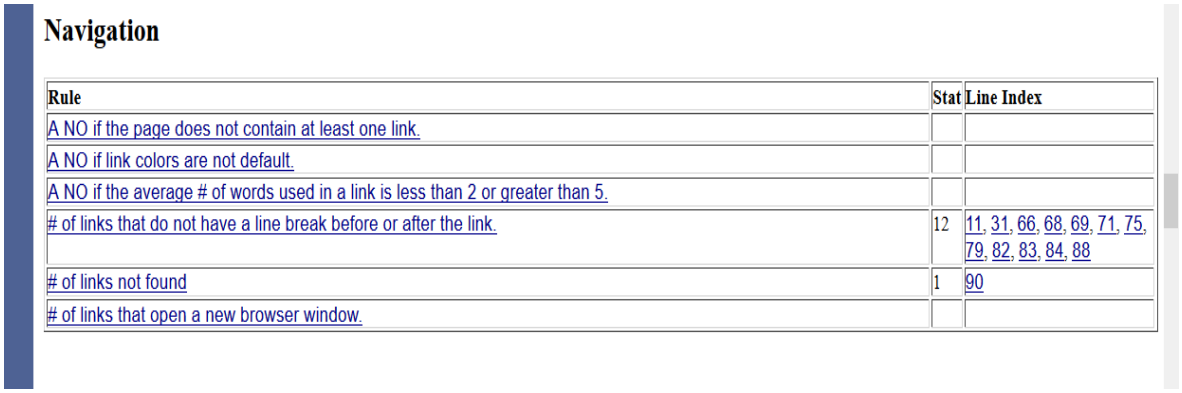


Figure 2.5: setting-up WebSat for usability evaluation

After the user hits the submit button, a set of analysis results is displayed in the form of tables like this one



Navigation

Rule	Stat	Line Index
A NO if the page does not contain at least one link.		
A NO if link colors are not default.		
A NO if the average # of words used in a link is less than 2 or greater than 5.		
# of links that do not have a line break before or after the link.	12	11, 31, 66, 68, 69, 71, 75, 79, 82, 83, 84, 88
# of links not found	1	90
# of links that open a new browser window.		

Figure 2.6: Results from a WebSat tool

2.2.1.4.2 WebCAT

According to Sanfit and gallagos (2008), WebCAT helps usability engineers to construct and conduct a simple category analysis across a website. It also helps IT auditors determine how well the categories and items are understood by users.

WebCAT is an extensible tool to extract meta-data and generate RDF descriptions from existing Web documents. Implemented in Java, it provides a set of APIs (Application Programming Interfaces) that allow one to analyse text documents from the Web without having to write complicated parsers. WebCAT is a Java package for extracting and mining meta-data from Web documents. It is an important building block in several other text mining tools, handling most of the low level processing operations (De Marsico & Levialdi, 2004).

De Marsico and Levialdi (2004) further pointed out that the software is divided in three "core" blocks: the parser, the miners and the output generator. The parser performs the "low-level" processing operations, receiving Web documents in HTML or converted from other different file formats. It analyses the textual contents, divides the text into atomic units, and extracts hyper-links and meta-data. Afterwards, the miners make use of the parsed information, generating additional meta-data properties for the documents. Examples of miners include the language identification and classification modules. Finally, the output module produces RDF documents from the extracted/mined metadata.

2.2.1.4.3 WebVIP

Chiew and Salim (2005) developed WebVIP. It is a tool used to augment traditional user testing on a given set of tasks. Usability engineer uses this tool to instrument a website for local or remote testing. WebVIP performs several steps in the process of instrumenting a website. It starts with making copies of a website to a test directory, it then parses the HTML pages and lastly adds JavaScript code to the page. This tool collects user interaction data after which it prompts usability engineer to choose the types of user events to log.

These choices are converted to JavaScripts which are then inserted into each page of the target website. In subsequent operation, users interact with the newly instrumented pages, a logfile is created that contains time-stamped entries of the specified interaction events.

The Web Variable Instrumenter Program (WebVIP) allows rapid instrumentation of a web site by the webmaster. This way webmaster can capture a log of user behavior on the site. After the instrumentation of the web site, test subjects are asked to complete tasks. Their interactions like manipulating buttons and checkboxes and navigating among pages are captured in a log file. This file is analysed later. After the analysis user interaction patterns are determined (De Marsico & Levialdi, 2004).

2.1.1.4.4 FLUD

John, Heer, Waterson and Landay (2000) developed FLUD. During user interaction with the pages, the selected events types are captured, time-stamped, and recorded in session-specific FLUD log files. Log files are created in the standard FLUD format and hence used by any application that accepts such files. Currently, NIST FLUD parser is the only application that has been developed to interpret FLUD log files.

According to Chiew and Salim (2005), the parser generates a pretty-print html version of the log data which can be used to inspect user interaction patterns and data files suitable for visualization with VisVIP . The Web Metrics team plans to develop other FLUD-compliant modules, such as a statistical analysis tool and a visualization for exploring intra-page interaction patterns. It is encouraged to use the format as a basis for creating additional usability exploration tools.

The FLUD (Framework for Logging Usability Data) provides a representation of user interaction. User interaction shall be convenient for usability test that means interaction shall be general enough to support the test (Jugini & Laskowski, 2001). The Framework for Logging Usability Data consists of a file format and a parser for representation of the behavior of website users. The log data is used for analyzing and improving the usability of web applications. As captured data is complicated a file format is needed to allow various software components to exchange information.

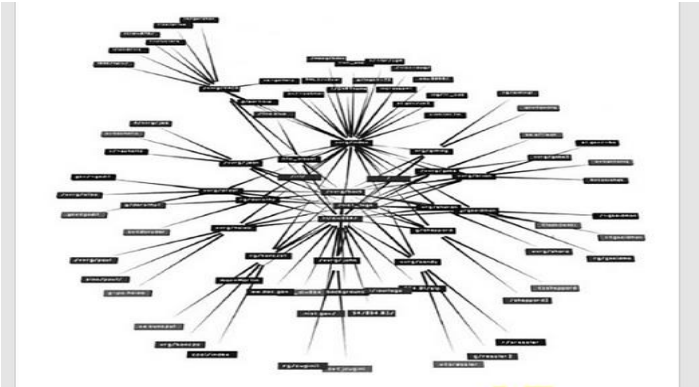


Figure 2.7: VISVIP Visualisation

2.2.1.4.5 SEO

According to (SiteAnalyzer, nd), SEO is a website optimisation tool which allows developers to analyses their website and generate a multi-point audit sorted by category e.g. accessibility, design, texts, multimedia and networking etc. Analysis report is made of several criteria based on the optimization of server configuration, HTML tagging, text content, multimedia content, internal and external networking and page popularity. This complete checkup made of more than 60 criteria based on SEO design, content, performance, accessibility and security provides issues necessary for fixing.

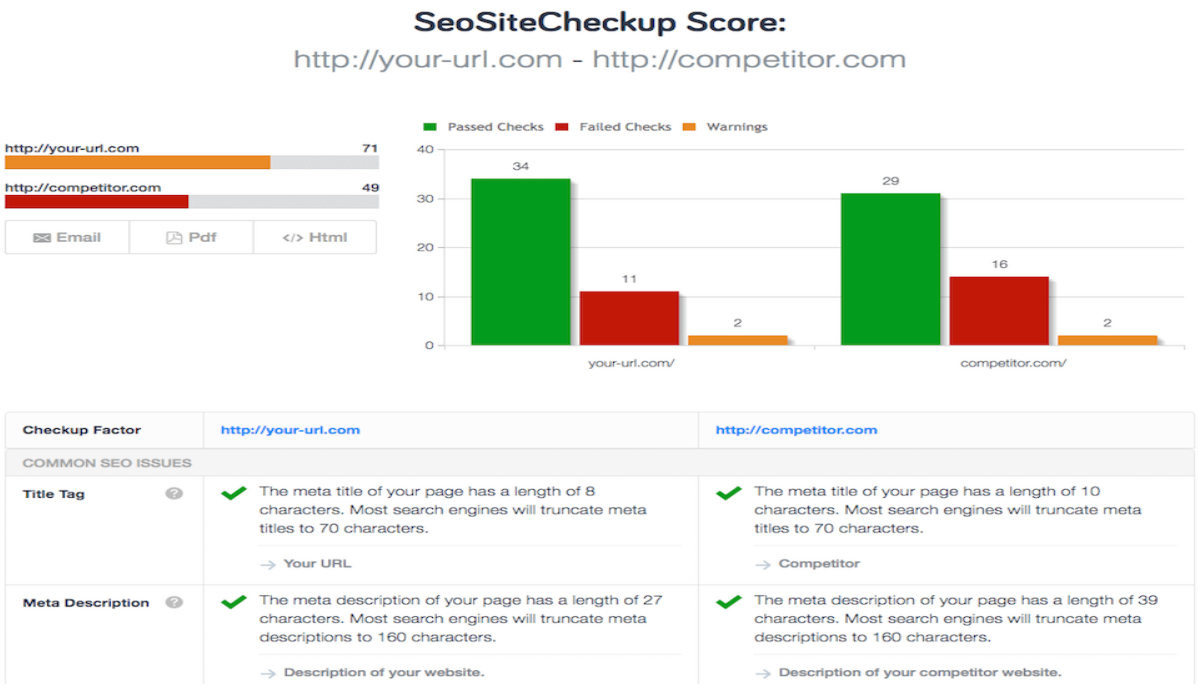


Figure 2.8: SEO usability evaluation tool

2.2.2 Pattern-matching approach.

One major property of pattern-matching approach, is that a set of usability patterns are used to determine the efficiency and effectiveness. A mismatch signifies a usability design problem and therefore a solution of usability issue should be established. Another major property of the usability pattern inspection is to provide an expert with a very rich set of guidelines (usability patterns) on the level of the interaction design (Julie, 2012)

These patterns are collected from several widely known collections and confirm them with rationales. This enables the expert to concentrate on the design level as opposed to the task or user level. As argued earlier in the statement of problem of this thesis, there is a need for evaluation methods, which can efficiently be applied by non-experts in small and

medium enterprises as opposed to highly specialized usability experts in large companies. This work emphasizes on design oriented approach because it could be more natural than abstract guidelines like heuristics (Propp & Forbrig, 2009).

2.2.2.1 WUP

Tonio et al., (2011) developed WUP, a tool that captures logs generated at client side. The tool support in automatic analysis of the captured logs. A users perform some predefined tasks specified by an evaluator and evaluation is carried out by comparing the click-stream patterns of a user and a developer. A difference in steps can imply a potential usability problem.

Usability evaluation of web application is still a difficult and time consuming task, often performed manually using this tool. WUP considers client-side data on user interactions and JavaScript events and allows the definition of custom events, giving evaluators the flexibility to add specific events to be detected and considered in the evaluation.

Tonio, Farbio and Vagner (2011) commented that the tool supports evaluation of any web application by exploiting a proxy-based architecture and enables the evaluator to perform a comparison between actual user behaviour and an optimal sequence of actions.

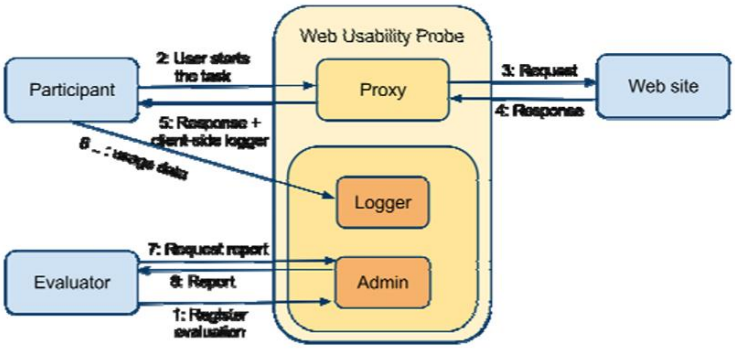


Figure 2.9: An overview of the approach proposed in web usability probe

WUP is a good tool for usability evaluation as it is easy to use but, users must be willing to participate during evaluation session as an instructor provides instructions using a dialogue box. A user can be hesitant to participate in a session, and information is provided in graphical form which requires an expert to interpret.

2.2.2.2 Google Analytic

Google analytic is a tool used for a website usability evaluation, it has a potential of being used for evaluating web application interface. The only limitation is that it mainly provides information in statistical form. Google analytic has the potential to be configured to capture custom events at client-side. It offers a number of statistical information and reports, but it is rather limited in terms of number of events that it captures for each session (Dumas & Redish, 1999).

Google Analytic supports a wide range of features e.g. simple installation through integration with Google Analytics API which authenticate and select the site to be tracked.

Universal or the asynchronous Google Analytics tracking code is the fastest and most reliable tracking code, gives visitor metric dashboards in WordPress install Option to enable demographics and interest reports etc. (Chris & Syed, nd).

Google Analytics' approach show high-level dashboard-type data for the casual user and more in-depth data further into the report set. Funnel visualisation techniques enables Google Analytics to analyse and identify poorly performing pages. This analysis indicate users' referrer, time spend and their geographical position. It also provides more advanced features, including custom visitor segmentation e.tc

According to Clifton (2012), it has an admin section, under which Account, Property, and View menus can be selected. Then under the View level menu settings, Goals can be selected. This will bring up the Goal overview screen which shows which goals one has enabled and gives the option of creating new goals as shown in fig 2.2

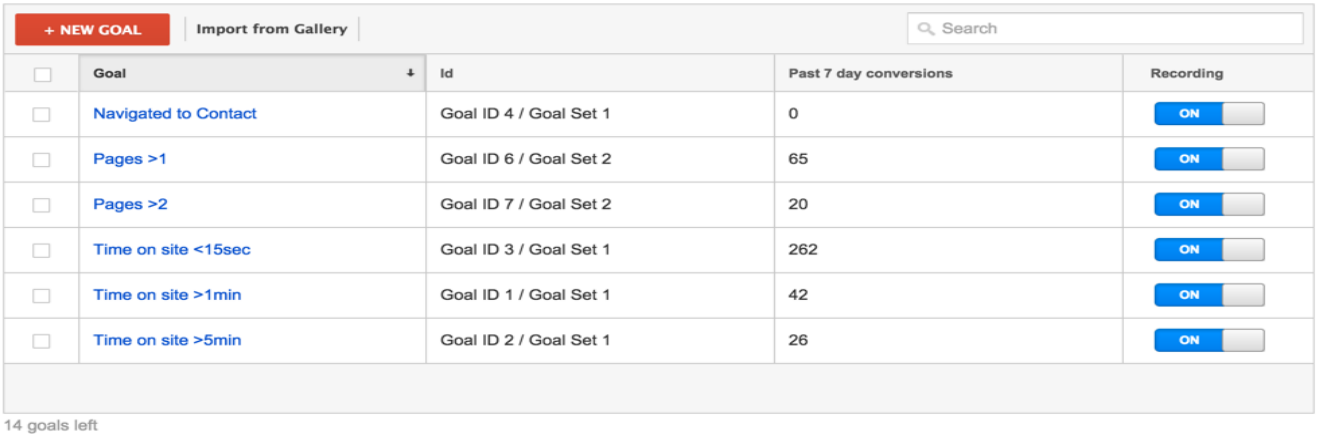


Figure 2.10: Goal setting in Google Analytic

2.2.2.3 Click streams data analysis of web traffic.

Riivo and Karl (2009) in their work *on click streams data analysis of web traffic,* proposed data mining of frequent user access patterns from web log files. Their key objective was to find out the most frequent browsing pattern by analysing the visitors' browsing session. Frequent data set being used means that some sets of data are eliminated because it does not belong to any said data set, and hence some usability problems cannot be identified. One of the key aims of automation in remote evaluation is to increase the coverage of feature to be evaluated which this method does not address.

2.2.3 Task-based approach.

Propp and Fabrig (2010) in their work on *A Virtual Smart Environment for Usability Evaluation* proposed a task model tool for usability evaluation of different applications. This tool conducts experts' evaluation and user studies during a user-centred design process, supporting iterative evaluation. Task models can enhance visualisation of the usability test log as a part of the methodology for design of usable products.

Work done by Le, Chaudhuri, Chung, Thompson and Demiris (2014) further pointed out that this tool visualises usability logs that uses hierarchical structure of the task models to group and visualise observers' annotations. This way of visualisation is very natural and allows investigation of the test dynamics and comparison of participant's behaviour in various parts of the test.

Maly and Slavik (2007) in their work on:- *Towards visual analysis of usability test logs using task models,* proposed use of task models to enhance visualisation of the usability test logs. The tool uses hierarchical structures of task-models to group and visualise observed annotations. Visualisation method is based on alignment of the visual representation of task where binding between the task model and log visualisation is used. In HCI research field, task models are widely used for model-based development of interactive systems.

2.2.2.4 EMA

Task based approaches analyse discrepancies between designers' task model and actual use. Some like EMA uses data flow task-models along with behaviours heuristics to detect specific user behaviours that may indicate usability problems and immediate task cancellation or a shift in direction during task completion is two behaviours detected by this tool (Ivory & Hearst, 2001).

EMA uses a manually-created data-flow task model and standard behavior heuristics to flag Usage patterns that may indicate usability problems (Balbo, 1996). It extends the MRP (maximum repeating patterns) approach (repeated command execution) to detect additional patterns, including immediate task cancellation, shifts in direction during task completion, and discrepancies between task completion and the task model. EMA outputs results in an annotated log file, which the evaluator must manually inspect to identify usability problems (Balbo, 1996).

2.2.2.5 QUIP

QUIP is another tool used in this approach. It aggregates traces of multiple user interactions and compares the task flow of users to design task flow. It encodes quantitative time-based and trace-based information into directed graphs. In order to use this tool, the evaluator must instrument the UI to collect the necessary usage data, and manually analyse the graphs to identify usability problems (Hammontree, Wailer & Nayak, 1994).

Each node represents a user action, and arrows indicate actions taken by users. The width of arrows denotes the fraction of users completing actions, while the colour of arrows reflects the average time between actions (darker colours correspond to longer time). Reprinted with permission of the authors.

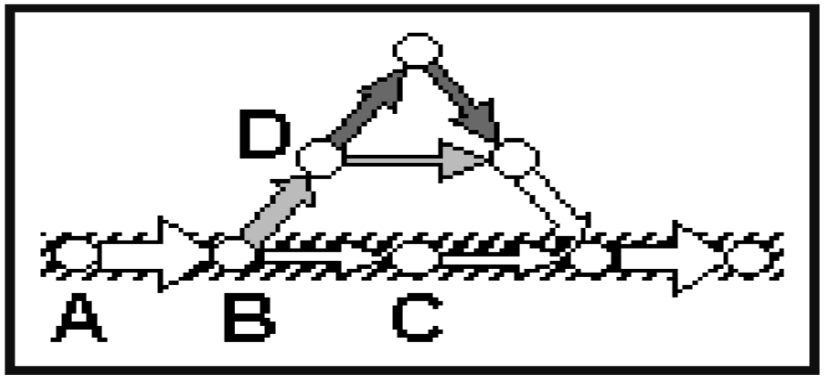


Figure 2.11: QUIP's comparative representation of task flow of users to the design flow

2.2.2.6 Multi ModalWebRemUsine.

Fabio and Carmen (2007) developed Multi ModalWebRemUsine. The basic idea of this tool is to analyse user logs through the semantic information contained in the task model. On one hand there is a task model that describes how developers expect that users perform their activities and on the other hand, there are logs indicating the action performed by users while interacting with an application.

If a task sequence performed diverts from those enabled by the task model, there is clearly a mismatch that needs to be analysed by an evaluator because either is too rigid or there is something unclear in the user interface which prevents a user from performing the expected sequence of task. In general, Multi ModalWebRemUsine is mainly based on comparison of planned user behaviour. However, requires an expert to set-up CTTE and to carry out analysis (Shneiderman, 2010)

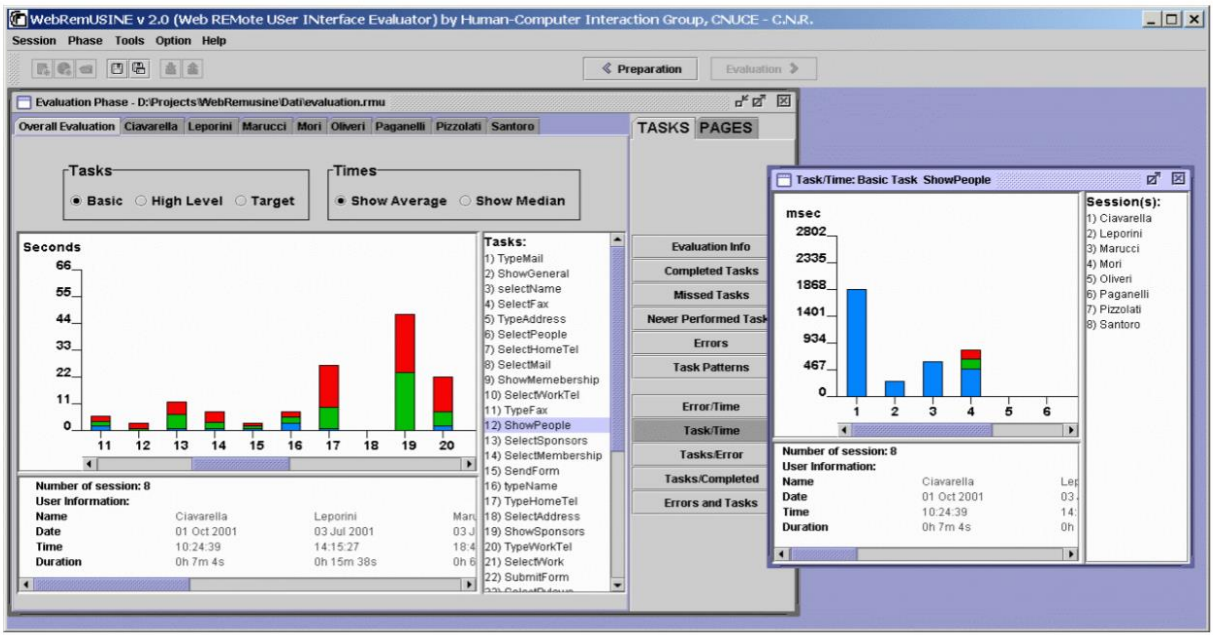


Figure 2.12: Multi ModalWebRemUsine interface.

2.2.4 Inferential approach.

Inferential analysis of application log files includes traffic-based analysis i.e. page per visitor and time based analysis i.e. click paths/ page view duration. These techniques attempt to infer something about usability or users interest in page content from log files. This analysis is largely inconclusive, since server logs are only a partial trace of user behaviour. Furthermore, server log files are missing valuable information about what task users want to accomplish (Paganelli & Paterno, 2003)

Since Web- based interfaces have grown in importance, new and adapted UEMs have emerged to address this type of user interfaces. One of the most representative examples is the heuristic evaluation method proposed by Nielsen (2003). Inferential analysis of web

log files includes statistical, on-line analytical processing, mining and visualisation. Statistical approaches include traffic based analysis (e.g. page-per visitors or visitors-per-page) and time based analysis (e.g. click paths and page-view duration) (Valdman, 2001).

Software tools such as WebTrend, NetGenesis and NetTracker are used in this approach and provides analysis in graphical and textual format (i.e. statistic tables and charts). This statistical analysis is largely inconclusive for web server logs since they provide a partial trace of user behaviour and timing estimates may be skewed by network latencies. Server log files are also missing valuable information about what task users want to accomplish. Nonetheless, statistical analysis technique have been useful for improving usability and enable ongoing, cost-effective evaluation throughout the life of a site (Creswell, 2009).

2.2.2.7 On-line Analytical processing

OLAP is an approach for interactively exploring multi-dimensional data. It enables the analyst to construct custom views of the data so that one can explore relationships among various dimensions to make inference about the data. OLAP may enables the evaluators to identify areas of the site that need to be improved. The interactive exploration and summarisation of web log data that is possible with OLAP is in stark contrast to statistical analysis approaches.

2.2.2.8 Web usage mining and visualisation

According to Ivory and Hearst (1999), web usage mining entails the application of data mining techniques such as deriving association rules or decision tree to analyse the

behaviour captured in log files. The evaluator can use machine Learning techniques to develop predictive models of user behaviour.

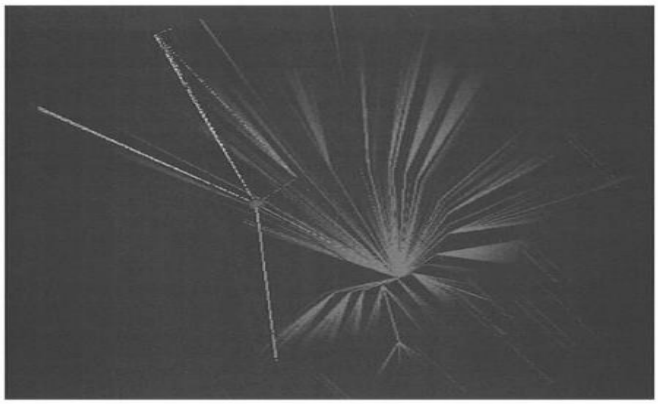


Figure 2.13: Dome Tree visualisation(Adrian F, Emilio I, and Silvia A, nd)

2.2.5 Automated inquiry method

Automated inquiry are used to measure user satisfaction. Users are asked to fill a questionnaire or make a comment on a dialogue box. The data collected will be analysed to generate information that is used to describe usability of a web application. This describes the overall usability of a web interface. Varghese and Pfleger (2012) developed SUMI is one of the tools used in automated inquiry. The tool provides results based on extensive standardisation database e.g. spreadsheet

2.2.6 ZapMeta

ZapMeta is a Meta search engine. A user can search listings from multiple search engines all in one place. ZapMeta gets its' search results from AOL Search, Google, Yahoo, Ask.com, Wisenut and Yahoo Web. It also retrieves directory results from the Open

Directory and shopping listings powered by PriceGrabber. Search results give the options for sorting results by relevance, popularity, title, source, or domain (Nagaraju & Trivikram, 2013).

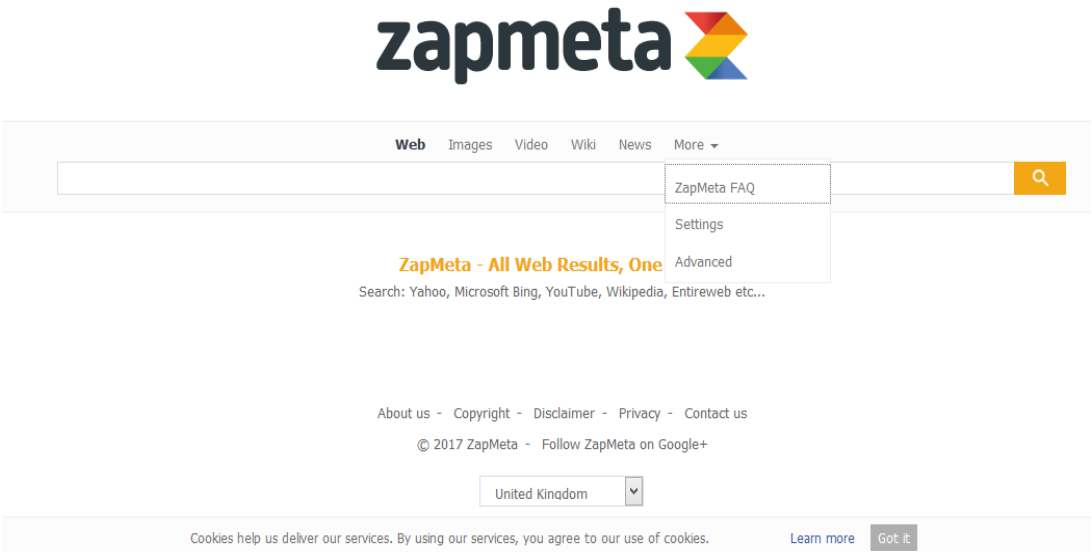


Figure 2.14: Zapmeta

2.2.7 WAMMI

It measures a website in terms of attractiveness, controllability, efficiency, helpfulness and learnability. Visitor comments to open questions are gathered and a range of different opinions from various visitors are presented. The presentation include detailed statement analysis (cross tabulated with the profile scores). When a web application is above average (50) according to their international database a green colour is used and red is used when below.

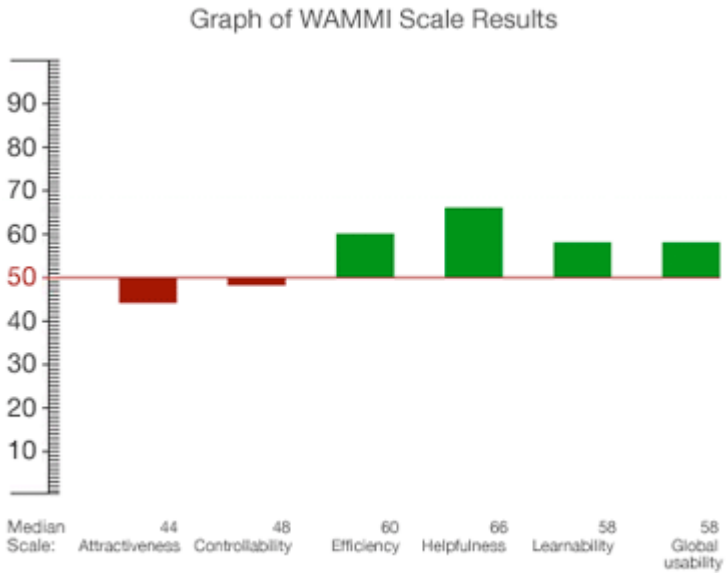


Figure 2.15: WAMMI auto-generated results

2.2.8 Critique of the existing literature, summary and research gaps

Since inception, internet has made massive advances in transformation into a form we know of today (Alexiei & Sarah, 2014). Web applications plays a central role in the world of modern business and information dissemination. This shift has in turn imposed a need of professionals to develop and maintain web applications.

Development and maintenance of web applications is not enough for it is the level of usability of this web applications that determines effectiveness and efficiency to achieve a desired goal. Usability of an interactive software application is seated within the design of the user interaction components. Therefore, it is crucial to do usability evaluation of a web application interface. Evaluation of users in their real work environment is very costly. This is an obstacle to sound evaluation.

Betra and Bishu (2007) observed that increased internet connection in working environments makes it possible to collect users' session data and automating analysis is possible for identification of usability issues. Automated and traditional usability evaluation are equally efficient and effective for application usability evaluation therefore, automated remote evaluation is a promising area to pursue.

There has been extensive work relating to automating usability evaluation which has resulted to development of a number of tools which perform website usability analysis. One of the reasons for automating usability evaluation is to reduce the cost of hiring an expert. .Automation can provide excellent solution towards overcoming limitation of having expertise in usability evaluation and hence, developers will concentrate in developing and improving existing applications using information from remote tools (Propp, Buchholz & Forbrig, 2008).

Despite advantages that automatic usability evaluation provides, it has been observed that with advances in technology, automation has never been effective enough to enable tools to generate specific usability information and even further critique a web application. It has also been noted that there is inconsistency in results generated as each tool gives different information for same application interface (Dingli and Misfud, 2010).

Work by Dingili and Misfud (2011) on *USEFul* further points out that attempts of automation have also resulted in tools that are too specific and application related, very costly and oversimplified. And therefore, tools for evaluation should be in conjunction

with standard usability evaluation techniques and information should also be presented in a format that is easily understood by developers.

This thesis presents asynchronous auto-logging methodology application that follows pre-defined clickstream pattern which are derived from dialogue designs (STN). Logging-on the user clickstream in order to capture user generated events is done. Analysis of this clickstreams can provide useful hints regarding possible usability problem. This application allows developers to directly get specific information regarding usability of their web applications.

Ivory and Hearst (2001) provide a good discussion of tools for usability evaluation according to a taxonomy based on four dimensions. These are, method class (the type of evaluation), method type (how the evaluation is conducted), automation type (the evaluation aspect that is automated) and effort level (the type of method required to execute the method). According to this classification, Asynchronous auto-logging solution for usability testing involves automatic capturing logs generated at client-side and supporting automatic analysis to ease the identification of the usability issues.

Google Analytic has the potential to be configured to capture custom events at client-side and it offers a number of statistics information and reports, but it is rather limited in terms of number of events that it captures for each session (Tonio, Fabio & Vagner, 2011). Model-based approaches have been used to support usability evaluation. An example of model based approach is WebRemUsine developed by Paganelli and Paterno (2003) which is a tool for remote usability evaluation of web applications through browser logs

and task models. This tool require expertise to use and it is also time consuming to a developer.

According to Tonio, Fabio and Vagner (2011), WUP is a tool that captures logs generated at client side, the tool will then support in automatic analysis of the captured logs. Users perform some predefined tasks specified by an evaluator. Evaluation is carried out by comparing the click-stream patterns of a user and a developer. A difference in steps can imply a potential usability problem.

WUP is a good tool for usability evaluation as it is easy to use but, users must be willing to participate during evaluation session as an instructor provides instructions using a dialogue box. OLAP is an approach for interactively exploring multi-dimensional data. Relationship among various dimensions are explored in order to make inference about the data.

Ivory and Hearst (1999) observed that OLAP may enable the evaluators to identify areas of the site that need to be improved but interactive exploration and summarisation of web log data that is possible with OLAP is in stark contrast to statistical analysis approaches. SUMI results are analysed by a computer programme called SUMISCO (Kirakowski, 1996). This is the simplest approach but it requires the efforts of web application user to get the data. The feedback from the user could be subjective. It is also time consuming and determination of specific usability issue requires expertise (Kujala, Vaananen, Karaponas & Sinnela, 2011).

2.2.9 Asynchronous Auto-Logging Methodology.

2.4.9.1 Dialogue Designs.

One important aspect of HCI is the dialogue which is the interaction that takes place between a human user and the computer. A dialogue refers to the structure of the interaction. Dialogue design helps to understand a web application design better. Formal representation makes it possible to analyse dialogues to identify usability problems. There are several formalism that can be used to represent dialogues (Shneiderman, 2010). STN was preferred in this work because they are the most intuitive among all formalisms. It assumes that a dialogue essentially refers to a progression from one state of the system to the next in the system state space.

2.4.9.2 Determination of Paths.

The model supports evaluation of web application by exploiting clickstream pattern and comparing them with predefined paths. It is based on an intermediate proxy server whose purpose is to capture log file from remote users. The model does not require use of plug-in installation in the client side or specific client configuration.

The script used by the model are stored in the proxy server and thus there is no security conflict when the page is accessed from the client since the page appears as coming from the proxy server. As users carry out their tasks, clickstreams are recorded and accessed by the model for analysis of specific usability issues. A link from the start to the end form a path. A path has a specific goal. A menu in a website makes the start of a path and the last

link is the end of a path. Each path has a specific goal/objective. A clickstream pattern that does not follow the predefined click paths are considered as a violation. Paths created using dialogue design eases identification of usability issues, hence developers will access the information from remote tools. All logged data are sent asynchronously to the model while the users move from one page to another.

2.4.9.3 Determination of usability issues using paths.

The development of a proxy-based model considering client-side data encounters different challenges regarding to identification of specific usability issues from user interaction data. For example if a user moves from one link to another then back, this pattern may imply a usability issue according to the models' design. Assigning a minimum time to a particular task and creating associations among system states may also help in identifying potential usability issues.

Dingli and Misfud (2011) suggested that, usability is generally measured using a number of observable and quantifiable metrics that overcome the need to rely on simple intuition. ISO/IEC 9126-4 recommended that usability evaluation metrics should include effectiveness, efficiency and satisfaction. Each of these recommended elements of usability has two metrics as shown in table 2.1 below. These metrics were used to determine the suitability of the application for use in website usability evaluation.

Table 2.1: Metrics for usability evaluation

	Metrics
Effectiveness	Task completion rate
	Number of errors
Efficiency	Time based efficiency
	Overall relative efficiency
Satisfaction	Task level satisfaction
	Test level satisfaction

CHAPTER THREE

METHODOLOGY

3.1 Research design

This chapter details out the research methodology for the present study and explains a suitable approach to achieve the research objectives. The specific objectives of this study were, to identify automated remote user session evaluation approaches, to examine determination of usability problems by the approaches identified and the format of information presentation, identify gaps in the literature concerning use of the identified approaches by developers and to develop asynchronous auto-logging application, use it in determination of usability problem and test its usability.Literature review, research objectives and the four research questions directed this research work. The research was carried out in the phases as shown in Figure 3.1 below.

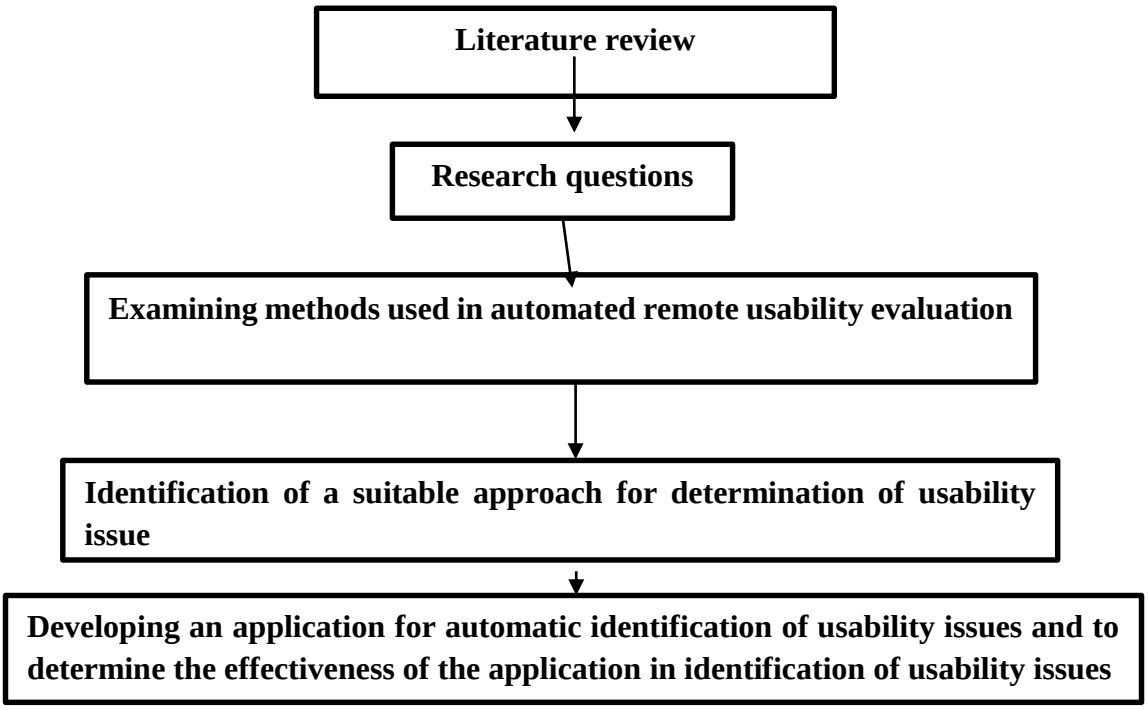


Figure 3.1: Phases of research

A study of automated remote user evaluation approaches, various tools in each approach and the nature of information generated by these tools was carried out to determine existing automated remote user evaluation technologies that have been implemented. This, established a background support for the document. Literature review revealed that there many taxonomies of classifying automated usability evaluation methods.

A classification of automated remote usability evaluation approaches based on (Ivory & Hearst, 1999) was used in this work. Five approaches and some selected tools in each of these approaches were comprehensively and critically surveyed. The survey revealed that

expertise is required in order use these tools, and that analysing data produced by these tools is time consuming.

In order to determine efficiency and adequacy of information generated, automated remote user session evaluation technologies and their tools were studied. For highly interactive web applications to be achieved, frequent updates and modifications are needed, and therefore effective usability methodologies that can be applied and easy to use are required. Automated remote usability evaluation is a new paradigm that will provides a solution to this challenge.

This thesis is of the same argument as Fidas et al., (2007) that automated remote usability evaluation techniques are effective and efficient that they can be used to evaluate web-based applications in a complementary way to the traditional usability methodologies conducted in usability laboratories. Thus, if automated remote usability evaluation is implemented it can provide valuable information of user experience. The main challenge in using this evaluation techniques is some properties of usability cannot be evaluated and the approach only suggests a potential usability issue rather than determining the specific usability problem.

It is rather considered as the ability of the system to provide the necessary data to the usability engineers in order to evaluate the usability of a system by modelling user interactions, and analysing subjective feedback from the users. The engineer, then relays the information to a developer. A general conclusion derived from the survey is the

absence of an integrated tool to support sufficiently automatic capturing and analysis of a user behaviour and automatically generating a report to a developer.

Dingli and Misfud (2011) pointed out that as far as automated remote usability evaluation activities are concerned, there are many potential benefits, including reducing the costs of evaluation and improving consistency in evaluation results Tools that support the analysis of log files and automatically generate information to a developer seems to be a good solution.

Metric-based analysis of log files provides advanced data analysis techniques through observation made with help of video recording, hence valuable information is provided. But this evaluation method is mainly affected by network latencies and also requires the efforts of an expert. Although a usability expert can gain a better insight into users' behavioural patterns, this method is time consuming.

In this study work, the researcher found that pattern- matching analysis of log files provides the most complete solution, if a combination of user reporting techniques and performance measurement indicators are integrated. Information provided by this approach require an expert to review and analyse.

Fidas et al., (2007) suggested that, if the interest is shifted towards participatory design techniques, then automated inquiry is a good choice. The transformation of semi-structured methods like card sorting to game-like elicitation methods offered by MindCanvas improves user engagement and makes participant recruitment easier.

Task-based analysis of log files determines the discrepancies between designers' task model and actual use. It uses visual representation of data and requires an observer to annotate. Given a wider sampling of usage data, log file analysis using task models and pattern-matching approaches are promising research areas to pursue.

Task-based approaches that follow the USINE model in particular provide the most support among the methods surveyed since it compares a task model to usage traces. This helps usability engineer to understand user behaviour, preference and errors.

Although authors claim that this approach works well for WIMP UIs, it can to be adapted to work for web application usability evaluation. In addition, since USINE already reports substantial analysis data, these data could be referenced to usability guidelines in order to support automated critique.

Inferential-analysis of log files provides statistical, on-line analytical processing, mining and visualisation but this analysis produce quantitative measure that an evaluator must interpret to identify usability problem. Timing estimates is also skewed.

Another conclusion derived is that remote usability inspection methods seem not to be supported by any of the presented tools. To some extent this is expected, since these methods are different in the usability problems they evaluate and seem to benefit less from the remote character of the techniques, as presented in this document.

Remote usability is a challenging and interesting area of research and technology. Click-stream data are easy to collect but analysis and interpretation are time-consuming and

require expertise which developers do not have. Therefore, a tool with useful integrated functionalities that can determine usability problem could potentially enhance the overall evaluation process.

This thesis argues that, one way to expand the use and benefits of this automated remote usability evaluation is to leverage on dialogue designs. This is even more important for web interfaces, since server logs provide a complete record of user interactions and can follow a predefined pattern. These techniques also provide valuable insight on real usage data from remote users.

At the time of research, the literature seemed to report no solution to the problems of automatically evaluating web applications and automatically generating information to developers without employing the services of a usability engineer. To address this challenge, dialogue designs (STNs) were used. STN were used to identify click paths. This reflects a novel contribution of this work to the field of automatic web applications usability evaluation. Analysing Clickstreams Using Sub sessions (Jesper, Anders & Janne, 2000) was excellent but optimization of clickstreams is a problem.

However it is seen as one of the best attempts of identifying and analysing usability problems. Automation provides excellent solutions towards overcoming limitation of having experts in usability evaluation and hence a developer will concentrate in development and improving existing applications using information from remote tools. STN is a form of dialogue notation is a precise formal specification for the pattern or

structure of interaction between a user and a system. Fig 3.2 shows how the application can be implemented.

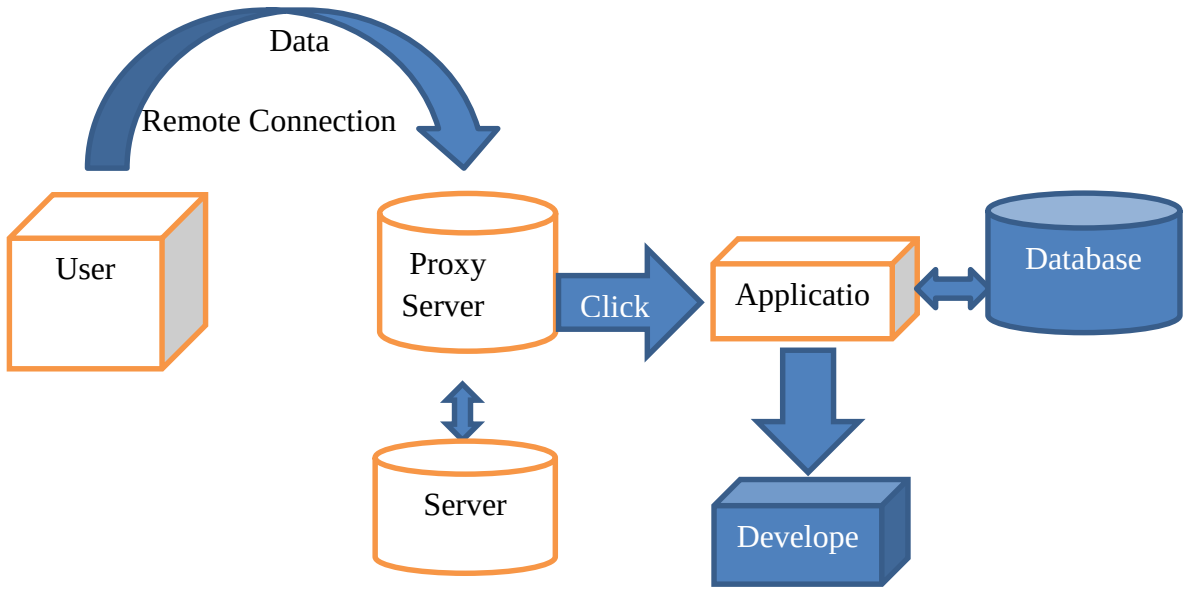


Figure 3.2: Asynchronous remote usability application

3.2 Development methodology

A system development method was used to develop and test the application. Waterfall model of system development was preferred. In a waterfall model, each phase must be completed fully before the next phase can begin. The waterfall model, offers numerous advantages for software developers. First, the staged development cycle enforces discipline: every phase has a defined start and end point, and progress can be conclusively identified through the use of milestones. The emphasis on requirements and design before writing a single line of code ensured minimal wastage of time and effort.

Getting the requirements and design first, improves the quality, it was much easier to catch and correct possible flaws at the design stage than at the testing stage, after all the components were integrated.

3.2.1 Requirements

Paths of the localhost of the application were identified using STN and the specific task of each path determined. A path is a sequence of links (clicks) from a start to an end. The preferred format (tabular and graphical) of information presentation to a developer was identified.

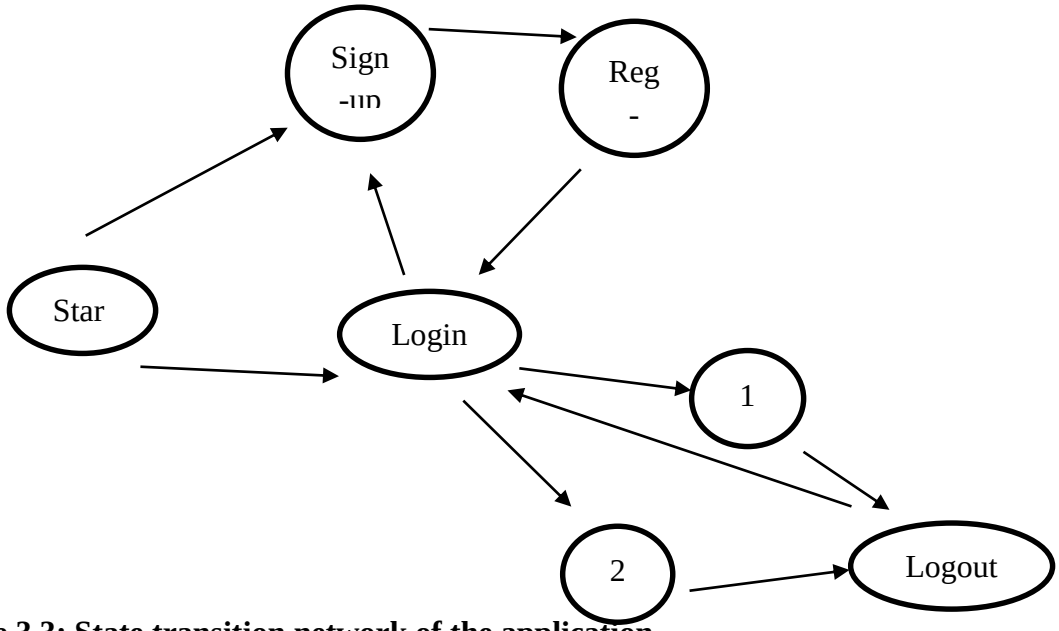


Figure 3.3: State transition network of the application

Key 1 select website to track **2** add website to track

System state Transition

Other possible requirements of the developed application were captured in this stage and documented in requirement specification document. Paths and clickstream patterns of the application's localhost were identified using STN and the specific task of each path determined as illustrated in fig 3.3 and fig 3.4.The focus then shifted to what is needed for the application to communicate to a developer. Transactional and report interface were then defined.

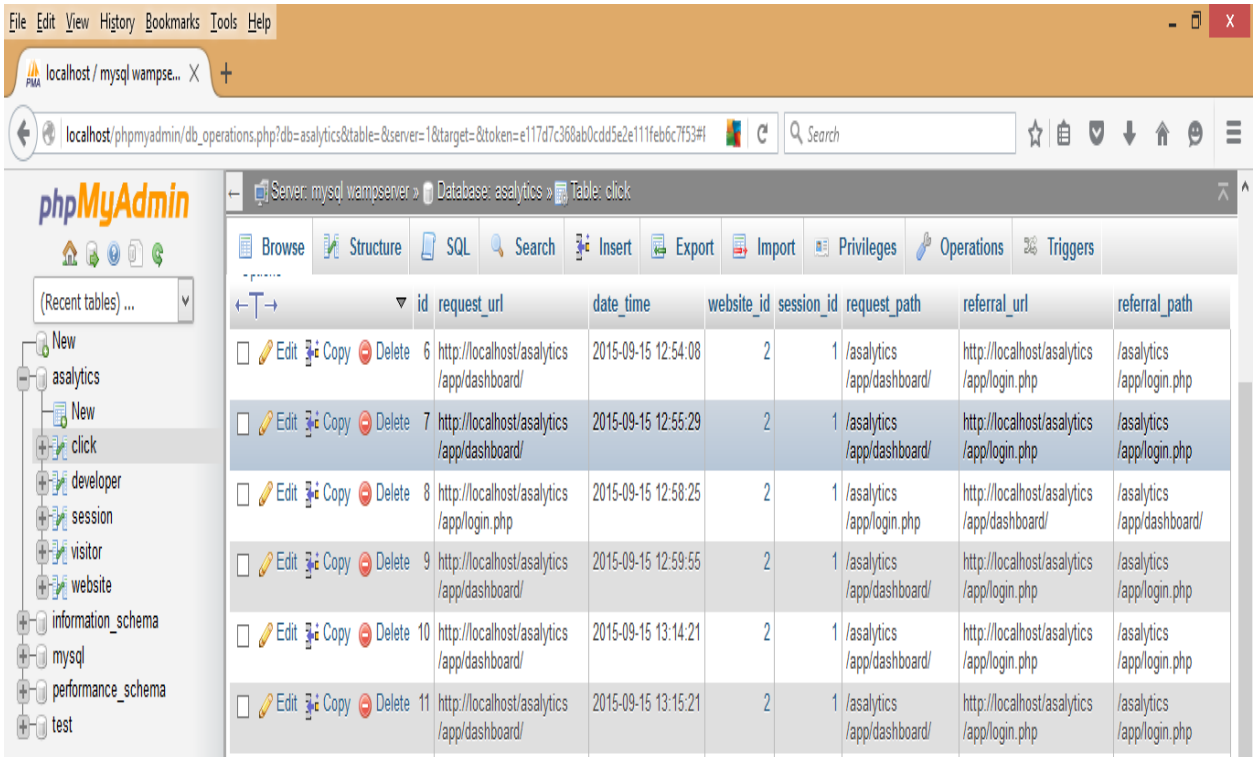


Figure 3.4: Clickstream pattern for the application's localhost

3.2.2 Design phase

System architecture and series of components interaction was defined at this stage. Details of how each component work and communicate with the application's interface was also

defined and clickstream pattern of the localhost's predefined paths were identified. The format of report presentation to developers was also identified.

3.2.3 Implementation

With the input from system design, the system was developed in small programs (units), these were later integrated and each unit tested for its functionality. Clickstream path framework was embedded to facilitate detection of violation.

3.2.4 Testing

This stage involved proving that the application meets each of the designs and requirement specifications. The working of the identified clickstream structure in detection of violation was determined. This evaluation was done in terms of violations in a click stream paths.

a) Illustration from the application

The localhost of the developed application was used in testing clickstream structure (pattern) violation in usability evaluation. For a certain task to be carried out in a particular system state, one or a sequence of task maybe carried out. This sequence of task is referred in this work as a path. In this developed application, clickstream pattern for a specific task forms a path. Any deviation from this predefined path is considered as a violation.

A scenario where a registered user intends to register a website at 2 in fig 3.3 and to evaluate it for usability issues was used. The predefined paths for a registered user is indicated are start, login and Add-website-to-track. A path indicated in start- sign-up -

register was chosen for the purpose of this study. The first system state is sign-up followed by register in the add website to register as shown in fig 3.5 below, since these are not the desired states, the developer will move back from register to start page then proceed to login page. After logging-in, the developer can register the website. JKUAT website was used for registration and subsequently for evaluation as shown in fig 3.5

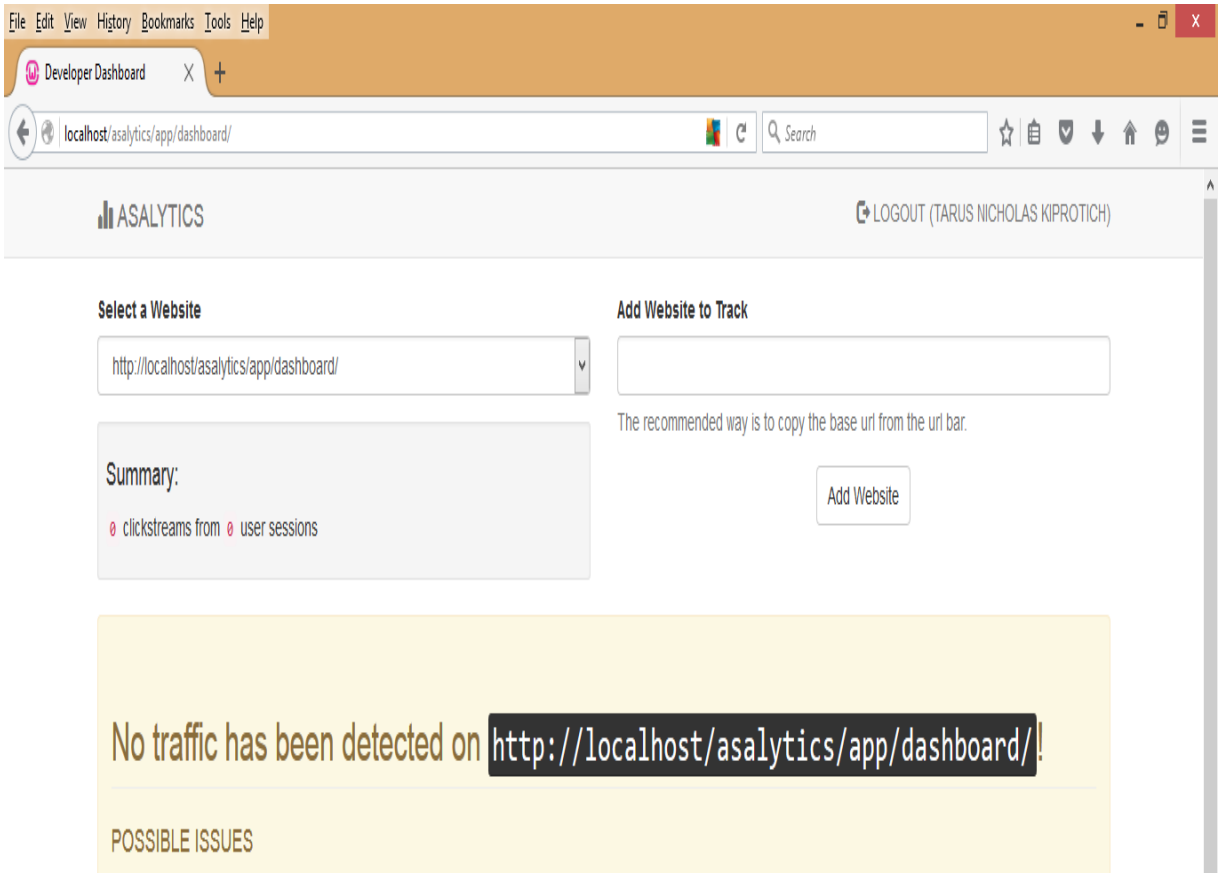


Figure 3.5: A page for adding website to track

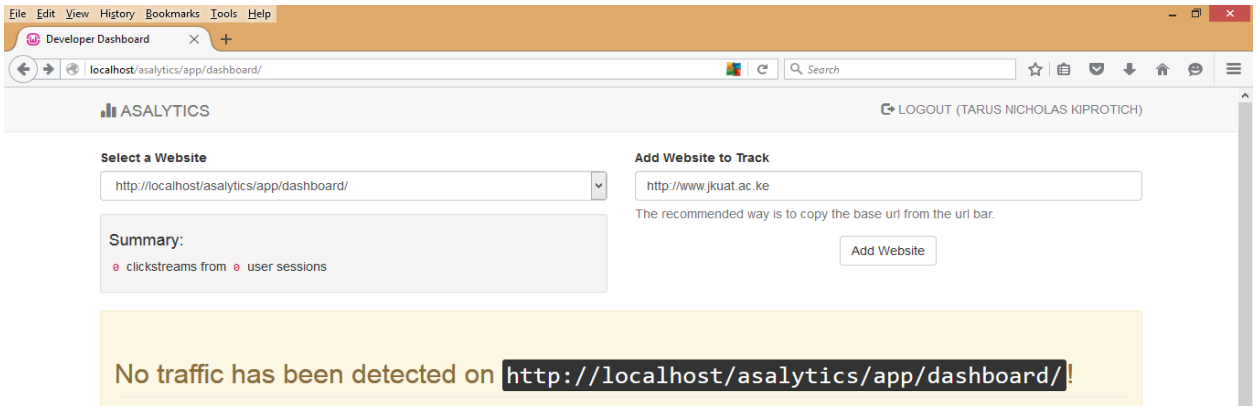


Figure 3.6: Registration of jkuat website

b) Experimental design

Procedure

Fifteen participants were recruited for the purpose of this study 10 from Visions Interactive and 5 from Interpay. These two, are IT companies based in Nairobi. The research was done in the two companies separately. To save on company's time, only 10 minutes was used for introduction and 15 minutes for testing the developed application. The task was further divided into two parts so as to allow key operations of the two companies to continue without interruption as one-half of their staff assisted in research the other half handled companies' businesses.

The researcher introduced himself to participants and screened them for knowledge in IT and familiarity in usability evaluation methods by randomly asking questions. From participants screened most them of them were identified to be having sufficient IT skills

and 2 were conversant with usability evaluation methods. The researcher further explained the purpose of research and gave participants the pre-defined tasks in appendix 3 of this work. These tasks are a representative of scenarios that a regular user would experience when using a website and also, provided an opportunity to gather information on variables such as time spent on completing a task and the number of errors made.

The four session's data was collected. The time for task completion was taken and the errors made by each participant noted. The researcher further noted the difficulty experienced by some participants' inability to finish tasks in the fifteen minutes allocated.

At the end of the test, the evaluator collected the subjective feedback, debriefed the participants and thanked them for their participation. It was through the researcher's personal and friends work relationship with the companies that made this research work possible. Emails, personal visit and telephone calls were used to make the two companies understand the purpose of the research and assure them that the data so provided will be used only for academic research.

Usability Metrics for Effectiveness

a) Completion Rate

According to Sauri (2011), effectiveness can be calculated by measuring task completion rate. Referred to as the fundamental usability metric, the completion rate was calculated by assigning a binary value of '1' if the test participant manages to complete a task and '0' if the participant does not. Effectiveness is represented as a percentage by using the equation below

$$Effectiveness = \frac{number\ of\ tasks\ completed\ successfully}{total\ number\ of\ tasks\ undertaken} \times 100$$

b) Number of Errors

Another measurement of effectiveness involves counting the number of errors the participant made when attempting to complete a task. Errors are unintended actions, slips, mistakes or omissions that a user made while attempting a task. They were recorded by the application as violations in this work and calculated as follows

$$Percentage\ error = \frac{number\ of\ errors}{total\ number\ of\ task} \times 100$$

Usability Metrics for Efficiency

According Sauri (2011), efficiency is measured in terms of task time which is the time (in seconds and/or minutes) the participant takes to successfully complete a task. The time taken to complete a task can then be calculated by simply subtracting the start time from the end time as shown in the equation below.

Task Time = End Time – Start Time

Efficiency can then be calculated by either time-based approach or by working overall-relative efficiency. Time-Based Efficiency is given by the equation below

$$Time\ Based\ Efficiency = \sum_{j=1}^{R} \sum_{i=1}^{N} \frac{n_{ij}}{t_{ij}} \div NR$$

Where:

N =The total number of tasks (goals)

R = The number of users

n_{ij} = The result of task i by user j; if the user successfully completes the task,

then $N_{ij} = 1$, if not, then $N_{ij} = 0$

t_{ij} = The time spent by user j to complete task i. If the task is not successfully completed, then time is measured till the moment the user quits the task

a) Overall Relative Efficiency

The overall relative efficiency uses the ratio of the time taken by the users who successfully completed the task in relation to the total time taken by all users. The equation can thus be represented as follows.

$$Overall\ Relative\ Efficiency = \sum_{j=1}^{R}\sum_{i=1}^{N} n_{ij}t_{ij} \div \sum_{j=1}^{R}\sum_{i=1}^{N} t_{ij} \times 100$$

Usability Metrics for Satisfaction

User satisfaction were measured through standardized satisfaction questionnaires which were administered after the usability test session.

a) Task Level Satisfaction

After participants attempting the task (irrespective of whether they manage to achieve its goal or not), they were immediately given questionnaires so as to measure the difficulty of using the developed application. The most popular post-task questionnaires are: ASQ, NASA-TLX, SMEQ, UME, SEQ. NASA-TLX (Appendix 2) scale was preferred and used in this work

b) Test Level Satisfaction

This was measured by giving a formalized questionnaire to each test participant at the end of the test session. It serves to measure their impression of the overall ease of use of the developed application. For this purpose, SUS (System Usability Scale) developed by John (1986) was used in this work as it is still valid to date. Participants ranked each question from 1 to 5 based on how much they agreed with the statement they were reading. 5 meant they agree completely, 1 meant they disagreed completely.

Information was finally gathered through analysis of collected data, questionnaires and interviews. Researcher observations were also done in order to substantiate the data gathered. Some of the participants were also contacted individually to build rapport and gauge their understanding about research specific objectives. All questionnaires were received and were found to be fully filled in. With the unconditional assistance from the two companies, data was collected comfortably from the respondents working in the two selected IT companies.

CHAPTER FOUR

RESEARCH FINDINGS

In this chapter the results of the developed application and data analysis are presented. The data were collected and then processed in response to the problems posed in chapter 1 of this thesis. Two fundamental goals drove the collection of the data and the subsequent data analysis. Those goals were to determine effectiveness of the developed application in identification of usability issues and determine the application's ease of use from user (participants) experiences. These objectives were accomplished. The findings presented in this chapter demonstrate the potential for merging theory and practice.

4.1 Finding from system development

Figure 4.1: Task 1, start page

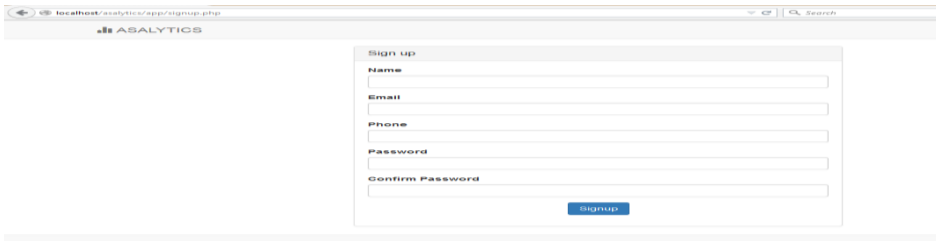


Figure 4.2: Task 2, signup page for a new user

Traffic of the localhost was detected and the report presentation is in graphical format while traffic is in tabular format as in the scenario where a developer logged in and signed up as shown in fig 4.1 and fig 4.2 respectively. The developer wanted to evaluate it remotely for usability issues and to reach this state according to the application's design, the user has to login, add website to track and then select website to track. These are predefined paths.

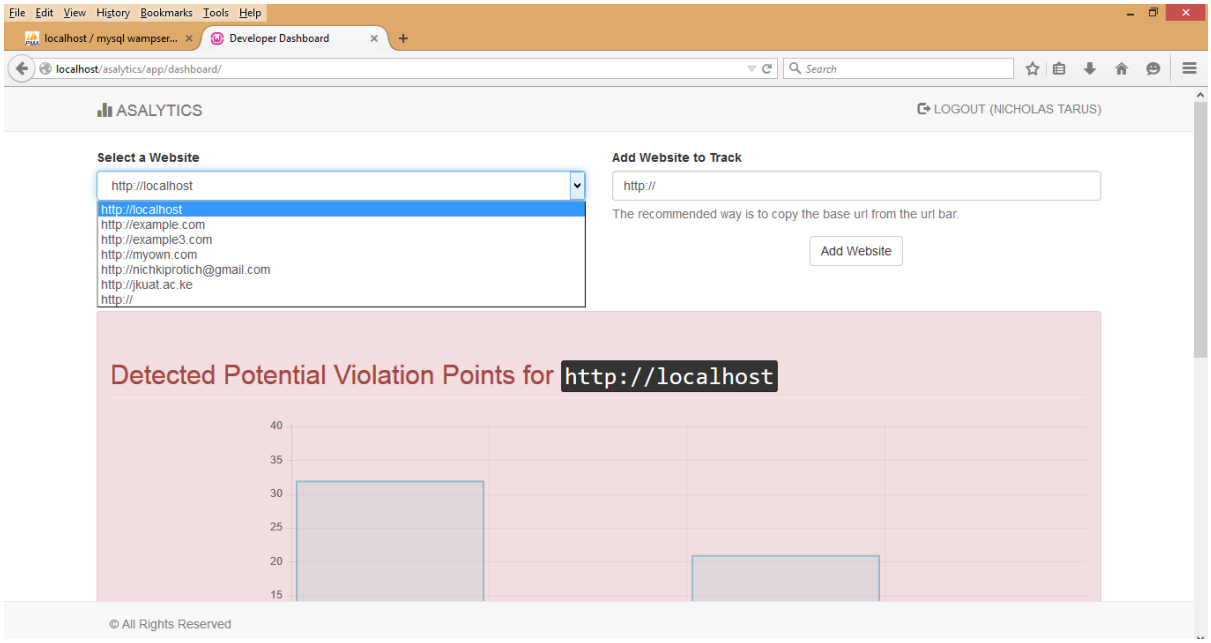


Figure 4.3: Task 3- Selection of unregistered website

Website to track was selected as shown in fig 4.3 but the only registered website the application could track is it's localhost and therefore website was to be added to the list of website to track as shown in fig 4.4 and 'select website to track' from the menu as shown in fig 4.5 chosen. The traffic could not be detected since the application was not remotely connected to http://www.jkuat.ac.ke fig 4.6.

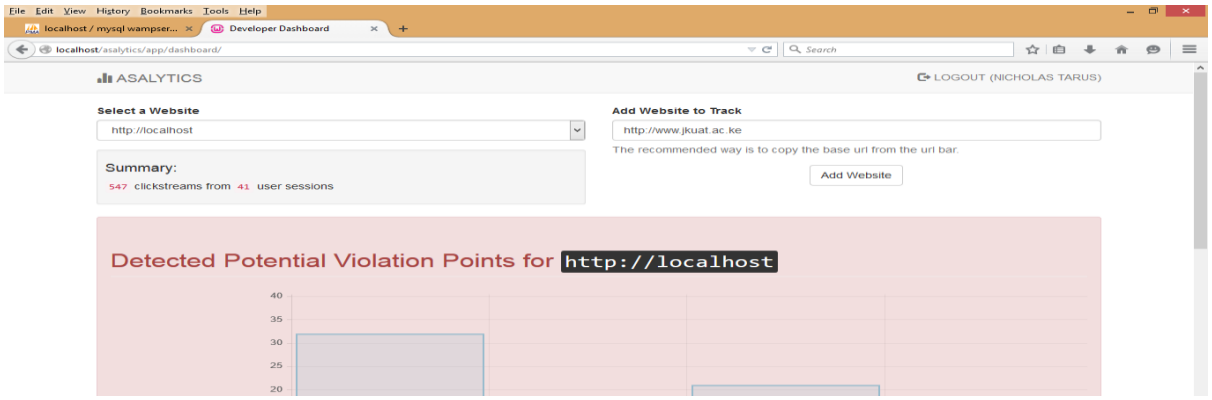


Figure 4.4: Task 4- User registers jkuat website (http://www.jkuat.ac.ke)

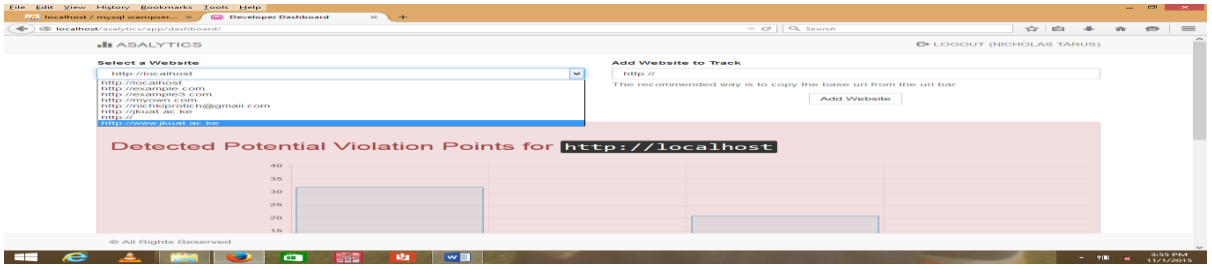


Figure 4.5: Task 5- Selection of the registered website 'http://www.jkuat.ac.ke' from the 'select a website' menu.

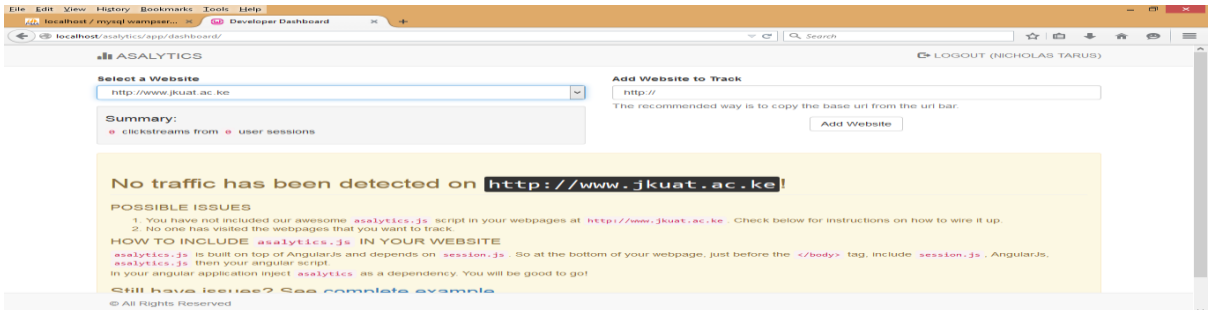


Figure 4.6: Task 6-Evaluation of the registered website for potential usability issues

The paths taken by the developer are not **predefined** and are recorded as **violations** according to the application design. This application was developed from the need to provide developers with information on usability of their web application. There could be hundreds perhaps thousands of users in diverse geographical location and developers will get near real time information on usability of their applications from a remote tool.

4.2 Experimental design results

Table 4.1 Participants time to complete task and errors (violations) made

	Task				Time taken	Expert's time	
Participant	**1**	**2**	**3**	**4**			**Errors(Violations)**
1	1	1	1	0	15	5	4
2	1	1	1	1	7	5	1
3	1	1	1	1	15	5	0
4	1	1	0	0	15	5	6
5	1	1	1	0	15	5	2
6	1	1	1	1	6	5	2
7	1	1	1	0	15	5	2
8	1	1	1	1	7	5	1
9	1	1	0	0	15	5	6
10	1	1	1	1	8	5	0
11	1	1	1	0	15	5	4
12	1	1	1	1	7	5	4
13	1	1	1	0	15	5	7
14	1	1	0	0	8	5	5
15	1	1	1	0	15	5	4

Suitability of the application for use in web applications' usability evaluation was determined by calculating and by using participants' ratings of the metrics identified by (Justin, 2015).

4.2.1 Usability Metrics for Effectiveness

a) Completion Rate

$$Effectiveness = \frac{number\ of\ tasks\ completed\ successfully}{total\ number\ of\ tasks\ undertaken} \times 100$$

$$= \frac{49}{60} \times 100$$

$$= 81.67\%$$

b) Number of Errors

$$Percentage\ error = \frac{number\ of\ errors}{total\ number\ of\ task} \times 100$$

$$= \frac{48}{60} \times 100$$

$$= 80\%$$

4.2.2 Usability Metrics for Efficiency

a) Time Based Efficiency

Each of the fifteen participants' time for completing the task was recorded. The number of participants who successfully completed the task and those who did was also noted. n_{ij} for completed task was assigned '1' and uncompleted task assigned '0'. Time based efficiency was then calculated as follows

$$Time\ Based\ Efficiency = \sum_{j=1}^{R} \sum_{i=1}^{N} \frac{n_{ij}}{t_{ij}} \div NR$$

$$\frac{n_{ij}}{t_{ij}} = 2.4$$

$$\text{NR} = 4 \times 15 = 60$$

Therefore, time based efficiency = $2.4 \div 60 = 0.04$ goals/min

b) Overall Relative Efficiency

It was calculated by placing values into the following equation

$$Overall\ Relative\ Efficiency = \sum_{j=1}^{R}\sum_{i=1}^{N} n_{ij}t_{ij} \div \sum_{j=1}^{R}\sum_{i=1}^{N} t_{ij} \times 100$$

$$\sum_{j=1}^{R}\sum_{i=1}^{N} n_{ij}t_{ij} = 108$$

$$\sum_{j=1}^{R}\sum_{i=1}^{N} t_{ij} = 178$$

Therefore the overall relative efficiency $= \frac{108}{178} \times 100 = 60.67\%$

4.2.3 Usability Metrics for Satisfaction

a) Work load test using NASA TLX workload index

The difficulty of the task was measured by requesting the participants choose by underlining any one of the descriptions in appendix 3. The questionnaires were collected and the results tabulated in table 4.2 below

Table 4.2 Workload Index

			Indices		
Severity	Mental demand	physical demand	Effort required	Performance	Frustration Level
High	4	0	6	9	5
Low	11	15	9	6	10

b) Calculating Usability Score using SUS

System Usability Scale is still valid – it measures what it sets out to measure and has shown itself time and time again to be solid and dependable (Thomas, 2015). Ranking of the 10 template questions (Appendix 2) was received from the participants. The ranking from 1 to 5 was based on their level of agreement. The overall score was determined as follows:

- For each of the odd numbered questions, 1 was subtracted from the score.
- For each of the even numbered questions, the value was subtracted from 5.
- The new values were taken and added up to get the total score. Then, this score was multiplied 2.5.

Table 4.3 SUS Scores

Question	Participants Av score		SUS Score
One	4	(4)-1	3
Two	1	5-(1)	4
Three	3	(2)-1	1
Four	2	5-(2)	3
Five	3	(3)-1	2
Six	3	5-(3)	2
Seven	3	(3)-1	2
Eight	1	5-(1)	4
Nine	4	(4)-1	3
Ten	1	5-(1)	4
Total	**Raw**	**Score**	**28**

The total raw score= 28* 2.5= 70 SUS score/100

Mean of SUS score and related adjective

92-100 = Best

85-91 = Excellent

72-84 = Good

52-71 = Fair

38-51 = Poor

25-37 = Unimaginable

CHAPTER FIVE

IMPLICATION OF RESEARCH FINDINGS

5.1 Main Findings

It is believed that good evaluation techniques are key to ensuring usable web site. This work describes a study carried out to explore the feasibility of using clickstream patterns to obtain information about usability of web application. This developed an asynchronous automatic application for conducting an automated remote web applications' usability and empirically determined its suitability for use in determination of automated usability evaluation.

One of the most important findings of this study is the level of effectiveness it achieved. Its task completion rate was 81.67 % although one should always aim for a completion rate of 100%, the study carried out by Sauri (2011), revealed that the average task completion rate should be not less than 78%. The second metric of determining the effectiveness and suitability of this approach was calculating participants' percentage error.

This is determined by diving the errors with the total number of tasks. The results showed 80% as percentage error, although this value is significantly higher, Sauri (2011) concluded that the average number of errors per task is 0.7, with 2 out of every 3 users making an error and that only 10% of observed tasks are performed without any errors, thus leading to the conclusion that it is perfectly normal for users to make errors when

performing tasks and that since 80% is significantly lower than the expected 90%, the application is more effective. Though a significant difference was identified for the time taken to complete the tasks, they all yielded to the expected results**.**

The novelty of the application increased with significantly higher values for time based and overall efficiencies. The time based efficiency yielded to 0.04 goals/min which was below the expected value of 1 goal/min but this was primarily because the environment was new to the participants. Some participants left evaluation process thus affecting the statistics. The overall relative efficiency yielded to 60.67% which slightly above average. This may be explained by the application's ease of use and its simple interface layout. The application automatically collects clickstreams, generates evaluation data and report information in a simple format which is easily understood. Using the application require less efforts.

The NASA-TLX workload indices suggest that the total workload experienced by users was lower in the application's environment as significant differences were observed for mental demand, physical demand, effort and frustration subscales. This may be due to the lower initial setup required for the application's condition, web application evaluation using this approach requires only that users interact with their browsers and one can view information on usability of their application in a browser.

However, the researcher felt that, effort and frustration subscale values at 6 and 5 respectively were higher (table 4.3). The number of participants who commented that the effort level is high was 40% of the total which is close to half of the total number the

participants. Those who commented that frustration was higher is 33% of the total. The researcher attributed these to the time taken for the test participants to become acquainted with the new environment. The subjective ratings provided by the test participants in the final subjective rating questionnaire revealed significant differences in terms of user satisfaction and ease of use.

SUS score for the application was 70 with a matching adjective of fair. According to Thomas (2015) the average System Usability Scale score is 68. If a website score is under 68, then there are probably serious problems with a website usability which should be addressed. A matching adjective of fair, indicates suitability of the application for use in web usability evaluation. This may be due to automatic generation of information by the application, though it took time for the test participants to become acquainted with the environment.

Although it was researchers feeling that the application achieves a score of 92/100 or higher, explanation for this result could be the inherent in the fact that this approach was a new experience for the participants.

Another interesting piece of information seemed to emerge when the subjective feedback on future preference was analysed. Results showed that participants preferred to use the asynchronous automatic approach to other approaches.

This approach supports web application interface evaluation of geographically distributed, diverse users who are in their native work environments as stated in 2.4.2 of this thesis

work. Given that it automatically generates usability test results, it could therefore be a viable alternative to other remote usability evaluation approaches.

5.2 Limitations

The main objective of conducting this research was to explore various automated remote usability evaluation approaches with a view of developing an application that will collect and provide usability information to developers remotely. Due to limited resources and time, only two companies in Nairobi were requested to participate in the exercise. It is only through the researcher's personal and friends work relationship with the companies that ensured participation and completion of the research.

Most of IT companies in Nairobi have extremely hectic schedule. The small sample size may not accurately represent their views, but the respondents were IT professionals and therefore were more likely to have in-depth knowledge of the subject to provide accurate reflection.

The resources from previous research and studies are very limited. There is very little information available on the subject of dialogue design to carry out automated usability evaluation and the areas dedicated to remote evaluation are even fewer. There are other tools and approaches that are also used in automated remote usability evaluation that are not discussed in this thesis. The aim of this thesis is to provide a general approach of how remote automated usability evaluation is carried out.

It is hoped that the thesis could provide more reference to automated remote usability evaluation approaches and reflections on the use of dialogue designs to improve usability evaluation of web application interfaces. Provide specific information on usability of web application.

CHAPTER SIX

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

6.1 Summary and Conclusions

The purpose of this chapter is to summarize the thesis research and suggest research recommendations for further analysis. A summary of the major results will be described. The sections 6.2 and the 6.3 of this chapter discusses implications of the research and propose recommendations for further research on the asynchronous automatic approach.

This work argues that the application is an easy to use and to install service that will not require any effort on the side of the web application users and neither on side of a developer as the application will automatically collect click-streams, analyse and generate information. This approach is designed and developed to satisfy the requirements specified in its' objectives.

The main purpose of this approach is to assist web designers directly get information on usability of their application and the consequence is, web applications will be created in a more usable manner and thus enhancing users' online experience. One can further appreciate the significance of this application after being aware of the importance of usability on the web application and the need for its evaluation.

Based on the methods surveyed, it is the researchers' opinion that research to further develop this application could result in a promising remote automated techniques. An obvious limitation of this approach is its' inability to determine a specific usability issue. The approach can be extended to determine a specific usability issue by developing click-stream structure that can reference a specific usability problem. This referencing scheme can be further extended by incorporating usability guidelines.

The application can be enhanced by enabling it to store users' usability evaluation results. During future evaluations, this stored information can be referenced to demonstrate the improvements or degradations of the website's level of usability.

Additionally, since the tool relies on heavy backend processes which are likely to be time-consuming, it would be ideal to notify the developers of the evaluation progress by continuously displaying the evaluation results on the dash board. The application can further be interactively set to display the evaluation results of a particular time. Possible future work might take this idea of remote automatic usability evaluation a step further by integrating such a application directly within website development environments.

The lessons learnt in this work is that the dynamic nature of web applications still poses problems to automated usability evaluation. Frequent changes are made from time to time due to user requirements and expectations. New advances e.g. motion user interfaces, ruby on rails, angular js, conversation UI etc. in web development change user expectation too. This application used predefined clickstream paths and therefore with any change in web application design a change in clickstream path in this approach is necessary.

Automated remote usability evaluation is characterised by data collection in a distributed environment and analysis done at a central place. Recent studies have indicated that, automation can be as effective as traditional usability evaluation and therefore, the time and cost of evaluation will be been reduced significantly.

6.2 Recommendations

Based on the results of this study, participants feedback, informal observations by the researcher and the researchers' own experience in setting up the study, a list of recommendations are drafted for the benefit of future usability engineer considering this approach for their usability evaluations.

As with any remote evaluation approach it would be ideal to employ real users in order to collect real representative data. According to Bloomberg, Jordan, Angel, Bailit, Goonan and Straus (1993) it is necessary to adopt a holistic view that behaviours can be understood well in the context in which they occur. Therefore, an experimental test on this approaches need to be in the natural work setting of the user in order for it to be a true representation of the users 'experience.

Additionally, some of the participants in this study commented on how conscious they were in participating in the study and were constantly aware of the evaluator's presence. Hence in order to gather more relevant usability data based on real usage it is necessary for users to participate from their natural work environments. As with any usability studies, tasks that are a representative of real scenarios are used.

The exercise was short so as not to use companies' valuable time but, based on the researchers' conversations with the participants and their feedback in the subjective questionnaire, it was obvious that most participants appreciated the short time span of the study. Hence it is necessary to keep the number of tasks to a minimal reasonable number within which one would be able to get relevant information.

Finally, almost all participants were conducive to the idea of automated remote evaluation and showed interest in the working of the application and looked forward for another participation. Hence it would be ideal to advertise the novelty of this approach to entice web application developers to use it for their application evaluation.

It was important to compare the effectiveness of the asynchronous automatic approach and the identified approaches in the determination of web applications' usability issues. The effectiveness was determined in terms of the amount of time taken to complete a task and the effort required. The same web application (localhost) was used for testing this approach.

Although no cost data was collected in this study, the many cost benefits of the asynchronous automatic approach were apparent. First and foremost the results showed that there was difference in participation time among these approaches**.** There will be no need for the developers to travel and hence, travel costs usually associated with traditional in-lab evaluation are avoided in this approach. Secondly, developers will not take their time off to evaluate their applications. Hence, they will not incur any monetary loss due to time off from work.

6.3 Future research

The study presented in this work is only an initial step. Some necessary suggestions for future studies are outlined here. Studies involving usability expert test participation to address the potential bias of the researcher. Studies involving more participants to ensure the validity and reliability of the results. Studies using geographically dispersed participants in a real time environment, to measure the level of trust between the researcher and the participants. Studies using participants with less technological experience, to determine their level of comfort with the methodology.

Studies where measures like cost effectiveness, development needs, data type collected required by a developer and other facility requirements could be collected and evaluated. Such studies could provide a valuable information based on which one would choose a suitable and financially viable approach for their usability evaluation needs in the future.

A major area for future research can be the application of the asynchronous automatic approach in evaluation of different software applications other than websites.

However, this approach need not be limited to web applications and can be tested for evaluating various other applications that are built on different platforms. In addition, it would also be interesting to explore if usability guidelines that can be adopted for this usability techniques. Results of these studies would help in tweaking this approach and also making it more suitable for automatic remote usability needs.

Hence it is imperative to keep up with the changes in technology and investigate these new technologies in order to enhance asynchronous auto-logging methodology. Future advancements in development of click stream structure for a specific usability issue will aid in designing a more robust application.

REFERENCES

Andersen, J., Giversen, A., Jensen, A. H., Larsen, R. S., Pedersen, T. B., & Skyt, J. (2000, November). Analyzing clickstreams using subsessions. In *Proceedings of the 3rd ACM international workshop on Data warehousing and OLAP* (pp. 25-32). ACM.

Balbo, S. (1996). EMA: Automatic analysis mechanism for the ergonomic evaluation of user interfaces. *Rapport technique, DSIRO Division of Information Report*, *96*, 44.

Batra, S., & Bishu, R. R. (2007, July). Web usability and evaluation: issues and concerns. In *International Conference on Usability and Internationalization* (pp. 243-249). Springer Berlin Heidelberg.

Bevan, N. (2009, August). What is the difference between the purpose of usability and user experience evaluation methods. In *Proceedings of the Workshop UXEM* (Vol. 9, pp. 1-4).

Bloomberg, M. A., Jordan, H. S., Angel, K. O., Bailit, M. H., Goonan, K. J., & Straus, J. H. (1993). Development of clinical indicators for performance measurement and improvement: an HMO/purchaser collaborative effort. *The Joint Commission journal on quality improvement*, *19*(12), 586-599.

Braender, L. M., Kapp, C. M., & Yeras, J. (2009). Using web technology to teach students about their digital world. *Journal of Information Systems Education*, *20*(2), 145.

Buchholz, G., & Propp, S. (2008, July). Towards usability evaluation for smart appliance ensembles. In *International Workshop on Design, Specification, and Verification of Interactive Systems* (pp. 294-299). Springer Berlin Heidelberg.

Carta, T., Paternò, F., & de Santana, V. (2011). Web usability probe: a tool for supporting remote usability evaluation of web sites. *Human-computer interaction–INTERACT 2011*, 349-357.

Chiew, T. K., & Salim, S. S. (2003). Webuse: Website usability evaluation tool. Malaysian Journal of Computer Science, 16(1), 47-57.

Clifton, B. (2012). *Advanced web metrics with Google Analytics*. John Wiley & Sons.

Cugini, J., & Laskowski, S. (2001). Design of a file format for logging website interaction. *NIST Special Publication, 500*, 248.

De Marsico, M., & Levialdi, S. (2004). Evaluating web sites: exploiting user's expectations. *International Journal of Human-Computer Studies*, *60*(3), 381-416.

Dingli, A., & Cassar, S. (2014). An intelligent framework for website usability. *Advances in Human-Computer Interaction*, *2014*, 5.

Dingli, A., & Mifsud, J. (2011). Useful: A framework to mainstream web site usability through automated evaluation. *International Journal of Human Computer Interaction (IJHCI)*, *2*(1), 10.

Dumas, J. S., & Redish, J. (1999). *A practical guide to usability testing*. Intellect books.

Fernandez, A., Insfran, E., & Abrahão, S. (2011). Usability evaluation methods for the web: A systematic mapping study. *Information and Software Technology*, *53*(8), 789-817.

Fidasc, K. (2007). Remote Usability Evaluation methods and Tools: A survey. *Proceedingsof PCI2007, Patras, Greece*.

Hammontree, M., Weiler, P., & Nayak, N. (1994). Remote usability testing. *interactions*, *1*(3), 21-25.

Hasan, L., Morris, A., & Probets, S. (2009). Using Google Analytics to evaluate the usability of e-commerce sites. Human centered design, 697-706.

Hong, J. I., Heer, J., Waterson, S., & Landay, J. A. (2001). WebQuilt: A proxy-based approach to remote web usability testing. *ACM Transactions on Information Systems*, *19*(3), 263-285.

Insfran, E., & Fernandez, A. (2008, September). A systematic review of usability evaluation in web development. In *International Conference on Web Information Systems Engineering* (pp. 81-91). Springer Berlin Heidelberg.

Ivory, M. Y., & Hearst, M. A. (1999). Automated usability evaluation of WIMP and Web interfaces. *Submitted for publication*.

Ivory, M. Y., & Hearst, M. A. (2001). The state of the art in automating usability evaluation of user interfaces. *ACM Computing Surveys (CSUR)*, *33*(4), 470-516.

Jovanovic, N., Kruegel, C., & Kirda, E. (2006, May). Pixy: A static analysis tool for detecting web application vulnerabilities. In *Security and Privacy, 2006 IEEE Symposium on* (pp. 6-pp). IEEE.

Kirakowski, J. (1996). The software usability measurement inventory: background and usage. *Usability evaluation in industry*, 169-178.

Kirakowski, J., & Bevan, N. (1997). MAPI: MUSiC assisted process improvement. In *Human comfort and security of information systems* (pp. 51-59). Springer Berlin Heidelberg.

Kujala, S., Roto, V., Väänänen-Vainio-Mattila, K., Karapanos, E., & Sinnelä, A. (2011). UX Curve: A method for evaluating long-term user experience. *Interacting with Computers*, *23*(5), 473-483.

Le, T., Chaudhuri, S., Chung, J., Thompson, H. J., & Demiris, G. (2014). Tree testing of hierarchical menu structures for health applications. *Journal of biomedical informatics*, *49*, 198-205.

Macleod, M., & Rengger, R. (1993). The development of DRUM: A software tool for video-assisted usability evaluation. *People and Computers*, 293-293.

Madan, A., & Dubey, S. K. (2012). Usability evaluation methods: a literature review. *International Journal of Engineering Science and Technology*, *4*(2), 590-599.

Malý, I., & Slavík, P. (2006, October). Towards visual analysis of usability test logs using task models. In *International Workshop on Task Models and Diagrams for User Interface Design* (pp. 24-38). Springer Berlin Heidelberg.

Monaco, E., Lackey, S., Skawinski, E., Stanley, R., & Young, C. D. (2011). Accessibility and Usability Issues. *E-Governance and Civic Engagement: Factors and Determinants of E-Democracy: Factors and Determinants of E-Democracy*, 128.

Nielsen, J. (2003). Usability 101: Introduction to usability.

Otaiza, R., Rusu, C., & Roncagliolo, S. (2010, February). Evaluating the usability of transactional web sites. In *Advances in Computer-Human Interactions, 2010. ACHI'10. Third International Conference on* (pp. 32-37). IEEE.

Paganelli, L., & Paternò, F. (2003). Tools for remote usability evaluation of Web applications through browser logs and task models. *Behavior Research Methods*, *35*(3), 369-378.

Paternò, F., & Ballardin, G. (2000). RemUSINE: a bridge between empirical and model-based evaluation when evaluators and users are distant. *Interacting with computers*, *13*(2), 229-251.

Paternò, F., & Santoro, C. (2008). Remote usability evaluation: Discussion of a general framework and experiences from research with a specific tool. *Maturing Usability*, 197-221.

Paternò, F., Russino, A., & Santoro, C. (2007, November). Remote evaluation of mobile applications. In *International Workshop on Task Models and Diagrams for User Interface Design* (pp. 155-169). Berlin Heidelberg.

Petrie, H., & Bevan, N. (2009). The evaluation of accessibility, usability, and user experience. In *The universal access handbook* (pp. 1-16). CRC Press.

Propp, S., & Forbrig, P. (2010, October). Vise–a virtual smart environment for usability evaluation. In *International Conference on Human-Centred Software Engineering* (pp. 38-45). Berlin Heidelberg.

Propp, S., Buchholz, G., & Forbrig, P. (2008). Task model-based usability evaluation for smart environments. In *Engineering Interactive Systems* (pp. 29-40). Berlin Heidelberg: Springer.

Runkler, T. A., & Bezdek, J. C. (2003). Web mining with relational clustering. *International Journal of Approximate Reasoning*, *32*(2-3), 217-236.

Sadagopan, N., & Li, J. (2008, April). Characterizing typical and atypical user sessions in clickstreams. In *Proceedings of the 17th international conference on World Wide Web* (pp. 885-894). ACM.

Sauro, J. (10). Essential Usability Metrics.

Senft, S., & Gallegos, F. (2008). *Information technology control and audit*. CRC Press.

Shneiderman, B. (2010). *Designing the user interface: strategies for effective human-computer interaction*. Pearson Education India.

Standard, I. (1998). Ergonomic requirements for office work with visual display terminals (vdts)–part 11: Guidance on usability. ISO Standard 9241-11: 1998. *International Organization for Standardization*.

Stephanidis, C. (Ed.). (2000). *User interfaces for all: concepts, methods, and tools*. CRC Press.

Thomas, N. (2015). How To Use The System Usability Scale (SUS) To Evaluate The Usability Of Your Website.

Valdman, J. (2001). Log file analysis. *Tech. Rep. DCSE/TR-2001-04*.

Varghese, K., & Pfleger, S. (Eds.). (2012). *Human Comfort and Security of Information Systems: Advanced Interfaces for the Information Society* (Vol. 2). Springer Science & Business Media.

Williamson, J. R. (2012). *User experience, performance, and social acceptability: usable multimodal mobile interaction* (Doctoral dissertation, University of Glasgow).

APPENDICES

APPENDIX 1: SUBJECTIVE FEEDBACK QUESTIONNARE

Instructions: Select by putting a tick on the columns indicated 1-5 by following the key provided

Key

1. Strongly disagree
2. Disagree
3. Undecided
4. Agree
5. Strongly agree

		1	2	3	4	5
1	I think that I would like to use this system frequently					
2	I found the system unnecessarily complex.					
3	I thought the system was easy to use.					
4	I think that I would need the support of a technical person to be able to use this system.					
5	I found the various functions in this system were well integrated.					
6	I thought there was too much inconsistency in this system					
7	I would imagine that most people would learn to use this system very quickly.					
8	I found the system very cumbersome to use					
9	I felt very confident using the system.					
10	I needed to learn a lot of things before I could get going with this system.					

APPENDIX 2: NASA-TLX RATING SCALE (HART, 2002)

Instructions: choose by underlining any one from the middle and the last column to the left

Title	Endpoints	Description
Mental Demand	Low/High	How much mental and perceptual activity was required (e.g., thinking, deciding, calculating, remembering, looking, searching, etc.)?Was the task easy or demanding, simple or complex, exacting or forgiving?
Physical Demand	Low/High	How much physical activity was required (e.g., pushing, pulling, turning, controlling, activating, etc.)? Was the task easy or demanding, slow or brisk, slack or strenuous, restful or laborious?
Effort	Low/High	How hard did you have to work (mentally and physically) to accomplish your level of performance?
Performance	Good/Poor	How successful do you think you were in accomplishing the goals of the task set by the experimenter (or yourself)? How satisfied were you with your performance in accomplishing these goals?
Frustration Level	Low/High	How insecure, discouraged, irritated, stressed and annoyed versus secure, gratified, content, relaxed and complacent did you feel during the task?

APPENDIX 3: TASKS

1. Open your browser.
2. Key-in: http://localhost/asalytics/app/signup.php on the address bar.
3. Sign-up by filling the form provided.
4. Login using the email and password you provided in (3) above.
5. Register a website in the ‘add website to track’ menu, then add.
6. Evaluate the website you have added by selecting it from the drop down menu indicated ‘select a website to track’.

APPENDIX 4: PSEUDOCODDE

```
(function(factory){
  'use strict';
  if(typeof define === 'function' && define.amd) {
    define(['angular', 'asalytics'], factory);
  }else if(typeof exports === 'object') {
    module.exports = factory(require('angular'), require('asalytics'));
  }else{
    factory(angular, session);
  }
}(function(angular, session) {
  'use strict';
  angular.module('asalytics', [])
  .run(['$http', function($http) {
      //here, we send the details about the visitors session to our databases
      var actionBaseUrl = "http://localhost/asalytics";
      var visitor = {
        'start': session.original_session.start,
      };
      var click = {
        'requestUrl': session.current_session.url,
        'requestPath': session.current_session.path,
```

```
  'referralUrl': session.current_session.referrer,
  'referralPath': session.current_session.referrer_info.path
};
var logClick = function(theClick){
  $http.post(actionBaseUrl+'/api/clicks/create',JSON.stringify(theClick))
    .success(function(data){
      //alert(JSON.stringify(data));
    })
    .error(function(data){
      console.error(data);
    });
}
$http.post(actionBaseUrl+'/api/websites/get',JSON.stringify(click))
.success(function(data){
    if(data.response.status =='OK' && data.response.website != false){
            click.websiteId = data.response.website.websiteId;
      $http.post(actionBaseUrl+ "/api/visitors/create", JSON.stringify(visitor))
        .success(function(data){
          if(data.response.status=='OK' && data.response.visitor != false){
            var visitorSession = {
              'visitorId': data.response.visitor.visitorId,
              'start':session.current_session.start,
```

```
            'browser':session.browser.browser,

            'os':session.browser.os

          };

          if (session.original_session.visits == 1 || session.original_session.time_since_last_visit > 1800000 ) {

            //create a new session for this user

            $http.post(actionBaseUrl+"/api/sessions/create", JSON.stringify(visitorSession))

              .success(function(data){

                if(data.response.status=='OK'){

                  //alert(JSON.stringify(data));

                  visitorSession = data.response.session;

                  click.sessionId = data.response.session.sessionId;

                        logClick(click);

                }

                    })

              .error(function(data){

                // alert(data);

                console.error(data);

              });

          }else{

            //fetch the existing session

            $http.post(actionBaseUrl+'/api/sessions/get', JSON.stringify(visitorSession))
```

```
        .success(function(data){
                        if(data.response.status=='OK'){
          //alert(JSON.stringify(data));
          visitorSession = data.response.session;
          click.sessionId = data.response.session.sessionId;
                        logClick(click);
         }
        })
        .error(function(data){
          //alert(data);
          console.error(data);
        })
      }
    }
  })
  .error(function(data){
    // alert(data);
    console.error(data);
    //to do
  })
}else{
 // alert(JSON.stringify(data));
```

```
            console.error(data);
            return;
          }
      })
      .error(function(data){
        // alert(data);
        console.error(data);
      });
  }])
}));
```