Thesis no: MSSE-2015-03

Evaluating Efficiency Quality Attribute in Open Source Web browsers

Anil Reddy Tigulla
Satya Srinivas Kalidasu

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona Sweden

This thesis is submitted to the Faculty of Computing at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Software Engineering. The thesis is equivalent to 20 weeks of full time studies.

Contact Information:
Author(s):
Satya Srinivas Kalidasu
E-mail: sakc11@student.bth.se
Anil Reddy Tigulla
E-mail: antj11@student.bth.se

University advisor(s):
Dr. Wasif Afzal
Faculty of Computing, BTH

Faculty of Computing
Blekinge Institute of Technology
SE-371 79 Karlskrona, Sweden

Internet : www.bth.se
Phone : +46 455 38 50 00
Fax : +46 455 38 50 57

ABSTRACT

Context: Now a day's end users are using different types of computer applications like web browsers, data processing tools like MS office, notepad etc., to do their day-to-day works. In the real world scenario, the usage of Open Source Software (OSS) products by both industrial people and end users are gradually increasing. The success of any OSS products depends on its quality standards. 'Efficiency' is one of the key quality factor, which portray the standards of product and it is observed that this factor is given little importance during its development. Therefore our research context lies within evaluating the efficiency quality attribute in OSS web browsers.

Objectives: As discussed earlier the context of this research lies in evaluating the efficiency of OSS web browsers, the initial objective was to identify the available efficiency measures from the current literature and observe which type of measures are suitable for web browsers. Then our next objective is to compute values for the identified efficiency measures by considering a set of predefined web browsers from all the categories. Later we proposed Efficiency Baseline Criteria (EBC) and based on this criterion and experiment results obtained, the efficiency of OSS web browsers had been evaluated. Therefore the main objective of conducting this research is to formulate EBC guidelines, which can be later used by OSS developers to test their web browsers and ensure that all the quality standards are strictly adhered during the development of OSS products.

Methods: Initially Literature Review (LR) was conducted in order to identify all the related efficiency quality attributes and also observe the sub-attribute functionalities, that are useful while measuring efficiency values of web browsers. Methods and procedures which are discussed in this LR are used as input for identifying efficiency measures that are related to web browsers. Later an experiment was performed in order to calculate efficiency values for CSS & proprietary set of web browsers (i.e. Case A) and OSS web browsers (i.e. Case B) by using different tools and procedures. Authors themselves had calculated efficiency values for both Case A and Case B web browsers. Based on the results of Case A web browsers, EBC was proposed and finally an statistical analysis (i.e. Mann Whitney U-test) is performed in order to evaluate the hypothesis which was formulated in experiment section.

Results: From the LR study, it is observed that efficiency quality attribute is classified into two main categories (i.e. Time Behavior and Resource Utilization). Further under the category of Time behavior a total of 3 attributes were identified (i.e. Response time, Throughput and Turnaround time). From the results of LR, we had also observed the measuring process of each attribute for different web browsers. Later an experiment was performed on two different sets of web browsers (i.e. Case A and Case B web browsers). Based on the LR results, only 3 efficiency attributes (i.e. response time, memory utilization and throughput) were identified which are more suitable to the case of web browsers. These 3 efficiency attributes are further classified into 10 sub-categories. Efficiency values are calculated to both Case A and B for these 10 identified scenarios. Later from Case A results EBC values are generated. Finally hypothesis testing was done by initially performing K-S test and results suggest choosing non-parametric test (i.e. Mann Whitney U-test). Later Mann Whitney U-test was performed for all the scenarios and the normalized Z scores are more than 1.96, further suggested rejecting null hypothesis for all the 10 scenarios. Also EBC values are compared with Case B results and these also suggest us that efficiency standard of OSS web browsers are not equivalent to Case A web browsers.

Conclusions: Based on quantitative results, we conclude that efficiency standards of OSS web browsers are not equivalent, when compared to Case A web browsers and the efficiency standards are not adhered during development process. Hence OSS developers should focus on implementing efficiency standards during the development stages itself in order to increase the quality of the end products. The major contribution from the two researchers to this area of research is "Efficiency Baseline Criteria". The proposed EBC values are useful for OSS developers to test the efficiency standards of their web browser and also help them to analyze their shortcomings. As a result appropriate preventive measures can be planned in advance.

Keywords: Efficiency Quality attribute, Open Source Software (OSS), Efficiency Baseline Criteria (EBC), Efficiency evaluation, Web browsers.

ACKNOWLEDGMENT

First, we would like to thank our parents for their kind love, patience and support throughout our thesis work. Then we would like to thank our University advisor Dr. Wasif Afzal for his kind patience and innovative ideas that helped us a lot during our thesis work. The way our advisor responded to e-mails is really appreciable. His guidance, feedback and support throughout our thesis have been immensely valuable.

We are also thankful to Mr. Mahesh Chandra Yelleswarapu, who stood instrumental to our thesis work and helped us a lot in formulating basic ideas during initial stages. Next we would like to thank Dr. Tony Gorschek and Dr. Jurgen Bostler, entire academic staff and library staff of Blekinge Tekniska Hogskola, for their kind guidance and material support. Finally we would like to thank our thesis opponents Mr. M.V.S Shilesh and Mr. Amrit Pandey for their valuable comments and suggestions on our thesis draft.

- Satya Srinivas Kalidasu.
- Anil Reddy Tigulla.

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Introduction to Open Source Software (OSS)

Generally Open Source Software's (OSS) are developed, tested and improved through public collaboration and distributed with an idea that must be shared among others. OSS software licenses (like GPL) provide users with 4 essential freedoms [1]:

- User can run the OSS program for any purpose.
- User can study the working of the OSS program and has privilege to modify a program according to his requirements.
- User can also redistribute the copies of OSS program at free of cost.
- In order to improve the program end user can release improved and modified versions of the product.

Some of the benefits of OSS over proprietary software are mentioned below [1]:

- Large pool of globally dispersed, highly talented and motivated professionals can be involved in OSS development.
- The release time of OSS products are very high.
- OSS project developers can reuse the existing code; as a result time taken for the development is rapidly decreased and no need to work from foundation part.
- The development cost of an OSS project is low compared to other developments.
- Less amount of project management resources are required while developing OSS projects.
- End users will show interest in using OSS as they are available free in the market and also the user has an option to modify the code of the software according to their needs.

According to Henrik et al. [2] discussed the following issues regarding OSS development:

- The outcome of the OSS development process will depend on the skills of participating developers.
- In the development process of OSS products there is proper standardized process. Developer can choose his own way of working to develop the product.
- Accurate design and planning are rarely done.
- The development process is in ad-hoc style.
- Quality assurance techniques are mainly ignored.

1.2 Problem definition and its importance

Samoladas et al. in [3] discussed the future scope for all the quality attributes considered in ISO 9126 model. Until now qualitative and quantitative research work is done for some of the quality attributes like functionality, reliability, usability, maintainability and portability. But *efficiency* quality attribute has not been studied until now [3] and according to the authors, research can be done on understanding, how the efficiency quality attribute is assessed in all types of software products (i.e. specifically in OSS products) [3][27][28].

In general, quality standards of OSS products are not equivalent, when compared to CSS and Proprietary web browsers. The efficiency factor can be considered as a key quality attribute that needs to be resolved in OSS products, in order to improve its performance and also to satisfy the end user. Roets et al. in [1], discussed and compared different types of SDLC models proposed by various researchers. According to [1], there is no specific SDLC model followed, during the development of OSS products, as a result the quality standards of the product are reduced. Therefore in order to improve the quality standards in OSS products, there is a need to study efficiency quality attribute.

It is important for the OSSD community to maintain high efficiency levels while developing their products, in order to achieve better quality standards. In this study, we had chosen web browsers as our case and studied the efficiency standards by considering CSS, OSS and proprietary web browsers. These results helped us to propose an Efficiency Baseline Criteria (EBC), which can be used by OSSD community to evaluate and improve the quality standards of OSS web browsers.

1.3 Thesis outline

The whole thesis is divided in to 7 chapters and each chapter contains the following information:

Chapter 1: (Introduction): This section provides introduction to OSS products, advantages and limitations in development process. It also defines the problem definition, scope and importance of writing this thesis. At the end of this section glossary of terms used in thesis are presented.

Chapter 2: (Background and Related work): In the background section a crisp overview of Open Source Software Development (OSSD) life cycle models proposed by various authors are discussed and later an overview of ISO 9126 model is provided. In the related work section, an overview of frameworks provided by various researchers in the area, “Evaluation of OSS products” are discussed.

Chapter 3: (Research Agenda): This section of thesis document contains aim, objectives, research questions (RQ’s) that are needed for conducting this research and an overview of mapping between objectives and RQ’s are provided.

Chapter 4: (Literature Review): In this section efficiency quality attribute and their sub-attributes are briefly discussed. This chapter focuses on studying the efficiency measures that are related to various software products and also discussed the functioning of sub-attributes that are considered for our case. This section also identifies the efficiency attributes that are suitable for the case of web browsers.

Chapter 5: (Experiment): This section of master thesis contains the experiment part, which was conducted on the guidelines provided by Wohlin et al. [7]. In this section experiment definition, planning, design, execution and results are discussed lucidly.

Chapter 6: (Discussion): In this section the results derived from experiment process are discussed and from that analysis, Efficiency Baseline Criteria (EBC) is proposed based on the results obtained for Case A web browsers.

Chapter 7: (Evaluating Efficiency of OSS Web browsers): In this section, we have performed hypothesis testing by selecting appropriate statistical analysis methodology and a comparative analysis between EBC results and Case B web browsers are also presented.

Chapter 8: (Validity threats): In this section, all the valid threats and their possible mitigation strategies that occur while performing our research work are discussed lucidly.

Chapter 9: (Conclusion): This chapter provides final conclusion part to our thesis by providing answers to our research questions.

Chapter 10: (Future work): This section describes future work that can be done in this area of research.

1.4 Glossary of terms

Notations	Meanings
OSS	Open Source Software
EBC	Efficiency Baseline Criteria
GPL	General Public License
TB	Time Behavior
OSSD	Open Source Software Development
RU	Resource Utilization
RT	Response time
TH	Throughput
TT	Turnaround time
MRU	Memory Resource Utilization
IOU	Input or Output resource Utilization
TRU	Transmission Resource Utilization

2 BACKGROUND AND RELATED WORK

Initially in this section, we want to analyze the various phases, concepts and models that are related to development of OSS products. This initial work is done in order to observe whether quality aspects are considered in OSS development lifecycle.

According to Vixie [4], OSS development (OSSD) will contain all the elements of traditional SDLC like analysis, design, implementation and support. Classic OSS projects such as BSD, BIND and Send Mail are some products, which follow the above phases in its development.

According to Schweik and Semenov [5], OSSD can be divided into 3 phase's i.e.

- Project initiation
- Going 'open'
- Project growth, stability (or) decline.

Jorgensen N., in the paper [6] proposed an OSSD model based on FreeBSD projects. The incremental change life cycle model proposed in [6] has six stages. They are

Stage 1: Code.

Stage 2: Review.

Stage 3: Pre-commit test.

Stage 4: Development release.

Stage 5: Parallel debugging.

Stage 6: Production release.

Generally quality is the key aspect of any product, for example manufacturing a bus or any automobile has certain level of quality. Similarly OSS products should possess some quality standards. By means of quality standards, we can judge the performance of the software (i.e. good or bad). Identifying the required quality attribute, which is useful for assessing the OSS products, is our primary focus at this stage. For this we performed an initial literature review on different quality attributes. There exist several models for software quality, which suggest various ways to bring together different quality attributes. Each one of these models tries to aggregate several attribute features to judge the software quality. One of the models is ISO 9126 [8][9][19]. It is a hierarchical model consists of 6 major attributes contributing to software quality. They are

- Functionality.
- Reliability.
- Usability.
- Maintainability.
- Portability.
- Efficiency

According to Feller et al. in [10], discussed a framework (i.e. for analyzing the OSS development approach) which is derived from 2 previous frameworks:

- Zachman framework (i.e. contains categories like what, how, where, who, when and why).
- Check land CATWOE (Client, Actor, Transformation, *Weltanschauung*, Owner and Environment) technique.

The brief overview of the framework (for analyzing the OSS development approach) is discussed in the below table [10]. In the section 6.1, we had proposed Efficiency Baseline Criteria, which will be an answer to the below questions proposed by Feller [10].

Table 1 - Feller framework on OSS development approach

Framework step	Focuses on OSS concepts
What (Transformation)	• What defines a software project as OSS? • What types of projects tend to be OSS?
Why (World View)	• What are the technological motivations for OSS development? • What are the economic motivations for OSS development? • What are the socio-political motivations for OSS development?
When and Where (Environment)	• What are the temporal dimensions of OSS development? • What are the spatial/geographic dimensions of OSS development?
How	• How is the OSS development process organized? • What tools are used to support the OSS model?
Who (Client, Actor, Owner)	• What are the characteristics of the individual developers contributing to OSS projects? • What are the characteristics of the companies distributing OSS products? • What are the characteristics of the users of OSS products?

According to Alfonso et al. in [11], discussed the main technical arguments provided by open source advocates. In that 4th argument states as follows, "*Open source software is more reliable*" [11], that means that code will be visible to the public, in case if any errors are identified in the code, then it will be easily fixed by any one of the participant in public, as a result this ensures a trust feature for any software. However in proprietary software the company people will take some time to fix the exact error, but it is more reliable than open source software because of its quality standard tools used by proprietary companies to fix these errors accurately [32].

According to Brian Fitzgerald in [12] discussed the tensions and paradoxes within OSS. They are:

- Cathedral vs. Bazaar development approach
- Collectivist vs. Individualist
- OSS vs. Free Software Foundation (FSF)
- Is OSS a Paradigm Shift in Software Industry?
- Is OSS comes under High Quality Software?

In the research article [13], Selvi et al. compared Performance Analysis of Proprietary (i.e. JSP, Javascript and ASP) and Non-Proprietary Software (i.e. Perl and PHP) for a website application by using OpenSTA performance tool and measured performance in terms of elapsed time, timer values and response of software. Finally, it is concluded that OSS applications elapsed time is very less with respect to timer values when compare to Proprietary.

Yunwen Ye et al. [14] discussed the roles in OSS communities; they are Project Leader, Core Member, Active Developer, Peripheral Developer, Bug Fixer, Bug Reporter, Reader and Passive User. Lee et al. [15] discussed about the influence of quality on OSS products in the form of 7 hypotheses. They are:

- *H1: Software quality has a positive effect on OSS use.*
- *H2: Community service quality has a positive effect on OSS use.*
- *H3: Software quality has a positive effect on user satisfaction.*
- *H4: Community service quality has a positive effect on user satisfaction.*
- *H5: User satisfaction has a positive effect on OSS use.*
- *H6: User satisfaction has a positive effect on individual net benefits.*
- *H7: OSS use has a positive effect on individual net benefits.*

Atieh Khanjani et al. [16], explained the concepts of open source development. They are:

- Distributed software.
- Free software.
- Available source code.
- Communicate through internet.
- Developers are users.
- Unpaid and large amount volunteers.

Similarly the quality assurance in open source software depends on 2 factors (i.e. code review and data testing) [16]. The process and procedure for quality assurance in open source software is explained 6 steps [16]:

- Requirement Definition and Design.
- Large Development Community.
- Testing.
- Maintenance.
- Documentation.
- Security Issues.

Mark Aberdour [18], discussed OSS quality management, in that the onion model was discussed, which consists of few number of core developers, increasing number of contributing developers, bug reporters and users. This onion model can be used by OSS development community. Similarly, Davide Taibi et al. [17] proposed an Open BQR (i.e. a framework for the evaluation of OSS), which address following 4 problems [17]:

- Only few aspects of OSS development are been focused in the existing methods.

- Some methods proceed to evaluating indicators before they are weighted, so some factors may be measured or assessed even if they are later given a very low weight or even a null one. This results in unnecessary waste of time and effort.
- Even though the source code of the OSS product is available, the internal and external product qualities are not adequately discussed in any of the OSS evaluation methods.
- The dependence of the users of OSS is not adequately assessed, especially the availability of support over time and the cost of proprietary modules developed by third parties.

From the above research studies it is evident that the quality assurance factors are not given much importance in software development lifecycle of OSS. So in our thesis, we assess the quality attributes in OSS products by considering an experiment and finally suggest a set of guidelines for the OSSD community. These guidelines can be directly used in development of OSS products. Now our primary challenge is to identify the important quality attribute that is required for assessment of OSS products [29][30][31], which was clearly discussed in chapter 4.

3 RESEARCH AGENDA

3.1 Aim and Objectives

Our aim is to evaluate efficiency quality attribute in OSS Web browsers by comparison and finally propose an efficiency baseline criteria, which will be helpful to the OSSD community in order to maintain efficiency standards in OSS web browsers.

To achieve our aim, we had considered the following objectives:

- To identify the importance and measures of efficiency quality attribute in software products.
- To check and analyse whether the above identified measures of efficiency are applicable to our context of web browsers.
- To calculate values to the identified measures for Case A web browsers (i.e. Google chrome, Internet Explorer, Mozilla Firefox and Opera) and to derive an efficiency baseline criterion based on the obtained values.
- To suggest an efficiency baseline criteria to OSSD community, which will be helpful for them in order to estimate and improve the efficiency of OSS web browsers.
- To calculate values to the identified measures for OSS web browsers (i.e. Midori, Seamonkey, Qupzilla and Arora).
- To compare the efficiency values of EBC with OSS web browser and perform hypothesis testing, in order to evaluate the efficiency quality attribute in OSS web browsers.

3.2 Research Questions

RQ 1: How efficiency quality attribute is measured in software products and which efficiency attributes are suitable for the case of web browsers?

This research will focus on understanding and identifying the efficiency quality attributes in software products. Further we will also investigate each efficiency quality attribute in detail by conducting a literature review. Later efficiency quality attributes that are suitable for the case of web browsers are identified in this study. The motivation for selecting efficiency quality attribute is that there are few studies stating the need to investigate efficiency standards in software products [3][17][30].

RQ 2: What is the procedure to calculate the efficiency values of web browsers and what are the values for those attributes?

This research initially defines step by step procedures for calculation of values for each efficiency quality attribute identified through literature review. Further efficiency values are calculated for the identified attributes by using above procedures on selected set of web browsers. In order to calculate efficiency values for web browsers, an experiment was organized. For comparison purpose, we had categorized web browsers in to 2 cases and further supplemented with two sub research questions.

RQ 2.1: What are the efficiency values for major set (i.e. Case A) of web browsers and how to baseline these observed efficiency values?

In this research efficiency values are calculated for Case A web browsers (i.e. Mozilla, Google Chrome, IE and Opera) by following the predefined procedures. Later these results are further analysed and base lined. As a part of this section, we had proposed EBC criteria based on the results obtained from Case A web browsers. The motivation for proposing EBC is to provide an efficiency standard criterion for OSSD community that can be used to maintain quality standards during the development and maintenance process of OSS web browsers.

RQ 2.2: What are the efficiency values for OSS Web browsers?

In this research efficiency values are calculated for Case B web browsers (i.e. Midori, Qupzilla, Arora and Seamonkey) by following the same predefined procedures. Later these efficiency values are analysed in the results section.

RQ 3: How to evaluate efficiency quality attributes in OSS Web browsers with the help of experiment results and proposed Efficiency Baseline Criteria (EBC)?

In this research, we had compared the efficiency values of Case B web browsers with both experiment results and proposed EBC values and analyzed the performance of OSS web browsers. This analysis helps the OSSD community to analyze the efficiency of OSS web browsers when compared to proprietary web browsers.

3.3 Mapping objectives to RQ’s

Table 2 - Mapping of research questions, objectives and research methodology

Objectives	Research Questions	Research Methodology
Ob1	RQ1	Literature Review (LR)
Ob2	RQ2	Experiment
Ob3	RQ2 (RQ 2.1)	Experiment
Ob4	RQ2 (RQ 2.1)	Experiment
Ob5	RQ2 (RQ 2.2)	Experiment
Ob6	RQ3	Experiment - Analysis

4 LITERATURE REVIEW

The main aim of this literature review was to study the efficiency quality in software products especially in OSS products and identify efficiency attributes that are suitable to the scenario of web browsers. The motivation for selecting efficiency quality attribute is that there are few studies stating the need to investigate efficiency standards in software products [3][17][30]. For addressing our concerns and to achieve the aim of literature review, initially a RQ 1 was formulated. By answering this research question, we would be able to identify measures related to efficiency quality attribute and also identify sub-attributes related to software products. The research question formulated in order to carry our research is as follows.

RQ1: How efficiency quality attribute is measured in software products and which efficiency attributes are suitable for the case of web browsers?

To carry out this research initially a search string was formulated and searched in databases (i.e. IEEE, Inspec, Google Scholar, EI Compendex, Scopus). The whole process is defined below:

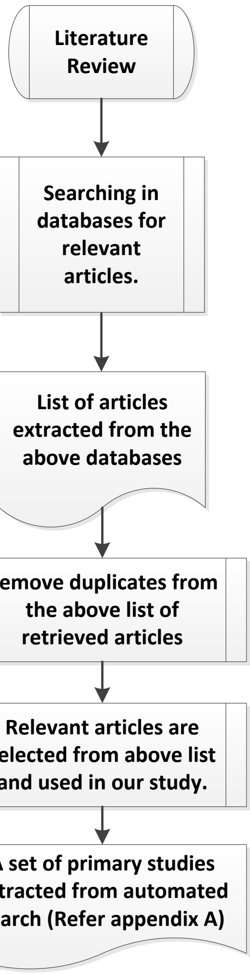

Figure 1 - Selection criteria for research articles

We had totally extracted 68 primary study articles. Based on type of the article criteria, we had categorized primary studies into 3 categories. Out of which 29 (i.e. 43%) are Journal articles, 31 (i.e. 46%) are Conference articles and 8 (i.e. 11%) are other articles (i.e. books, workshop etc.) which are represented in the below pie diagram. These results show that 60 out of 68 articles (i.e. 88% of total articles) are well peer-reviewed.

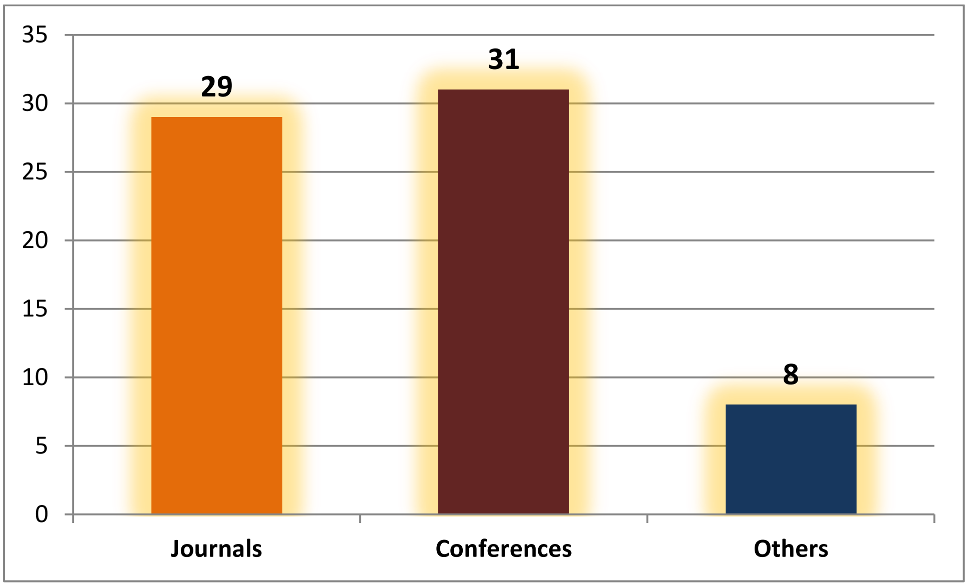

Figure 2 - Distribution of primary sources based on source type

Based on the articles retrieved (i.e. list of articles are presented in Appendix A), we had analyzed and presented the results in three sections. Section 4.1 gives a brief overview of efficiency quality attribute in general and section 4.2 discuss the quantification of efficiency quality attributes & sub attributes and its influence in software products especially in web browsers. Later in section 4.3, we define efficiency quality attributes that are suitable for the study of web browsers.

4.1 Efficiency Quality Attribute

Initially we understood the term “Software Efficiency” by studying literature. In this section a brief overview of various definitions defined for efficiency quality attribute were discussed lucidly. According to Padayachee et al. [95], the term efficiency is defined as “The capability of the software product to provide desired performance, relative to the amount of resources used, under stated conditions”. Shing et al. [20] defined as “The degree to which the software makes optimal use of system resources”. Alvaro et al. [80] defined it as “The ability of a software component to provide appropriate performance relative to the amount of resources used”. McCall [21] defined it as “Relative extent to which a resource is utilized (i.e. storage space, processing time, communication time etc.)”. Until now we had discussed some definitions related to software efficiency. Later we identified different characteristics and sub-characteristics that are related to efficiency quality attribute. According to the research articles [70] [98], Time behavior and Resource utilization are two important characteristics to define efficiency of any system and these are discussed in the below table.

Table 3 - Classification of Efficiency quality attribute

Quality Attribute	Characteristic	Sub Characteristic
Software Efficiency	**Time Behavior**	• Turnaround time. • Throughput. • Response time.
	Resource Utilization	• Memory Resource Utilization. • Input and Output Resource Utilization. • Transmission Resource Utilization.

In general, efficiency quality attribute is used to depict the performance levels of any product. In the current market trends, it is considered as one of the important measure that is to be constantly monitored during the SDLC phases [108]. Efficiency attribute mainly expresses the ability of a component to provide appropriate performance, relative to the total number of resources used [62][80][95]. Efficiency can be measured during runtime by using a sampling method, where the process starts with recording the start and the end times of an activity [70]. Efficiency quality attribute can be considered as important characteristic feature, while measuring the quality of the software products [62][110]. Therefore efficiency quality attribute can be used to provide answer to the question like *"How quickly does the system respond for a given task?"* [94].

In the current literature, it is observed that efficiency quality attributes are further classified into various sub attributes, which were discussed lucidly in every quality model. For example in McCall's quality model [108][110], which is known as the first quality model proposed in 1977, categorized efficiency quality attribute in to Execution efficiency and Storage efficiency sub attributes, which are measured subjectively on a scale ranging from 0 (low) to 10 (high) [108].

Similarly in Boehm's quality model [108][110] efficiency quality attribute is been categorized in to 3 sub attributes like Accountability, Device efficiency and Accessibility. Boehm's and McCall's quality models are mostly useful when a bottom to top approach is being followed (i.e. Software quality measures can be effectively defined but there will be some difficulties while specifying the quality requirements) [108]. Finally in ISO/IEC 9126 quality model proposed in 1991 by International Organization for Standardization, categorizes the software quality characteristic feature in to 2 sub-attributes, they are Internal and External quality characteristics. These two characteristics are derived basing on the inspiration works from McCall and Boehm's quality models. Among them efficiency quality attribute is defined under external quality characteristic [109]. According to the ISO/IEC 9126 standards efficiency quality attribute is defined as *"capability of the system to provide appropriate performance, relative to the amount of resources used under the stated* *conditions* [109]*"*. According to the article [108] Suryn et al., had concluded that ISO/IEC 9126 model is a better quality model when compared to the other two models (i.e. Boehm and McCall) by illustrating it with two main reasons [108]. They are

Firstly, it is found that both McCall and Boehm quality models are mainly focused on the "product perspective" of the quality, whereas ISO/IEC 9126 supports all the perspectives of quality. Secondly, these two model frameworks (i.e. Boehm and McCall) support only bottom-up approach which is not suitable for the domain of SQM. While the ISO/IEC 9126 framework supports both top-down and bottom-up approaches.

Therefore from the above facts it is concluded that ISO/IEC 9126 model is superior to other two models and the framework of ISO/IEC 9126 is much more suitable for the domain of Software Quality Engineering. From the above discussion, we had decided to use sub-attributes mentioned in ISO/IEC 9126 standards for quantifying efficiency quality attribute in software products.

According to a research study, which is jointly done by Software Engineering Institute (SEI) and Architecture Trade-off Analysis Method (ATAM), their experiment results [56] concluded that the efficiency quality attribute has 13.6% of significance, when compared to other quality attributes. This value shows the number of times the concept is expressed as a top level quality attribute divided by total number of scenarios. In their opinion, efficiency quality attribute plays a pivotal role in any software development process. They had totally considered 20 attributes out of 49 available [59] and calculated significance percentage (%) values. For example attributes like performance response time has a significance value of 3.6%, performance of latency has 3.2%, performance of throughput has 2.1% and performance of resource utilization has 1.9% of significance ratios. By these significance values, one can understand the impact and importance of efficiency quality attribute and its sub-attributes in software products [56][106].

Becker, while discussing performance related metrics in the ISO 9126 standards [98], stated that ISO 9126 is the better standard model for describing the quality of software systems. According to the ISO 9126 standards efficiency quality attribute is further divided into two sub-attributes [110]. They are Time behavior and resource behavior. The whole classification of efficiency quality attribute in ISO 9126 is depicted in the below figure [64][98][105][106][110].

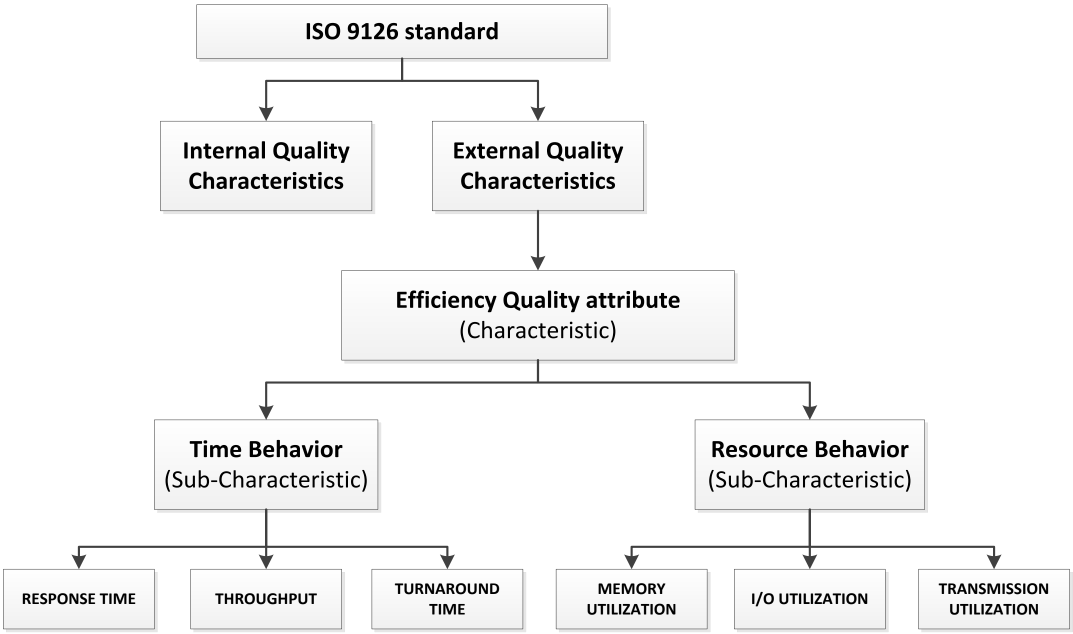

Figure 3 - Classification of efficiency quality attribute based on ISO 9126 standard

From the above figure, it is clearly evident that according to ISO 9126 standard [98], "efficiency quality attribute" is further classified into two sub-characteristics, they are "time behavior" and "resource behavior". Further "time behavior" is categorized into "response time", "throughput" and "turnaround time" and similarly "resource behavior" is divided into "memory utilization", "I/O utilization" and "transmission utilization".

Based on the above results, we had categorized retrieved primary articles in to time behavior and resource behavior. 68 primary articles were grouped into 39 articles which covers only time behavior and 10 articles covers only resource utilization and 19 common articles covers both time behavior and resource utilization, which is shown in below figure. It shows around 57% of our primary sources will cover time behavior characteristics, 15% will cover resource behavior characteristics and 28% are common articles which describe both behaviors.

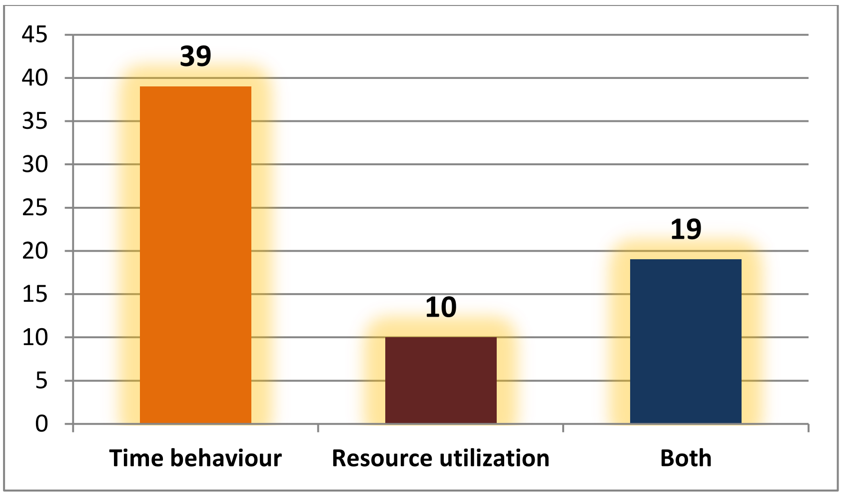

Figure 4 - Classification of primary sources based on the efficiency sub-attributes

4.2 Quantification of Efficiency Quality Attribute

In this section initially primary articles related to efficiency measures are represented based on sub attributes classification as defined in ISO 9126 model and presented in the form of bar graph. The x-axis shows attribute wise classification and y-axis shows the counts of articles retrieved. Each bar weight shows the count of articles obtained for each measure, which signifies the selected primary article in specified years. The figure is not mutual exclusive, why because for example single article may be specifying about more than one attribute. Below figure represents the bar graph of our study.

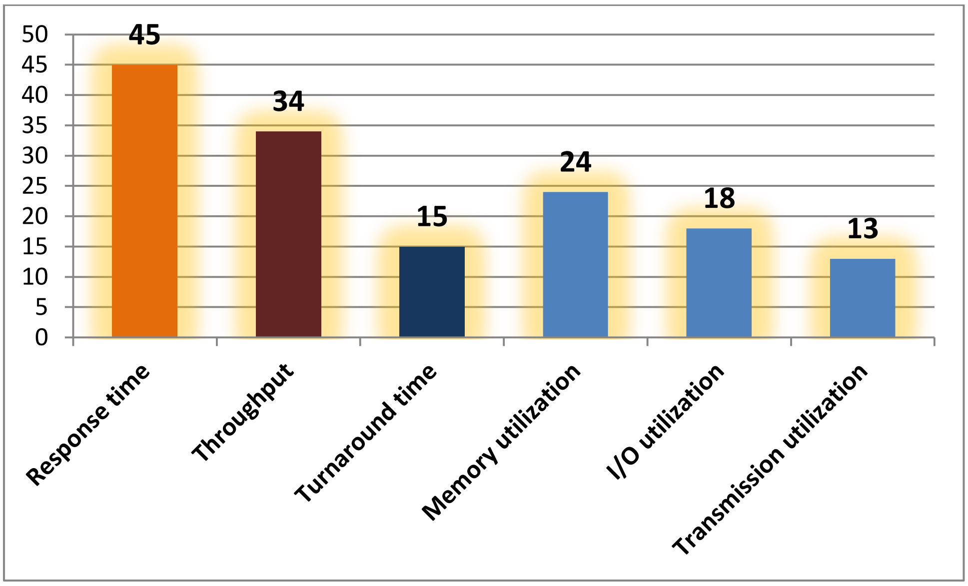

Figure 5 - Bar graph showing articles count - Attribute wise

Now we will analyze efficiency attributes and its sub attributes as discussed in ISO 9126 model by initially discussing the definition and then defines its formulae used to calculate the respective efficiency value and finally its usage in real time context scenarios.

Time behavior (TB)

According to [70], time behavior is stated as, *"The capability of the software product to provide appropriate response and processing times and throughput rates when performing its functions under stated conditions"* [70]. This quality attribute can be used to provide answer to the question like *"How quickly does the system respond for a given task?"* [94][95].

Resource Utilization (RU)

According to [70], resource utilization is stated as, *"The capability of the software product to use appropriate amounts and types of resources when the software performs its function under stated conditions"* [70][74][87]. This quality attribute can be used to provide answer to the question like *"Does the provided system utilize resources efficiently?"* [65][94][95]. Generally metrics used to measure these resource utilization attributes are network bandwidth, network usage, size of available memory, memory usage, processing power and processing usage [86][87].

4.2.1 Response time

4.2.1.1 Conceptual definitions and formulae

Response time can be stated in the following ways:

- *"A Response time attribute measure the time taken since a request is received until a response has been sent"* [80][81].
- *"A response time can be measured as the total time consumed for completing a specified task. It can be recorded, as the time span between the start of the task and its completion"* [98].
- In the case of virtual machines, response time can be defined as *"the response time is the elapsed time between a request to read a block in a front end driver and its response in the interrupt handler of the driver"* [50].
- *"Response time of web applications can be defined as, the time between submission of the request and the time when the client finishes receiving the response"* [58][73].
- Actual Response Time (ART) can be defined as *"time spent between requests originated from client, reaches the web service server through application server, for which response being delivered to Application server and acknowledgement reaches web service server"* [54].
- *"Response time measures time between submitting a job and receiving its first response"* [63].
- *"Response time measurement indicates how long it takes to send a request and receive a reply over a network"* [72].
- Response time can be defined as, *"The response time includes both the queuing time in the gateway and the execution time in the server nodes"* [117].
- *"Response time is the elapsed time between the start and the termination of a service"* [75].
- In SOA applications response time is defined as, *"Duration of the time it takes from when a request first arrives at the datacentre and to the time the response for the request leaves the datacentre. The response time includes the queuing and execution delays at each component of the application"* [102].
- *"Response time is the time costed in the process, which is the length from the beginning that customer makes the request to the end that the system responses to the customer"* [74].

Response time is the total time taken by a system or service to respond the requests specified by different users [62]. One of the functions of load intensity is response time and can be measured in terms of arrival rates [61]. Response time can be measured using the formula

Average Response time (ART) = Wait time + Server execution time.

{OR}

Average response time = (RT of iteration 1+.....+iteration n)/ Number of iterations [72].

{OR}

Mean execution time = (Sum of processing time of all processes)/
(Total no. of processes) [96]**.**

In real time applications some of the external factors that are needed to be considered for response time are [111]:

- **Priority** - Response time values will be low for high priority and similarly converse is also applicable in this situation.
- **Dependency** - In a real time system if the result of a case B is dependent on case A then the response time for case B depends on the response time of case A.
- **Timing requirement** - It defines the expected response time after a request is initiated. In a real world scenario the user request may be initiated after a specific period of time. Therefore less waiting time reflects high response rate.

For any end user long system response time may cause frustration and as a result it will end up with lower customer satisfaction. This may affect the performance of artifact, resulting poor productivity [115]. For example in the past, response times of e-commerce systems are very high due to enormous size of graphics and unwanted applications running in the background, consuming more memory and bandwidth speeds. Therefore web developers are considering response time as a critical issue and emphasis is given to reduce the response time of e-commerce applications [58][115]. From a survey results done by Jupiter media metrix [53], it is stated that end users feel inconvenient, when the response time of a web page exceeds 8 seconds and end user's quit the web page if it takes more than 10 sec to load. Therefore for web designers minimal usage of keywords, images and flash objects helps them to achieve better response time for their designed web pages. In [53] Palmer et al. proposed some tips in order to decrease the loading time of a web page, they are developers must define the suitable server requirements and determine bandwidth and connection speeds. It always better to cut down the fancy graphics, applets, audio-video clips in a webpage, as a result the response time of a web browser increases [53]. According to the experimental research done by Shneiderman et al. [115] on the response time behavior, some guidelines are suggested which are acceptable to the end user. They are [78][115]

- Web applications with less response time are mostly accepted by the end user.
- Applications having response time more than 15 sec are not acceptable.
- The think time of the user decreases as the response time decreases.
- Sometimes faster response times may lead to increased error rates.
- Suggested response time for different tasks are [78]:
  - For mouse and keyboard movements - 50 to 150 milli seconds.
  - For tasks done repeatedly - 1 second.

- For simple common tasks - 2 to 4 seconds.
- For multifaceted complex tasks - 8 to 12 seconds.

"Qcheck" is a free software utility tool, which displays the response time values between any two computer systems, which are located everywhere in the network [72].

4.2.1.2 Response time attributes in various contexts

For instance in the case of client server application, response time can be defined as the time taken by a server to respond to a particular request. In this scenario one can calculate response time at client end by subtracting end time from start time with the help of using a stop watch. While measuring response time some external factors that are likely effect the end results are bandwidth speed (i.e. upload and download speed), server execution speed, number of hops considered for packet transfer, latency etc. During the discussion of performance attributes related to web applications [58], it is evidently proven that the response time is the critical factor to the users of interactive systems. It observed that modest variations around average waiting time are acceptable but longer waiting times are not accepted [58].

Response time can be categorized in to three attributes in the perspective of embedded application software. They are Best Case Response Time (BCRT) [99], Worst Case Response Time (WCRT) [50][70][99] and Mean Response Time (MRT) [70]. In the domain of Priority based Functional Reactive Programming (P-FRP) [103], 'Actual response time' is considered an attribute for response time and in order to implement it 'Gap enumeration' algorithm is used to determine the end results [103].

In the context of mail servers the attributes considered for response time are average response time and message throughput [64]. During the response time analysis of complex embedded real time systems a 'RapidRT' algorithm is proposed for the Worst case response time attribute, which can be useful for the generation of response time values [76]. Actual Response Time (ART) attribute is considered for web services used in the context of grid and distributed systems [54]. In the case of mail servers [64] attributes like average response time and message throughput are considered as attributes. 'Average Response Time' (ART) attribute is considered as one of the response time attribute in the scenario of multi-tiered web applications [117].

Considering the response time measurement in SOA based application there are some Service Level Agreements (SLA) that are to be considered during the process [102]. They are

- **Response time threshold** - It is defined as the deadline by which a request must leave the datacenter (i.e. response should be completed within the time frame). These deadlines values are negotiated between the consumer and cloud provider [102].

- **Desired conforming percentile** - It is defined as fraction of requests for which consumers wants the cloud developer to execute them by the deadline [102].
- **Actual conforming percentile** - It is defined as actual number of requests for which the cloud developers had ensured the consumers to deliver them by the deadline [102].

In the scenario of virtual machines, huge amount of resources are required, as the system executes each task independently [50]. Therefore an XHive model [50] has been proposed for the case of virtual machines. In this case for measuring response time two attributes are mainly considered. They are discussed in the below figure.

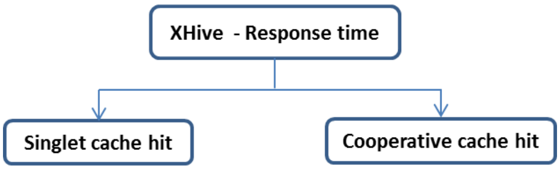

Figure 6 - XHive response time classification

In the context of component based applications, 'Average Response Time' attribute (ART) [104] is considered for measuring response time. For this a stock online test application is considered which is implemented by two middleware technologies like CORBA and EJB. Finally average response time values are been calculated in milliseconds for these two programming languages. As a result it is observed that EJB average response time values are proven to be small when compared to the values of CORBA [114]. Exact Worst Case Response Time (EWCRT) [57] attribute is considered for the case of fixed priority scheduled tasks with offsets and jitter [71][116]. Whereas in enterprise distributed real time and embedded systems response time attributes like worst execution time, average execution time and best execution time are considered [93].

Software aging is one of the important factor that is needed to be considered and it states that, the performance of software is degraded as the number of usage days are increased and there is more chances of crash, failure rates [51]. In order to measure the performance of such systems response time is the exact metric and attributes considered in this case are 'Mean Correct Response Time' (MCRT) and Failure rate [51].

Abort and Restart (ANR) model [113] has been proposed in the area of scheduling on multiprocessors and on distributed systems. In this scenario, performance attributes are evaluated and 'Maximum Response Time' (MRT) (i.e. worst case response time) is considered as the main attribute to measure the response time performance of the proposed ANR model [113]. While assuring performance of component based distributed systems, a 'Modified Mean Value Analysis' (MMVA) algorithm is used in order to improve the response time values in these applications [79][88].

In the domain of mission critical systems, 'Centre of Automation of Mission critical Systems' (CAMS) of Dutch Royal Navy had proposed an innovative approach in order to improve the software quality process [112]. They had designed this model by strictly implementing the ISO 9126 guidelines. As a result the term 'Response time' had been renamed as 'Response behavior' and defined three basic attributes under this category. They are collection time, transfer time and presentation time [112]. In this context, various other measures that are defined under ISO 9126 are discussed and their corresponding weights are also provided. Out of all other attributes response behavior had achieved 0.133 significance value and 0.045 for collection time, 0.054 for transfer time and 0.033 for presentation time respectively [112].

In the context of parallel rational database systems, estimation of response time attribute is carried out in three stages. They are preparation stage, mean resource response time estimation stage and mean query response time estimation stage [83].

4.2.2 Throughput

4.2.2.1 Conceptual definitions and formulae

Throughput can be stated in the following ways:

- *"Throughput attribute measures the output that can be successfully produced over a given period of time"* [81][100].
- *"Throughput attribute describes the amount of tasks which can be performed over a given period of time"* [98].
- *"Throughput measures the average execution speed of real time applications"* [111].
- *"Throughput quantifies the number of programs completed per unit of time"* [63].
- *"Throughput numbers tell the rate at which traffic can flow through a network in an given amount of time"* [72].
- In the application's perspective, *"how much useful (application-level) data is sent over a connection in unit time. Therefore throughput is measured as a ratio of the number of successfully delivered messages to the unit of time"* [77].
- "Throughput is the number of customer requests that the system can accept or deal with in a given time period" [74].

The formulas used to measure the throughput are:

Throughput = (No of transactions processed) / (Amount of time required) [114]**.**

{OR}

Throughput = (Number of tasks per unit of time) [107].

{OR}

Throughput rate = (Bytes sent by endpoint 1+Bytes received by endpoint 1)/
(end time of the process - start time of the process) [72].

Throughput is defined as the rate at which a system or service can process requests in a given amount of time. One important function that can be considered under this measure is maximum throughput. This function describes how throughput values vary with respect to load intensity [61][100]. "Qcheck" is a free software utility tool which displays the throughput values between any two computer systems which are located everywhere in the network [72].

4.2.2.2 Throughput attributes in various contexts

In the scenario of client server applications, throughput is defined as the number packets transferred in a given amount of time over the network and it is generally calculated in Mbits/sec [101]. In general one can define throughput as number of tasks executed in a certain period of time. Throughput can be categorized in to three attributes in the perspective of embedded application software. They are best case throughput ratio (BCTH), worst case throughput ratio (WCTH) and mean throughput ratio (MTH) [70] [55]. In the context of grid and distributed web services the following throughput attributes like NOPR, NOPRAR and NOPA are considered. They can be defined as [54]

- Number of packets, HTTP requests and responses, from client to application server for a single request (NOPR).
- Number of packets, HTTP requests and responses, for one complete request-response cycle on a web service from an application server (NOPRAR).
- Number of packets, HTTP requests and responses, from application server to web service server for acknowledgment (NOPA).

In the context of EJB (Enterprise Java Bean) application servers, throughput is measured in TPS (i.e. Transactions per Second) for six different EJB server applications [114] by varying the client loads from 100 to 1000. In this case 100, 300, 500 and 1000 clients are considered and throughput values of six EJB servers are calculated under these loads. These results suggest us that the throughput performance of applications is decreasing constantly, when the client loads are increased. From these results, it is observed that the throughput values depend on the external factors like system load etc., and the values keep on changing at different conditions. In the same scenario another case is taken in consideration, where two different component architectures like Session Bean (SB) and Session Bean combined with Entity Bean (SB+EB) are considered for measuring throughput and observe the difference in their throughput values [114]. From the results of these tests, it is observed that the throughput rate of (SB+EB) is reduced by 50%, when compared to SB component architecture. Therefore it is once again proved that the throughput rate decreases as the load increases [114]. In the domain of context aware mobile applications, there are two attributes considered for throughput. They are defined as [105]

- **Network Capacity** - It defines the network bandwidth provided between two hosts in a given amount of time and it is measured in number of bytes transferred per second.
- **Processor Capacity** - It defines the processor capacity of handling tasks in a given amount of time and it is measured in number of processes handled per second.

Throughput in the case of web applications is classified under two categories. They are input and output throughputs of the system. The input throughput states the rate of requests submitted per unit time to the server. The output throughput states the rate of responses received by the client per unit of time [58]. In the case of internet data centers the attributes considered for throughput are 'average throughput' which is measured using number of requests handled per second and 'maximum throughput' values measured in percentages [60]. While considering EJB components, the performance is measured by considering two EJB components (i.e. Stateless session bean only and Stateless session facade) and later throughput values are calculated. Finally throughputs of various e-commerce applications are tested on 6 different J2EE application servers and values are displayed [66].

In the context of virtual machines throughput attributes can be measured with the help 4 metrics. They are rxpck/sec (i.e. packets received per second), txpck/sec (i.e. packets transmitted per second), rxbyt/sec (i.e. bytes received per second) and txbyt/sec (i.e bytes transmitted per second) [89]. In the context of marine navigation, ships often come in the shipyard and go out and there comes the importance of performance issues. Throughput is one of the important factors that are to be considered in this issue and it is defined as the ratio of the number of successfully received transmissions, with the number of transmission attempts. It is assumed that one transmission occupies exactly one slot [67].

4.2.3 Turnaround time

4.2.3.1 Conceptual definitions and formulae

- *"Turnaround time is the waiting time, that a user experiences after issuing an instruction to start and complete a group of related tasks. It is the time span from starting the tasks until the last task finishes"* [98].
- *"Turnaround time quantifies the time between submitting a job and its completion"* [63].
- *"Turnaround time is the resolution time which has been calculated by end time minus start time"* [84].

The formula considered for measuring turnaround time of an application is defined below

Turnaround time = (Time taken to initiate the request at client end) + (Travel time) + (Time taken by server to execute the request) + (Waiting time).

4.2.3.2 Turnaround time attributes in various contexts

Turnaround time can be categorized in to three attributes in the perspective of embedded application software. They are Best case Turnaround Time (BTT), Worst case Turnaround Time ratio (WTT) and Mean Time for Turnaround (MTT) [70].

In the context of multi program workloads, a user oriented performance attributes considered for turnaround are 'Average Normalized Turnaround Time' (ANTT) and 'Maximum Normalized Turnaround Time' (MNTT) [63].

4.2.4 Memory utilization

4.2.4.1 Conceptual definitions and formulae

- *"The total amount of memory needed for a particular component in order to operate"* [80][81][92].
- *"Disk utilization is the amount of disk space used to store its code and the space used to store the temporary internet files"* [81].
- *"In real time applications, memory utilization can be defined as the total memory utilized by all the applications on an average"* [111].

The formulas used to calculate the total memory utilized by an application are defined as

Total memory utilized = (Memory consumed by application) +
(Memory consumed by its related functions).

{OR}

Memory utilized = (Total internal system memory occupied by an application).

{OR}

Memory utilization message density = (No. of memory related error messages) /
(No. of lines of code directly related to system calls).

4.2.4.2 Memory utilization attributes in various contexts

Memory utilization can be categorized in to three attributes in the perspective of embedded application software. They are maximum memory utilization (MMU), Mean occurrence of memory error and Ratio of memory error/ time [70]. 'Power Consumption Factor' (PCF) attribute is considered as one of the memory utilization attribute in the scenario of multi-tiered web applications. This attribute is used to represent the amount of CPU memory consumed for request executions in the system [117]. In the domain of real time applications, memory utilization performance attribute considered is 'Average CPU utilization rate' [111]. In the context of evaluating memory related performance issues for windows and its applications two attributes are formulated. They are processor utilization and average CPU utilization [97]. In real time applications disk utilization is one of the attribute that can be used to measure the memory performance of different real time products. Disk utilization can be used to identify the amount disk space utilized to execute particular tasks. This attribute is important in real time systems, since in real time distributed systems applications runs at different locations and conservation of disk space plays a vital role in these types of systems [91]. While assuring performance of component based distributed systems, a 'Replication and Allocation' (RA) algorithm is used in order to improve the memory usage and also 'Total CPU utilization' attribute is used to measure the memory utilization of these systems [88].

In the scenario of virtual machines, huge amount of resources are required, as the system executes each task independently [50]. Therefore an XHive model [50] has been proposed for the case of virtual machines. In this case for measuring memory utilization a total of seven attributes are considered. They are discussed in the below figure.

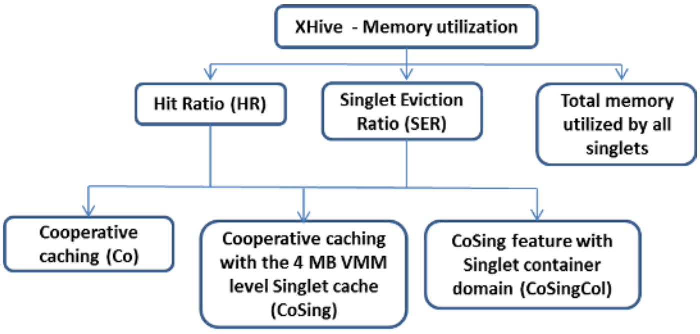

Figure 7 - XHive memory utilization classification

As a result memory attributes in this context of virtual machines can be measured with the help 4 metrics. They are wtps (i.e. write requests per second), bwrtn/sec (i.e. data written to a block of devices in a second), rtps (i.e. read requests per second) and bread/s (i.e. data read from block device in a second) [87][89]. In the domain of context aware mobile applications, there are four attributes considered for memory utilization. They are defined as [105]

- **Network Usage** - It defines the aggregate network bandwidth used between two hosts and it is measured in bytes.
- **Memory Usage -** It defines the aggregate memory used by the host system and it is measured in bytes.
- **Memory Capacity -** It defines the overall memory available at the host system and it is measured in bytes.
- **Processor Utilization** - It defined the aggregate processor usage at the host system and it is measured in integer.

4.2.5 I/O utilization

4.2.5.1 Conceptual definition and formulae

- *"I/O utilization is defined as the number of input or output buffers processed by an application in a specific time period"* [107].

The conceptual formulae defined for I/O resource utilization are:

I/O utilization = (Number of input/output buffers processed by an application). [107]

{OR}

I/O utilization message density = (No. of I/O related error messages) / (No. of lines of code directly related to system calls).

4.2.5.2 I/O utilization attributes in various contexts

In the scenario of mail server applications, I/O utilization can be measured in three attributes. They are number of active web mail clients available, Concurrent mail users per server, and Management of quotas on message and mail file size [64]. I/O utilization can be categorized in to five attributes in the perspective of embedded application software. They are I/O devices resource utilization, I/O loading limits, I/O related errors, Mean I/O fulfilment ratio and User waiting time of I/O devices utilization [70].

Jongmoo et al. evaluated the I/O utilization performance [52] by considering two applications like Cscope (used for 'C' source examiner), Glimpse (used for information retrieval) which run on FreeBSD, which is a UNIX based operating system. 'Disk I/O utilization' [52] is the attribute considered to evaluate the performance of these two applications and the metrics used for this attribute is 'total number of requests placed by the product to the operating system' [52]. Initially disk I/O utilization values are calculated for each application. Later these two applications are combined as a single process and then the disk I/O values are calculated. In real time applications attributes like maximum disk I/O utilization is used to measure the I/O performance of the system [90].

In real time distributed applications I/O resources play a pivotal role as the execution of tasks takes place at different locations. In order to measure the performance factors in this conditions two attributes were considered, they are worst case I/O utilization and average I/O resources utilization [91]. Now in the scenario of virtual machines, execution of various tasks take place independently at various locations and as a result the disk I/O utilization rate increases significantly [50]. When virtual machines access large amount of data, that is stored in shared networks then main problem occurs in this situation is the entire disk I/O resources are exhausted. Therefore there is a need to reduce the disk I/O operations for virtual machines that have shared working sets. For this reason an XHive [50] (Efficient Cooperative Caching for Virtual Machines) model is proposed, in order to reduce this threat. During this study on increasing the performance of I/O disk utilization a total of 3 attributes are considered for measuring [50]. They are represented in the below figure.

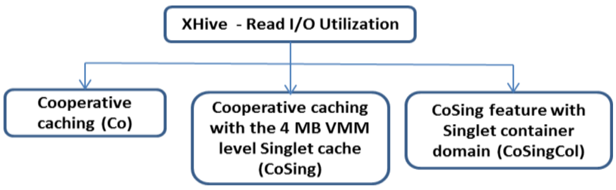

Figure 8 - XHive read I/O utilization classification

4.3 Attributes suitable for measuring web browsers

From the above results and by doing some general analysis, we had reformulated the efficiency attributes and found only three attributes that are suitable for measuring efficiency of web browsers. They are response time, throughput and memory utilization [70]. Other efficiency attributes like turnaround time, transmission resource utilization and I/O resource utilization are ignored mainly because of the following reasons, these attributes are more related to the networking side and they does not have any impact on the web browsers. The efficiency attributes that are found suitable for the study are discussed in the next part of this section.

4.3.1 Response time

Response time of a web browser is defined as the time taken by a browser to open and load a particular URL (i.e. web page). Generally web pages are of two types, they are HTML and Flash pages. In this case, we are only considering HTML pages as while loading flash pages a local script will be executed in the system and also it will not show any impact on the web browser. Further we can categorize webpages in to two categories (i.e. Cached and Uncached pages). The response time varies for cached pages (i.e. pages when once opened in a web browser, they will be stored in the system cache memory and while loading the same page again the page elements will be quickly retrieved from the temporary memory, but not from the server) and uncached pages (i.e. all the elements in the page are loaded directly from the server there will not be any traces in the system). Based on the above classification these categories are formulated for response time attribute. They are

- Response time - HTML pages - Cached data.
- Response time - HTML pages - Uncached data.

4.3.2 Throughput

Throughput is defined as total number of tasks that are executed in a given amount of time. In the case of web browsers, it can be considered as the total number of frames that a particular web browser is loading while performing a particular task is measured. The metric for measuring throughput of a web browser is 'Frames Per Second (FPS)'.

4.3.3 Memory Utilization

Memory utilization is defined as the total amount of memory consumed for executing a particular task. In the case of web browsers, memory utilization can be measured in different ways like memory consumed by a browser to open an empty window, memory consumed for opening a single page in the browser tab and memory consumed by the browser for opening multiple pages. Based on these parameters, the following three categories are formulated for memory utilization attribute. They are:

- Memory consumed for opening an empty browser window.
- Memory consumed for opening a HTML page in a single tab of a web browser.
- Memory consumed for opening multiple HTML pages in two different tabs and in a same web browser window.

5 EXPERIMENT

From the results of Literature Review, it is clearly evident that the Efficiency quality attribute is one of key software quality attributes and these results also indicate us to conduct further studies by comparing efficiency standards of different web browsers. Therefore an experiment is planned to be conducted by following the guidelines given by Wholin et al. [7] in this section.

5.1 Motivation

An experiment is always a formal, rigorous and controlled investigation. Controlled experiments are launched when one want to control the experiment environment and also to manipulate the behavior of attributes precisely and systematically. Generally in controlled experiments execution & measurement control are accurate and at the same time investigation cost and ease of replication are very high when compared to surveys and case studies [7]. Organizing experiment is not as easy as conducting a survey or a case study and one should carefully prepare, conduct and analyze the whole process properly. The advantage of conducting a controlled experiment is one can control subjects, objects and instrumentation and another advantage is it allows us to perform statistical analysis by formulating different hypothesis. The main aim is to manipulate one or more variables and control all other variables at fixed levels and later perform statistical analysis on the collected data. In this experiment we had calculated values for variables response time, throughput and memory utilization of various web browsers by keeping webpage size, bandwidth speed and server execution speed at constant.

Our main idea behind conducting this experiment is to study efficiency quality attribute in various web browsers and analyze the efficiency standards of each web browser. Later based on the values generated for each web browsers an Efficiency Baseline Criteria (EBC) is proposed and industry practitioners can use these EBC values to test the efficiency standards of their newly developed web browsers. These EBC values are more useful to OSS developers, who are looking to explore new horizons in the development of new web browsers. Therefore this experiment was planned and conducted by considering the needs of OSS developers and practitioners. According to [7] an experiment involves several different steps. They are:

- Experiment definition.
- Experiment planning.
- Experiment design.
- Experiment execution and
- Experiment results and analysis.

5.2 Experiment definition

- **Object of study:** The objects that were studied in this experiment are: efficiency quality attribute of different web browsers. In this experiment we have considered two sets of web browsers. One set consisting of web browsers from freeware & CSS (i.e. Case A) and other set consisting of OSS web browsers (i.e. Case B).
- **Purpose:** The purpose of this experiment is to calculate efficiency values (i.e. Response time, Throughput and Memory Utilization) for Case A (i.e. freeware & CSS) web browsers and then EBC values are proposed based on the results obtained. Later we compare the results of EBC with Case B (i.e. OSS web browsers) and figure out the efficiency standards of OSS web browsers.
- **Quality focus:** The quality focus was on the efficiency standards (i.e. Response time, Throughput and Memory utilization) of web browsers.
- **Perspective:** The perspective was from the OSS web browser developer's and researcher's point of view.
- **Context:** Context is defined as the environment in which an experiment is executed. In this experiment both author's themselves acted as users of web browsers and evaluated efficiency standards for Case A & Case B web browsers. The context of our experiment is "multi-test within an object study" [7] as we are examining a single object across a set of subjects.
- **Summary of definition:** Analyze *the efficiency standards of different sets of web browsers* for the purpose of *evaluation* with respect to *Response time, Throughput and Memory utilization* from the point of view of *the OSS web browser developers and researchers* in the context of *both the authors*.

5.3 Experiment planning

According to Wohlin et al. [7], there are six steps in the planning phase of the experiment. They are Context selection, Hypothesis formulation, Variables selection, Selection of subjects, Experiment design and Instrumentation.

5.3.1 Context selection

In this section, we deal with the selection of experiment environment where the whole process is executed. The context of this experiment lies within the selected set of web browsers, where test subjects evaluate the efficiency standards of these web browsers with the help of an online website, which was chosen by authors. In order to meet the experiment requirements, we had chosen different web browsers. Web browsers can be classified into 3 types; they are proprietary, open source and freeware. Proprietary web browsers are those products, where the owner holds the copyright law and end users have to purchase the full product in order to use it. The owner of the product will not allow end user to change the functionality and code of the product. Whereas Freeware products are available free of cost or at optional fee, but the end user will not have access rights to modify source code of the product. Similarly, Open source products are generally available at free of cost with complete access to source code of the product. For performing our experiment, web browsers had been classified under two cases. In the first case (i.e. Case A), we had chosen top 4 web browsers from both freeware and proprietary (i.e. Mozilla Firefox, Google chrome, Opera and Internet Explorer), these browsers are chosen based on the popularity and usage of the product by the end users. For the second case (i.e. Case B), we had chosen a set of 4 OSS web browsers (i.e. Midori, Seamonkey, Qupzilla and Arora). The efficiency values are calculated for both the Case A & B with the help of tools and methods that are discussed in the section 5.4.4 and overview of results were reported in 5.6 section. The below Table provides information related to the type and version of web browsers that are selected for the study.

Table 4 - Web browsers classification

CASE A		CASE B	
Web Browser	**Version**	**Web Browser**	**Version**
Google Chrome	23.0.1271	Midori	0.4.7
Mozilla Firefox	17.0.1	Seamonkey	2.14.1
Internet Explorer	9.0.8112	Qupzilla	1.3.5
Opera	12.12	Arora	0.10.0

5.3.2 Hypothesis formulation

According to Wohlin et al. [7], hypothesis can be formulated in two ways. They are Null hypothesis (H_0) and Alternative hypothesis (H_1). The hypotheses considered for our experiment are discussed below:

Null hypothesis (H_0):

Efficiency standards of Case B (i.e. OSS) web browsers are equivalent to the efficiency standards of Case A web browsers when compared.

This hypothesis can be further classified into three sub-hypotheses:

Null hypothesis ($H1_0$): Response time values of Case B (i.e. OSS) web browsers are equivalent to the Case A web browsers.

Null hypothesis ($H2_0$): Throughput values of Case B (i.e. OSS) web browsers are equivalent to the Case A web browsers.

Null hypothesis ($H3_0$): Memory utilization values of Case B (i.e. OSS) web browsers are equivalent to the Case A web browsers.

Alternative hypothesis (H_1):

Efficiency standards of Case B (i.e. OSS) web browsers are not equivalent to the efficiency standards of Case A web browsers when compared.

This hypothesis can be further classified into three sub-hypotheses:

Alternative hypothesis ($H1_1$): Response time values of Case B (i.e. OSS) web browsers are not equivalent to the Case A web browsers.

Alternative hypothesis ($H2_1$): Throughput values of Case B (i.e. OSS) web browsers are not equivalent to the Case A web browsers.

Alternative hypothesis ($H3_1$): Memory utilization values of Case B (i.e. OSS) web browsers are not equivalent to the Case A web browsers.

5.3.3 Variables selection

5.3.3.1 Dependent variables

The following are the dependent variables that are considered in our experiment:

- **Response time:** Generally response time is defined as the time taken by an application to respond a user request. It is the time between the launch of user request to first response from the server that reaches the client application. In order to measure response time Numion tool [40] is used in our experiment.
- **Throughput:** Generally throughput is defined as the total number of tasks that are executed in a given amount of time. In the case of web browsers, the total number of frames that a particular web browser is loading while performing a particular task is measured. The metric for measuring throughput of a web browser is 'Frames per Second'. In order to measure FPS in our experiment, we had used the famous "IE fish tank test" [42] application. This test is mainly useful to test the HTML5 functionality of the web browser, where it shows the number frames considered by the browser to display the output (i.e. based on movement of fish in the tank).
- **Memory utilization:** Generally memory utilization is defined as the total amount of memory consumed for executing a particular task. In the case of web browsers, memory utilization can be measured in different ways like memory consumed by a browser to open a blank window, memory consumed for opening a single page in the browser tab as well as opening different pages in multiple tabs. In order to measure the memory consumed by web browsers, we had used the "Resource monitor" option under "task manager" tool in our experiment.

5.3.3.2 Independent variables

The following are the independent variables that are considered in our experiment:

- **File size:** Response time values change, when the file size of the web application changes. In our experiment, we had considered only HTML web page, where the size is constant.

- **Bandwidth speed:** While calculating response time of a web browser, one important factor that affects the end result is the bandwidth speed. If the bandwidth speed (i.e. upload and download speed) changes the response time values will also change accordingly. Therefore there is a need to regulate or shape the speed of bandwidth in order to retain constant speed at the client end. For this we had used net limiter tool [43] to shape the bandwidth. Net limiter is a freeware tool [43] and it is commonly used tool to control the speed of bandwidth. In our experiment, response time values were generated at two different bandwidth speeds (i.e. 0.3 and 0.8 Mbps) and 0.8 Mbps constant bandwidth speed is used to calculate values for throughput & memory utilization. The main motivation behind selecting two different speeds is that the bandwidth speed will not be same in all areas across the globe. It varies from place to place, for example in India the Internet bandwidth commonly used is 0.3 to 1 Mbps and in Sweden we are getting more than 50 Mbps. The interesting point that is to be considered is one can reduce the speed of the bandwidth (i.e. if the user is having 100 Mbps speed connection, one can reduce it to 0.8 Mbps at client end by using net limiter tool) according to the needs, but one cannot increase the speed of the bandwidth (i.e. if the user is using 0.5 Mbps internet connection, user cannot boost the speed of the bandwidth to 1 Mbps). Therefore these reasons motivated us to calculate at two different low bandwidth speeds, so that there can be some flexibility to the end user. The process of delimiting the bandwidth speed is discussed in later sections.
- **Cached / uncached webpages:** Generally webpages are of two categories (i.e. Cached and Uncached). Response time varies for cached pages (i.e. pages when once opened in a web browser, they will be stored in the system cache memory and while loading the same page again the page elements will be quickly retrieved from the temporary memory, but not from the server) and uncached pages (i.e. all the elements in the page are loaded directly from the server, there will not be any traces in the system). Hence in our experiment we had considered both cached and uncached webpages.
- **Load variation:** Generally when the loads vary on the web browsers, throughput and memory utilization values changes. In order to measure throughput, we had used the fish tank test, where we varied the load by selecting 20, 100 and 250 fishes in the tank and observed that with variation in the load the FPS (i.e. throughput) values are changed. Similarly, for measuring memory utilization values, we varied the load by considering 3 scenarios like web browser with empty tab (i.e. no web page is loaded), single tab loaded with HTML webpage and multiple tabs (i.e. with combination of two HTML webpages).

5.3.4 Selection of subjects

According to [7], selection of subjects is very important in any experiment process. There are five different ways for selecting subjects in an experiment. They are simple random sampling, systematic sampling, stratified random sampling, convenience sampling and quota sampling. In this experiment, both the authors themselves acted as test subjects because the experiment execution process is lengthy & more functionality to be covered and also it will take more time to understand the execution process. Hence authors themselves became test subjects and efficiency values for each scenario are calculated for the selected set of web browsers.

5.4 Experiment design

5.4.1 Experiment design type

According to Wohlin et al. [7], there are three important design principles that are followed while designing an experiment [7]. In our experiment the design type "one factor with two treatments" is selected, because we have investigated one factor (i.e. Efficiency quality attribute) under two treatments: Treatment 1 (i.e. Case A web browsers) and Treatment 2 (i.e. Case B web browsers). The below table shows an overview of design type used:

Table 5 - Experiment design overview

One Factor	Treatment 1	Treatment 2
Efficiency Quality Attribute	Case A web browsers.	Case B web browsers.

5.4.2 Experiment scenario's

As discussed in section 4.3, following scenarios are considered in this experiment. They are:

- Response time - Uncached HTML web page - 0.3 Mbps.
- Response time - Cached HTML web page - 0.3 Mbps.
- Response time - Uncached HTML web page - 0.8 Mbps.
- Response time - Cached HTML web page - 0.8 Mbps.
- Throughput - 20 fishes in the tank.
- Throughput - 100 fishes in the tank.
- Throughput - 250 fishes in the tank.
- Memory utilization - Empty web browser.
- Memory utilization - Single HTML tab.
- Memory utilization - Multiple HTML tabs.

5.4.3 Pre-Procedures

For conducting this experiment, we had considered some assumptions and standard procedures that are to be implemented at the beginning of each scenario. While calculating response time of a web browser, one important factor that affects the end result is the bandwidth speed. If the bandwidth speed (i.e. upload and download speed) changes the response time values will also change accordingly. Therefore there is a need to regulate or shape the speed of the bandwidth in order to retain constant speed at the client end. For this we had used net limiter tool [43] to shape the bandwidth. Net limiter is a freeware tool [43] and it is commonly used tool to control the speed of bandwidth. In our experiment the response time values were generated for different bandwidth speeds (i.e. 0.3 and 0.8 Mbps). The process of delimiting [43] the bandwidth speed is discussed below.

Step by step procedure for delimiting the bandwidth speeds:

- First download and install the “Net limiter 3.0.0.11” tool [43] from internet. After installing it in the system, open “netlimiter.exe” file and there one can observe the various web browser names and besides that there will be an empty box and checkbox to tick (i.e. the name of the web browser only appears, when it is opened before opening of this software).
- There will two such options boxes for each web browser, one for the adjustment of upload and another one for the adjustment of download speed.
- Now check the speed of the bandwidth by using online tool at www.speedtest.net [44]. Click on this URL and begin the test, then it will display the upload and download speed. Based on those results, delimit the bandwidth speed that nearly matches our bandwidth speeds (i.e. for example, if your upload and download speeds are 0.5 Mbps then you can delimit to 0.3 Mbps).
- In order to delimit the speed just open the “net limiter” tool and see for the particular browser option, for example if user want to delimit the Google Chrome browser download and upload speed to 0.3 Mbps. Then select chrome link and besides that there will 2 input boxes, where the values 0.3 (i.e. delimiting value) should be entered and also provide firewall permission for the tool by clicking on the checkbox that is provided.
- Now once again check the bandwidth speed from www.speedtest.net and confirm that the speed of the bandwidth is delimited to 0.3 Mbps. In order to get more clear information on delimiting the bandwidth process, please refer this video at YouTube (i.e. http://www.youtube.com/watch?v=4zNn41VLwoE). In this way one can delimit the upload and download speeds using Net limiter tool.

Another important assumption that is to be considered is the web page size (i.e. normally this factor is not much considered and it is assumed that it has minimal effect on the results, but later it is proved that there will be lot of variations in the end results with the changes in size of pages). As the response time values and memory usage varies with the change in the page size (i.e. for example the response time of a 100 kb web page is less when compared to the response time for a 1000 kb web page. Similarly the memory consumption will also vary for these webpage sizes), therefore it very important that the web page size should be constant throughout the analysis. For this reason we had considered a website, where there will not change or update in the page sizes.

Another assumption considered is the execution of the server speed is constant throughout the data extraction process. The webpage considered for our case is in very little use at present and no changes are done to it. So there will not be any client requests at the server other than our present requests. Hence there is a possibility that there will be only one client request, that is been executed at the server and therefore it can be assumed that the server execution speed is constant throughout the data extraction process. While conducting the experiment, we had controlled the web browser activities by removing all update alerts, all add-ins and extensions.

At the start of each process the following initial steps (i.e. standard pre-procedures) are to be implemented. They are

- Check the speed of the bandwidth and delimit the speed of bandwidth to appropriate values using the Net limiter tool.
- While calculating values for uncached data, clear the temporary internet files, and also clear the browser cache memory and history. If the user is unfamiliar with the above process, they can use "private browsing" option that is available in every browser.
- While calculating the memory consumption, only the respective browser windows should be opened and no other browser window should be active.

Assuming system and internet configuration:

The following system and internet configuration is considered constant throughout the data collection process. In order to maintain consistency among the data collected, the whole process is done on single computer.

Table 6 - System configuration

Type	Configuration
Model type	**Dell XPS**
Processor	**Intel Core i5**
RAM	**4 GB**
System type	**64 bit operating system**
Processor speed	**2.53 GHz**
Operating system	**Windows 7 Home Premium**
Hard disk memory	**500 GB**
Internet connection type	**Wired connection using cable**

5.4.4 Instrumentation

In this section all the tools used in the experiment, along with their methodologies are discussed. Generally, in order to measure the response time, automatic stop watch tools are used which will automatically measure the clock time. Similarly to measure the frames loaded under given settings (i.e. throughput), we had considered the famous fish tank test designed by Microsoft and similarly to calculate the memory consumption, the 'resource monitor' option under the properties of system task manager was used in order to calculate the values. In this section, we will discuss the step by step procedure on usage of each tool. These tools are chosen in such a way that it can be implemented for any type of web browser. We had followed these below procedures to measure both case A and case B efficiency values.

5.4.4.1 Tool used for measuring Response time - Cached & Uncached data - HTML web page

For measuring response time of a HTML page, initially we searched in internet in order to identify the appropriate tool. From the search results, it is observed that 'Numion' stop watch tool [40] is useful to measure the response time of web browser. The Numion stop watch tool [40] is a simple online application, which calculates the time by running a small java script program on the computer, but not on the server side. It starts measuring time when the moment the web browser start loading the URL (i.e. web page) and stops when the browsers completes loading the page (i.e. stops after receiving 'done' signal from the web browser). The total time captured will include complete loading of images, all frames of the web page and its related java script programs [40]. Refer Appendix C for an overview of process. The steps that need to be followed in order to measure the response time values from the tool are as follows:

Step by step procedure:

- Follow the pre-procedure steps that are discussed in the 5.4.3 section.
- Then open the Numion page web link (i.e. http://www.numion.com/Stopwatch/index.html) and then paste the appropriate web link (i.e. URL) in the box provided.
- Then click on the 'start stop watch' button, then the corresponding HTML web page is loaded and stopwatch calculates the time taken by the web browser to in order the initiate the request of the user.
- In the next page, this tool displays the time at the top of the web browser. The time will be displayed in seconds.
- The value attained for the first time after completing the above steps is the response time of a web browser for uncached data. In order to attain the values for cached data, just refresh the web browser by clicking 'F5' key and observe the values (Note: While calculating values for cached data, please do not clear memory cache and temporary internet files).

5.4.4.2 Tool used for measuring Throughput values (i.e. in Frames per Second (FPS))

For measuring the FPS of a web browser, we had selected the famous "IE fish tank test" [42]. This test is mainly useful to test the HTML5 functionality of the web browser, where it shows the number frames considered by the browser to display the output (i.e. movement of fish in the tank). The output can be varied by 1, 20, 100, 250, 500 and 1000 fishes. It is noted that the FPS value cannot exceed 60 because the screen of the computer cannot load these many frames in a given second. These are the steps to be followed in order to get the throughput values for different browser with the help of IE fish tank tool [42]. Refer Appendix C for an overview of process.

Step by step procedure:

- Follow the pre-procedure steps that are discussed in 5.4.3 section.
- Then load the URL of Microsoft IE fish tank web page in each web browser (i.e. http://ie.microsoft.com/testdrive/Performance/FishIETank/Default.html).
- On the right hand side of page screen, one can observe the FPS meter, window size, and options to choose the number of fish and on the left side the movement of fishes can be observed.
- By varying the count of fishes, one can observe the changes in the FPS meter and note down those values for different web browsers. In this way one can calculate the FPS (i.e. frames loaded per sec) by a web browser.

5.4.4.3 Tool used for measuring Memory Utilization values

Memory utilization can be defined as the total memory consumed by the web browser, while different types of tasks were performed in the browser. One can use the "Resource monitor" option under "Task manager" to observe the consumption of the memory. For this instance, the process of measuring the total memory occupied for an empty web browser is discussed and same procedure can be adopted for any situation. These are the steps to be followed in order to know the memory consumption of different web browser. Refer Appendix C for an overview of process.

Step by step procedure:

- Follow the pre-procedure steps that are discussed in 5.4.3 section.
- Just open an empty web browser window. Now right click on the task bar and select the "start task manager" option. Then a separate window will be opened and then go to "performance" tab and then click on the "resource monitor" button.
- Now a separate window will open and in that click on the memory tab. Then one can observe the memory consumption of various web browsers.
- For example, if an empty Google chrome window is open then in the resource monitor window, one can observe chrome.exe file, where it will show the memory occupied by the particular browser.

- If multiple tabs or windows of the same browser are open then there may be different “chrome.exe” links. At that time, just add the values of all the links that are related to the same web browser and the final figure show the total memory consumption.

5.5 Experiment execution

According to Wohlin et al. [7], the experiment operation involves three steps: Preparation, Execution and Data validation.

5.5.1 Preparation

In prior to our experiment process, we have kept all our instruments ready and all our subjects are well trained in advance. Here both the authors themselves have performed the experiment process for Case A and Case B web browsers and then compared the efficiency values of both the cases. The instruments (i.e. tools used) that are used in the experiment were discussed in the 5.4.4 section. The experiment process for calculating Case A and Case B web browsers were explained in the below section.

5.5.2 Execution

In our experiment, initially both the authors had prepared a manual with execution steps and then performed the whole experiment process for Case A and Case B web browsers. In this section, we had defined the process of our execution steps. The measurement process of our experiment can be divided in to two parts. They are:

- Measuring Efficiency values of Case A web browsers.
- Measuring Efficiency values of Case B web browsers.

5.5.3 Execution process implemented for Case A web browsers

In this section, execution process for collecting data for Case A web browsers like Google chrome, Mozilla Firefox, Internet Explorer and Opera are discussed. Efficiency values are calculated for the selected scenarios using tools and procedures as discussed in 5.4.4 section. Finally results of each scenario are discussed in the 5.6 section.

5.5.3.1 Response time of web browsers - uncached data - HTML page

In this section response time for uncached HTML page is calculated for the four selected web browsers. For measuring response time, a HTML web page is chosen [45]. The web page selected is representative as it is actively used by many viewers daily and web traffic for this page is low when compared to other web pages. As discussed in the section 5.4.4.1, Numion stop watch tool [40] is used to measure the response time of the selected web browser. Always remember to clear temporary internet files and cache memory before starting each task. For each web browser the whole process is repeated for 25 times and the average value is considered, which denotes the response time of the particular web browser.

Step by step procedure:

- Response time of web browsers are calculated at two different bandwidth speeds (i.e. 0.3 and 0.8 Mbps). Now delimit the bandwidth speed by following the steps that are discussed in the section 5.4.3 and check the current bandwidth speed (i.e. upload and download speeds) using speed test online tool [44].
- Now select a particular web browser and open the Numion tool home page [40] in the browser window and copy the URL of HTML web page [45] in the box provided and follow the steps as discussed in the section 5.4.4.1.
- Repeat the same process for 25 times and the reading attained after each round is noted down in a table. The whole process is repeated for two different bandwidth speeds and all the readings are recorded in the table. The whole process of measuring is depicted in the below figure.

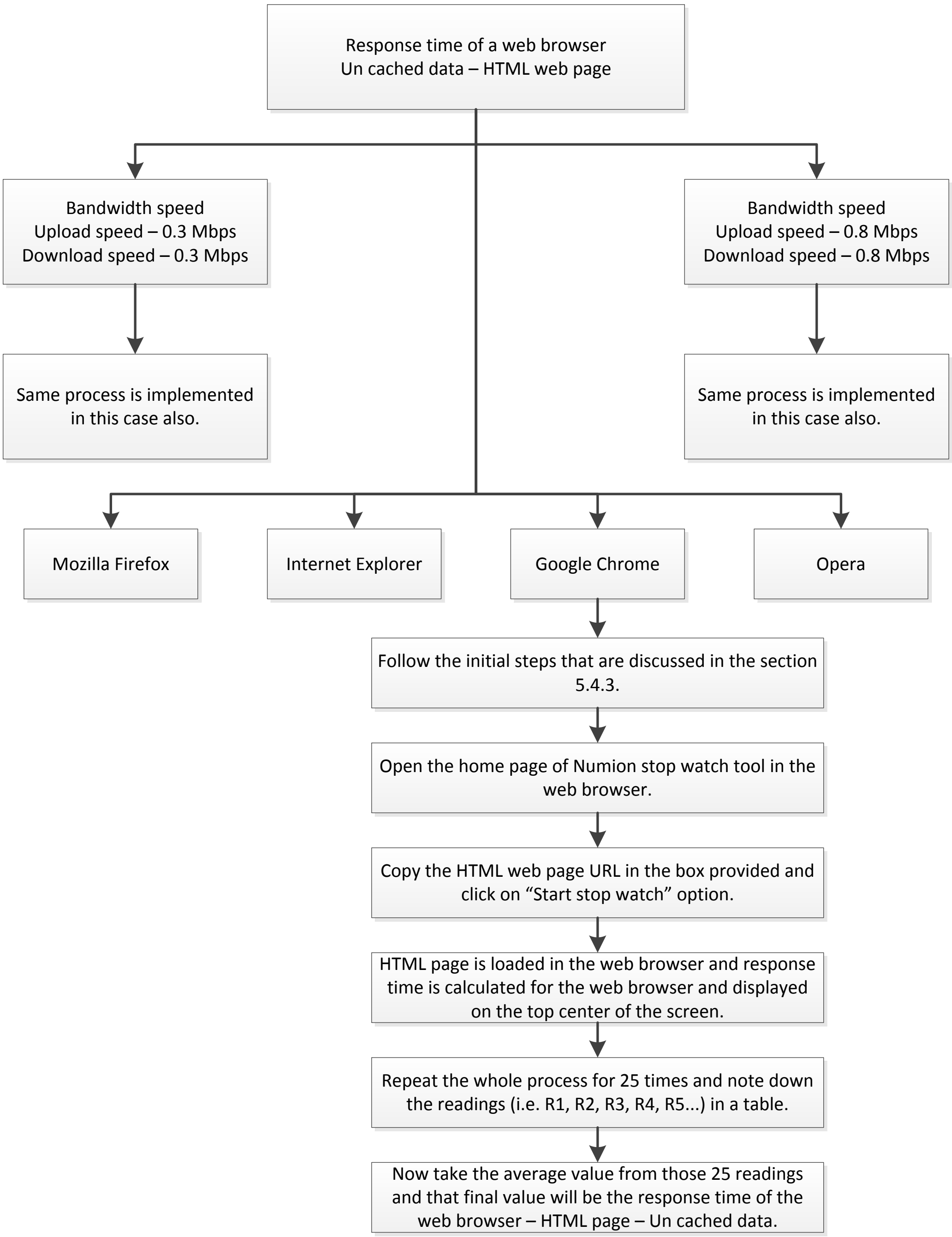

Figure 9 - Procedure to measure RT of a web browser - Uncached HTML

5.5.3.2 Response time of web browsers - cached data - HTML page

In this section response time for cached HTML page is calculated for the four selected web browsers. For measuring response time, a HTML web page is selected [45]. As discussed in the section 5.4.4.1, Numion stop watch tool [40] is used to measure the response time of the selected web browser. Always remember not to clear temporary internet files and cache memory before starting each task and also initially load the web page once before collecting the actual data (i.e. in order to store data in system cache memory). For each web browser the whole process is repeated for 25 times and the average value is considered, which denotes the response time of the particular web browser.

Step by step procedure:

- Response time of web browsers are calculated at two different bandwidth speeds (i.e. 0.3 and 0.8 Mbps). Now delimit the bandwidth speed by following the steps that are discussed in the section 5.4.3 and check the current bandwidth speed (i.e. upload and download speeds) using speed test online tool [44].
- Now select a particular web browser and open the Numion tool home page [40] in the browser window and copy the URL of HTML web page [45] in the box provided and initially load the page once (i.e. now the data is stored is cache memory). Then follow the steps as discussed in the section 5.4.4.1.
- Repeat the same process for 25 times and the reading attained after each round is noted down in a table. The whole process is repeated for two different bandwidth speeds and all the readings are recorded in the table. The whole process of measuring is depicted in the below figure.

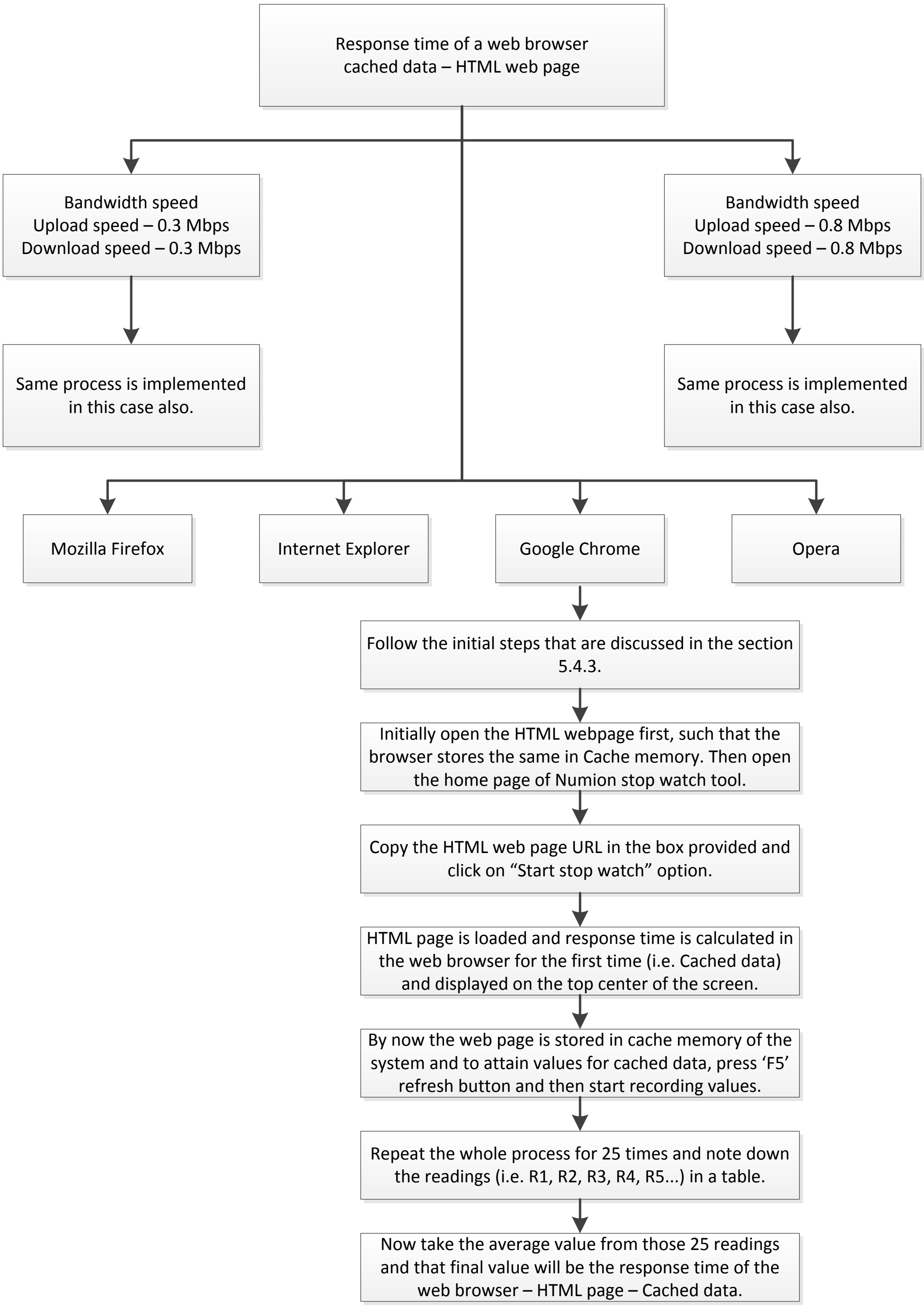

Figure 10 - Procedure to measure RT of a web browser - Cached HTML

5.5.3.3 Throughput (FPS)

In this section throughput of web browsers is measured. Generally the term “throughput” is defined as number of tasks executed in a given amount of time. In terms of web browser, throughput can be calculated in the form of number of frames loaded in a given amount of time. As discussed in the section 5.4.4.3, Frames Per Sec (FPS) can be measured using the famous “Fish tank” test [42] designed by Microsoft. This test is conducted online and while conducting this test, a constant speed is observed (i.e. 1 Mbps upload and download speed) at client end.

Step by step procedure:

- Initially follow the steps that are discussed in the section 5.4.3.
- This tool uses the movement of fish to determine the frame rendering process in a web browser and the FPS value tends to change as there is an increase in number of fish movement in the tank. It is also observed that the maximum value of FPS is 60 on any computer monitor [42].
- In this experiment process the fish movement is varied by considering 20, 100 and 250 fishes in the tank and the FPS value is observed under a constant bandwidth of 1 Mbps. The same process is repeated for 25 times and the values are recorded in a table. The whole process of measuring is depicted in the below figure.

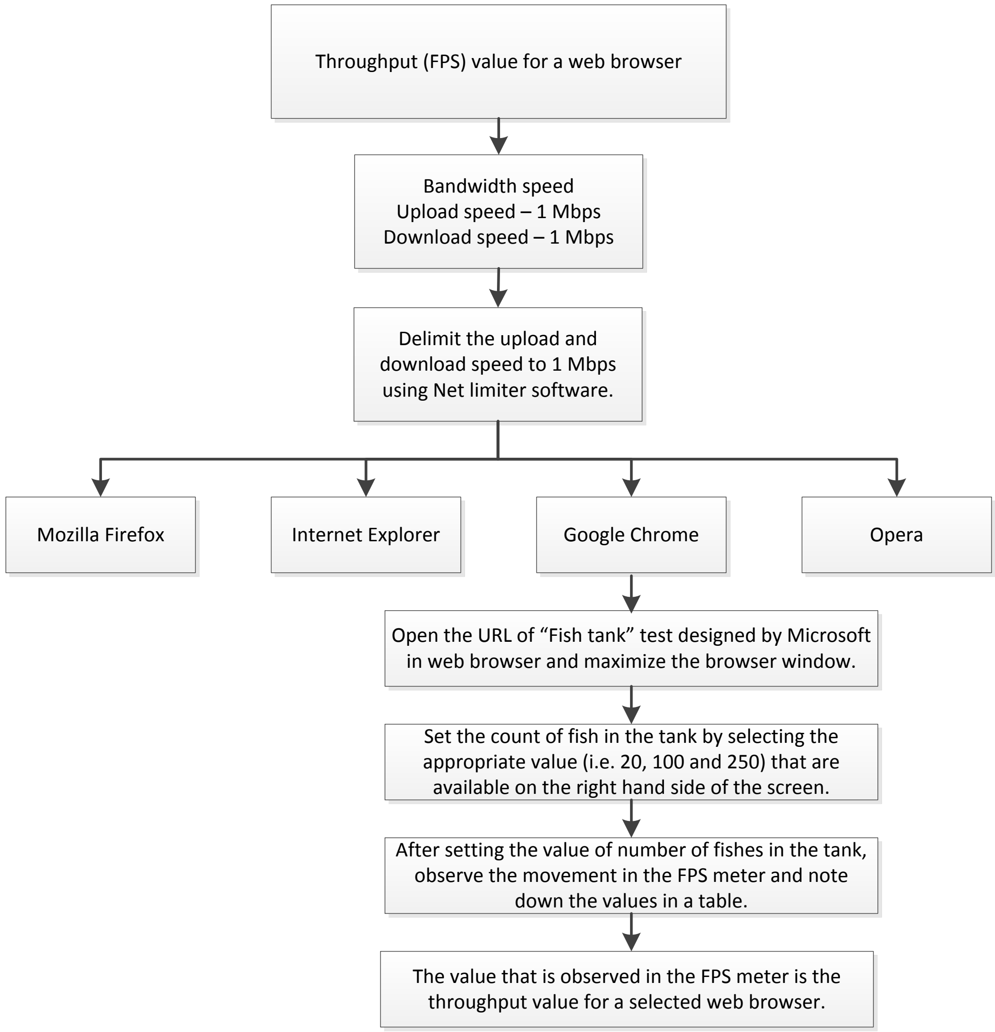

Figure 11 - Procedure to measure throughput of a web browser

5.5.3.4 Memory utilization - Empty browser

In this section memory consumed by an empty browser window is calculated for the four selected web browsers. For measuring the utilized memory by an empty browser, "memory resource monitor" option under the properties of "system task manager" is used. Always remember not to open any other window tabs of the same web browser while calculating values. At the client end a constant bandwidth speed of 1 Mbps is maintained with the help of net limiter tool.

Step by step procedure:

- Now close all the browser windows and open an empty private browser window which should contain only a single tab. Now follow the steps that are discussed in the section 5.4.4.4.
- Mainly observe the "Total memory size occupied" values that are shown on the system screen and note those values in a given table. The whole process is repeated for 25 times and average values are calculated. The size of the utilized memory is measured in Kilo Bytes (KB). The whole process of measuring is depicted in the below figure.

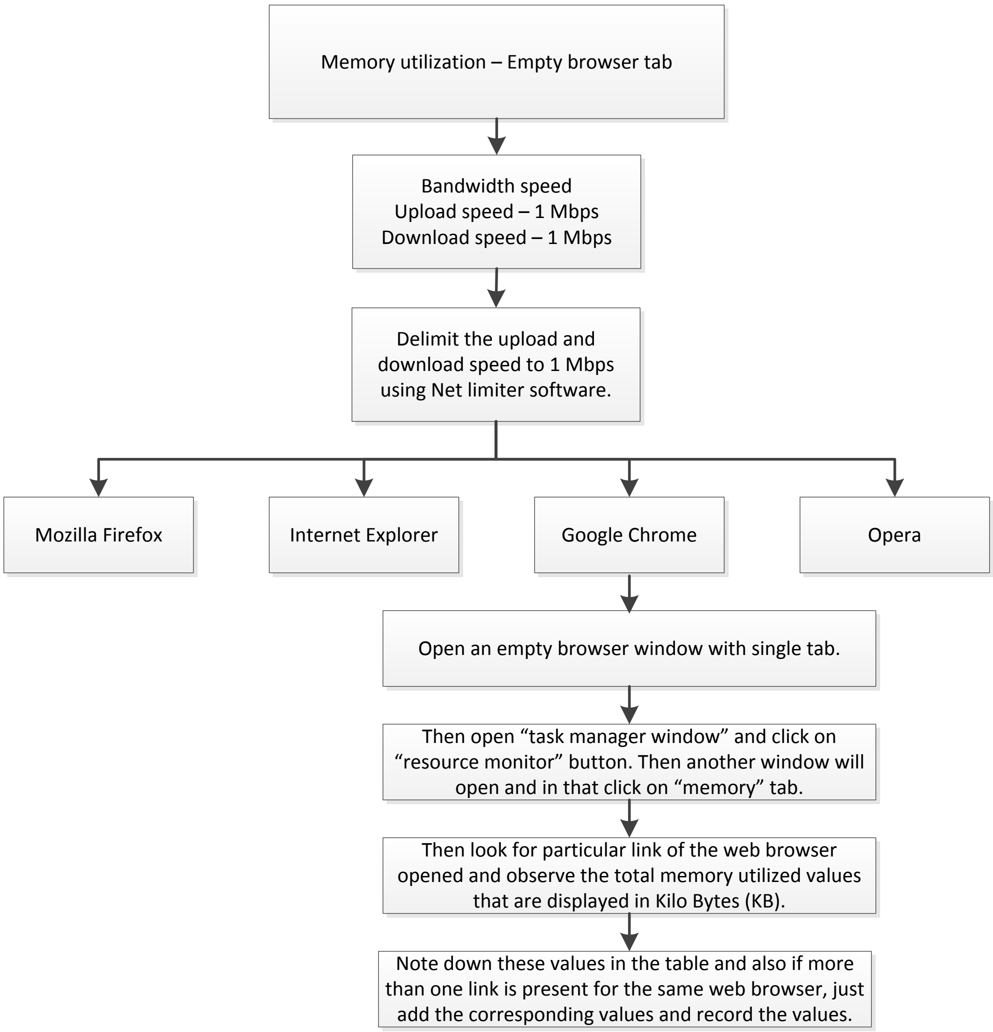

Figure 12 - Procedure to measure memory utilization - Empty web browser

5.5.3.5 Memory utilization - Single tab - HTML page

In this section utilization of memory is calculated when a single web page (i.e. HTML) is opened in a web browser window. For measuring this particular efficiency attribute, "memory resource monitor" option under the properties of "system task manager" is used to calculate the values. Always remember not to open any other window tabs of the same web browser while calculating values. At the client end a constant bandwidth speed of 1 Mbps is maintained with the help of net limiter tool.

Step by step procedure:

- Now close all the browser windows and open an empty private browser window which should contain only a single tab. Then paste the HTML web page URL [45] and wait until the page is loaded completely. Now follow the steps that are discussed in the section 5.4.4.4.
- Mainly observe the "Total memory size occupied" values that are shown on the system screen and note those values in a given table. The whole process is repeated for 25 times and average values are calculated. The size of the utilized memory is measured in Kilo Bytes (KB). The whole process of measuring is depicted in the below figure.

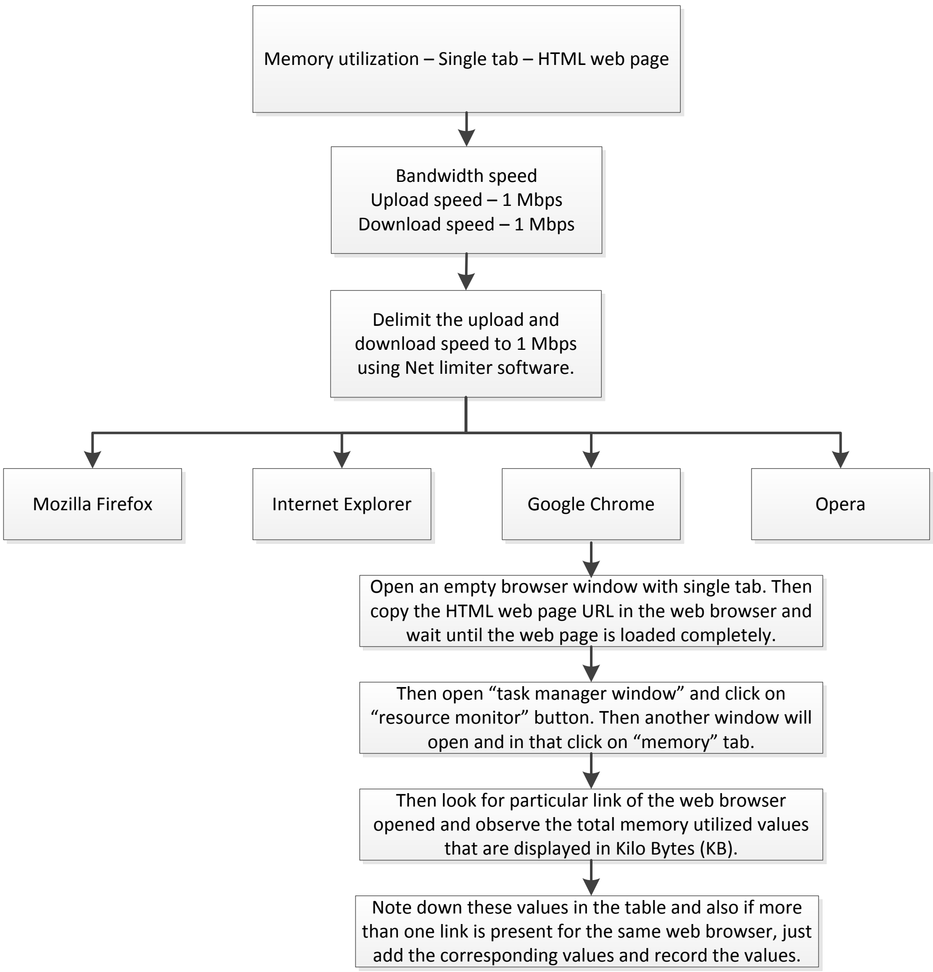

Figure 13 - Procedure to measure memory utilization - Single HTML tab

5.5.3.6 Memory utilization - Multiple tabs

In this section utilization of memory is calculated when multiple (i.e. in this case two tabs are considered) web pages (i.e. two HTML webpages) are opened in a web browser window. For measuring this particular efficiency attribute, “memory resource monitor” option under the properties of “system task manager” is used to calculate the values. Always remember not to open any other window tabs of the same web browser while calculating values. At the client end a constant bandwidth speed of 1 Mbps is maintained with the help of net limiter tool.

Step by step procedure:

- Now close all the browser windows and open an empty private browser window. Then paste the same HTML web page URL [45] twice in two different tabs. And also wait until both the pages are loaded completely. Now follow the steps that are discussed in the section 5.4.4.4.
- Mainly observe the “Total memory size occupied” values that are shown on the system screen and note those values in a given table. The whole process is repeated for 25 times and average values are calculated. The size of the utilized memory is measured in Kilo Bytes (KB). The whole process of measuring is depicted in the below figure.

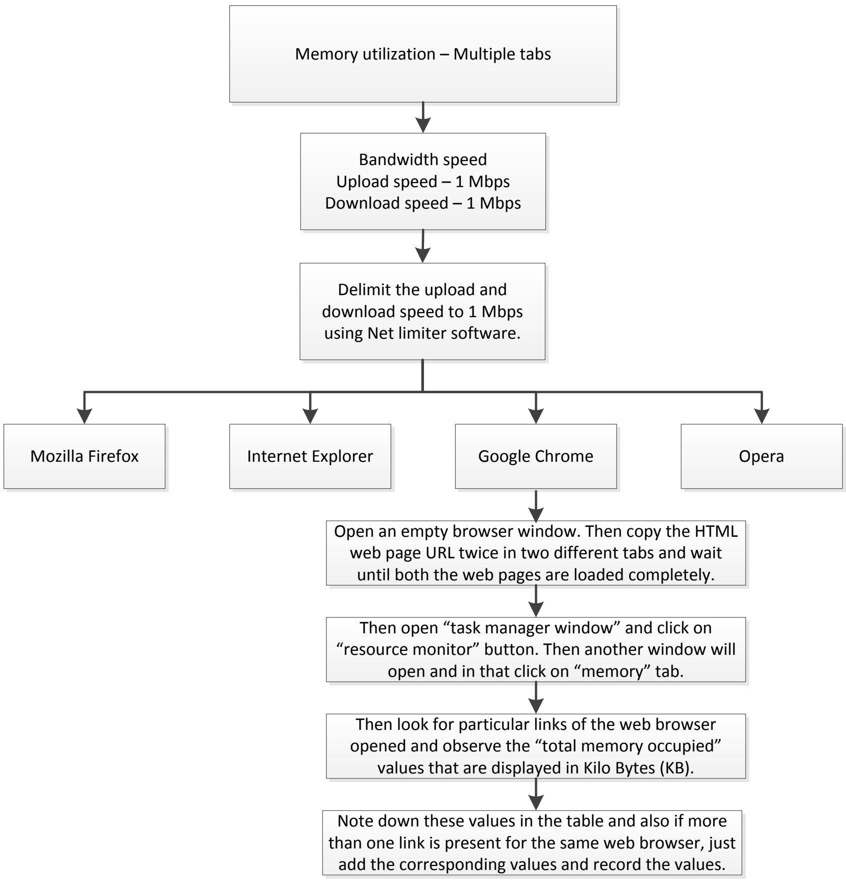

Figure 14 - Procedure to measure memory utilization - Multiple tabs

5.5.4 Execution process implemented for Case B web browsers

In this section execution process is defined for case B (i.e. OSS web browsers). In this case, 4 OSS web browsers were selected to observe the efficiency values. They are Midori, Arora, Seamonkey and Qupzilla. Efficiency values are calculated for the selected measures using tools (i.e. discussed in 5.4.4 section) and 'test subjects' are used to calculate the efficiency values of Case B web browsers. The data in this section is collected based on the observations of the authors.

5.5.4.1 Response time of web browsers - uncached data - HTML page

In order to calculate response time of OSS web browsers for uncached HTML web pages please refer the steps that are discussed in the section 5.5.3.1. The whole process is repeated for 25 times and RT values are calculated at 0.3 and 0.8 Mbps speeds. The results obtained were discussed in 5.6.1.2 section.

5.5.4.2 Response time of web browsers - cached data - HTML page

In order to calculate response time of OSS web browsers for cached HTML web pages please refer the steps that are discussed in the section 5.5.3.2. The whole process is repeated for 25 times and RT values are calculated at 0.3 and 0.8 Mbps speeds. The results obtained were discussed in 5.6.1.2 section.

5.5.4.3 Throughput (FPS)

In order to calculate throughput of OSS web browser, please refer the steps that are discussed in the section 5.5.3.3. The whole process is repeated for 25 times and executed at a constant bandwidth speed of 1 Mbps. Fish size in the tank are varied by 20, 100 and 250 while conducting the test. The results obtained were discussed in 5.6.1.2 section.

5.5.4.4 Memory utilization - Empty browser

In order to calculate the memory utilization of empty OSS web browsers, please refer the steps that are discussed in the section 5.5.3.4. The whole process is repeated for 25 times and executed at a constant bandwidth speed of 1 Mbps. The results obtained were discussed in 5.6.1.2 section.

5.5.4.5 Memory utilization - Single tab - HTML page

In order to calculate the memory utilization of OSS web browsers for single HTML tab opened, please refer the steps that are discussed in the section 5.5.3.5. The whole process is repeated for 25 times and executed at a constant bandwidth speed of 1 Mbps. The results obtained were discussed in 5.6.1.2 section.

5.5.4.6 Memory utilization - Multiple tabs

In order to calculate the memory utilization of OSS web browsers for multiple tabs opened, please refer the steps that are discussed in the section 5.5.3.6. The whole process is repeated for 25 times and executed at a constant bandwidth speed of 1 Mbps. The results obtained were discussed in 5.6.1.2 section.

5.6 Results and Analysis

5.6.1 Experiment results

In this section, results of Case A and Case B are shown for each scenario mentioned in the execution phase. As discussed in earlier sections for every scenario, efficiency values are calculated 25 times and for these results please refer Appendix B. In this section we had mentioned only the average values obtained for each scenario.

5.6.1.1 Efficiency values for Case A web browsers

5.6.1.1.1 Response time values for Uncached HTML web page

Response time values for the web browsers at 0.3 Mbps upload and download speed:

Table 7 - RT of web browsers - Uncached data - HTML page - 0.3 Mbps

Readings / Web browser	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Average values	16.830	12.488	13.791	14.659

Response time values for the web browsers at 0.8 Mbps upload and download speed:

Table 8 - RT of web browsers - Uncached data - HTML page - 0.8 Mbps

Readings / Web browser	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Average values	13.574	10.232	10.890	11.049

5.6.1.1.2 Response time values for Cached HTML web page

Response time values for the web browsers at 0.3 Mbps upload and download speed:

Table 9 - RT of web browsers - Cached data - HTML page - 0.3 Mbps

Readings / Web browser	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Average values	3.923	2.961	3.419	3.835

Response time values for the web browsers at 0.8 Mbps upload and download speed:

Table 10 - RT of web browsers - Cached data - HTML page - 0.8 Mbps

Readings / Web browser	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Average values	2.361	1.580	2.113	2.186

5.6.1.1.3 Throughput values

Throughput value for the web browsers at 1 Mbps upload and download speed:

Table 11 - Throughput values of web browsers - 1 Mbps

Count of the fish in tank	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
20 fishes	37	59	59	58
100 fishes	20	58	58	59
250 fishes	10	58	58	46

5.6.1.1.4 Memory utilization values - Empty web browser

Memory utilization values for the web browsers at 1 Mbps upload and download speed:

Table 12 - Memory utilization of web browsers - Empty browser - 1 Mbps

Type / Memory (in KB)	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Total memory size	72057	42141	79941	55579

5.6.1.1.5 Memory utilization values - Single HTML web page

Memory utilization values for the web browsers at 1 Mbps upload and download speed:

Table 13 - Memory utilization of web browsers - Single HTML tab - 1 Mbps

Type / Memory (in KB)	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Total memory size	92136	67832	101452	73819

5.6.1.1.6 Memory utilization values - Multiple web pages

Memory utilization values for the web browsers at 1 Mbps upload and download speed:

Table 14 - Memory utilization of web browsers - Multiple tabs - 1 Mbps

Type / Memory (in KB)	Mozilla Firefox	Internet Explorer	Google Chrome	Opera
Total memory size	116414	84391	115836	99184

5.6.1.2 Efficiency values for Case B web browsers

5.6.1.2.1 Response time values for Uncached HTML web page

Response time values for the web browsers at 0.3 Mbps upload and download speed:

Table 15 - RT of OSS web browsers - Uncached data - HTML page - 0.3 Mbps

Readings / Web browser	Midori	Sea Monkey	Qupzilla	Arora
Average values	22.170	22.061	24.646	20.694

Response time values for the web browsers at 0.8 Mbps upload and download speed:

Table 16 - RT of OSS web browsers - Uncached data - HTML page - 0.8 Mbps

Readings / Web browser	Midori	Sea Monkey	Qupzilla	Arora
Average values	18.856	19.925	21.443	18.935

5.6.1.2.2 Response time values for Cached HTML web page

Response time values for the web browsers at 0.3 Mbps upload and download speed:

Table 17 - RT of OSS web browsers - Cached data - HTML page - 0.3 Mbps

Readings / Web browser	Midori	Sea Monkey	Qupzilla	Arora
Average values	4.266	4.824	5.146	4.224

Response time values for the web browsers at 0.8 Mbps upload and download speed:

Table 18 - RT of OSS web browsers - Cached data - HTML page - 0.8 Mbps

Readings / Web browser	Midori	Sea Monkey	Qupzilla	Arora
Average values	3.220	3.836	3.949	3.578

5.6.1.2.3 Throughput values

Throughput value for the web browsers at 1 Mbps upload and download speed:

Table 19 - Throughput values of OSS web browsers - 1 Mbps

Count of the fish in tank	Midori	Sea Monkey	Qupzilla	Arora
20 fishes	43	14	27	56
100 fishes	21	12	20	52
250 fishes	19	09	16	49

5.6.1.2.4 Memory utilization values - Empty web browser

Memory utilization values for the web browsers at 1 Mbps upload and download speed:

Table 20 - Memory utilization of OSS web browsers - Empty browser - 1 Mbps

Type / Memory (in KB)	Midori	Sea Monkey	Qupzilla	Arora
Total memory size	87337	88270	59572	94434

5.6.1.2.5 Memory utilization values - Single HTML web page

Memory utilization values for the web browsers at 1 Mbps upload and download speed:

Table 21 - Memory utilization of OSS web browsers - Single HTML tab - 1 Mbps

Type / Memory (in KB)	Midori	Sea Monkey	Qupzilla	Arora
Total memory size	106266	106861	84977	114890

5.6.1.2.6 Memory utilization values - Multiple web pages

Memory utilization values for the web browsers at 1 Mbps upload and download speed:

Table 22 - Memory utilization of OSS web browsers - Multiple tabs - 1 Mbps

Type / Memory (in KB)	Midori	Sea Monkey	Qupzilla	Arora
Total memory size	118033	111950	104337	131160

6 DISCUSSION

6.1 Proposing EBC guidelines

Based on the results gathered in section 5.6.1 (i.e. Case A results), an Efficiency Baseline Criteria (EBC) is been proposed in this section. From the above data, least values are only considered for each efficiency attribute, irrespective of the type of web browser and a standard set of values are formulated for each efficiency quality attribute that are related to web browsers. The main motivation behind formulating EBC values are discussed in this section.

We have some online test like Test Drive [41], Speed Battle [34], Peace Keeper [49], Acid test [46] etc. to test the web browser performance but they are many differences between their process of evaluating and our process. Many of the above tests presented do not consider bandwidth speeds for actual calculation of values. These values tend to change with variation in bandwidth speeds and hence it is unpredictable to use these values for comparing efficiency of web browsers. But in our experiment, we had used net limiter tool to control the bandwidth speed at client end and also we had used two different bandwidth speeds (i.e. 0.3 and 0.8 Mbps) speeds for calculating EBC values for response time. Therefore our EBC results generated are more effective than other online tests.

The main problem with these online tests is the processes of calculating (i.e. evaluation procedure) the browser efficiency values are not available. Hence the results are unreliable and unscientific. But for deriving EBC values, we had clearly described the process of execution and followed a scientific approach by following the guidelines given by Wholin et al and also used proper statistical analysis methods to analyze the obtained results. Another problem with these online tests is server execution time is not considered, as a result response time tend to vary with change in server load. In our process of generating EBC values, we had considered server execution speed in to account.

Another concern with these online tests is browser compatibility. In this research work we are evaluating efficiency standards of OSS web browsers with other standard web browsers. It is observed that some tests are only compatible with some web browsers and not compatible with others. For example Test Drive [41] test is only compatible with IE but not with other browsers. But one can derive efficiency standard of any web browser with the methodology that has been used to derive EBC values. Hence our evaluation process does not have any web browser compatibility issues and can be used for analyzing efficiency standards of any web browser.

Another concern with these online tests are they do not consider the background activities that are been performed inside the web browser. Sometimes web browser may be updating some of the plugins internally or sometimes some plugins may be inactive state in one browser and active in another browser. Then the efficiency results obtained from these tests are not suitable for comparison among web browsers. But while calculating EBC values care is taken such that there are no internal web browser activities are going on in the system, hence making EBC results more reliable for comparison purpose. Another important point that we had considered while calculating EBC values is web page size. We had maintained webpage size constant throughout the experiment process such that it does not impact the throughput and memory utilization values.

Therefore from the above motivations, we infer that there is a need for the EBC standards. These standards can be used by any web browser developer (i.e. especially OSS developers) to test the efficiency attributes of their products. These results are more helpful to OSS community, since there are no specific standards that are been followed while developing a product.

In our case, we had derived values for EBC from the results that are generated in the section 5.6.1 (i.e. Case A results). In section 6.2, we had provided EBC values for each scenario that is considered during execution of experiment.

6.2 Efficiency Baseline Criteria values

6.2.1 EBC values for Response time

In this section EBC values that are obtained for response time scenarios are discussed. EBC values are derived at two different bandwidth speeds (i.e. 0.3 and 0.8 Mbps). From the below table, web browser which is having least value for each scenario is selected as EBC value. The values that are highlighted in "red" color in the below tables are considered for EBC values.

Table 23 - EBC values - Response time

Efficiency attributes	Bandwidth speeds	Mozilla Firefox	Internet Explorer	Google Chrome	Opera	EBC values
Response time - Uncached HTML page.	0.3	16.830	12.488	13.791	14.659	12.488
	0.8	13.574	10.232	10.890	11.049	10.232
Response time - Cached HTML page.	0.3	3.923	2.961	3.419	3.835	2.961
	0.8	2.361	1.580	2.113	2.186	1.580

6.2.2 EBC values for Throughput

In this section EBC values that are obtained for throughput are discussed. EBC values are derived at 1 Mbps constant bandwidth speeds, but the number of fishes in the tank are varied in numbers (i.e. 20, 100 and 250 fishes). From the below table, web browser which is having highest value for each scenario is selected as EBC value. The values that are highlighted in "red" color in the below tables are considered for EBC values.

Table 24 - EBC values - Throughput

Efficiency attributes	Number of fishes	Mozilla Firefox	Internet Explorer	Google Chrome	Opera	EBC values
Throughput	20	37	59	59	58	59
	100	20	58	58	59	59
	250	10	58	58	46	58

6.2.3 EBC values for Memory utilization

In this section, EBC values that are obtained for 'total memory size' are discussed. EBC values are derived at 1 Mbps constant bandwidth speed. In this case 3 scenarios are considered, they are empty browser, single tab - HTML page opened and multiple tabs opened. From the below table, web browser which is having least value for each scenario is selected as EBC value. The values that are highlighted in "red" color in the below tables are considered for EBC values.

Table 25 - EBC values - Memory utilization

Efficiency attributes	Mozilla Firefox	Internet Explorer	Google Chrome	Opera	EBC values
Memory Utilization - Empty browser.	72057	42141	79941	55579	42141
Memory Utilization - Single HTML tab.	92136	67832	101452	73819	67832
Memory Utilization - Multiple tabs.	116414	84391	115836	99184	84391

7 EVALUATING EFFICIENCY OF OSS WEB BROWSERS

Now in this section, efficiency standards of OSS web browsers are evaluated. The whole process is carried out by comparing Case B results with Case A results (i.e. Hypothesis testing) and further we had also compared EBC values with Case B results, which is presented in section 7.1. For this a research question is formulated in the research methodology section and it is answered in this section. The main motivation behind formulating this research question is to observe and evaluate the efficiency of current OSS web browsers in the market, when compared to the set of Case A web browsers.

RQ 3: How to evaluate efficiency quality attributes in OSS Web browsers with the help of experiment results and proposed Efficiency Baseline Criteria (EBC)?

7.1 Hypothesis testing

The data obtained from the results of the experiment are analyzed using the design principle "One factor with two treatments". Normality test results and the design principle are useful while selecting parametric or non-parametric tests [7]. Normality of the data identified for each case was verified using one sample Kolmogorov - Smirnov test (i.e. K-S test) and the level of significance is set to 0.05. In our experiment, we had two sets of data for each scenario. Initially we had performed one sample K-S test for a combined test set of 200 sample size (i.e. both Case A and Case B results) and based on the p-values obtained, we came to an agreement on the selection of parametric or non-parametric tests [7]. "R" tool [37] [47] is used to calculate the values for K-S test. The procedure for selection of parametric or non-parametric test based on the p-values obtained after performing K-S test are discussed in the below table.

Table 26 - Criteria for selecting parametric or non-parametric test based on K-S test results

Conditions	P- value obtained for combined data set	Selection of parametric or non-parametric tests
Condition 1	P >= 0.05	Parametric test
Condition 2	P < 0.05	Non - Parametric test

Normality test results can be used to select parametric or non-parametric test in order to test the hypothesis which was framed in the earlier sections. In this experiment, we plan to perform two sample T-test (i.e. parametric test) for the scenarios where data is normal and Mann Whitney U-test (i.e. non-parametric test) was performed for the scenarios, where the data is not normal. For conducting two sample T-test, we had used R tool [37] [47] to calculate "P-values" and for Mann Whitney U-test, we had used an excel spreadsheet [48] to calculate "Z- values". We need to consider normalized Z scores, while conducting Mann Whitney U-test because the data sample size is equal to 200 in each case (i.e. N_a=100 and

N_b=100). In the below sections, hypothesis testing results were calculated for each scenario. All values that are used for hypothesis testing calculations are considered from Appendix B.

7.1.1.1 Hypothesis testing for Response time values

7.1.1.1.1 Hypothesis selected

Null hypothesis (H_{1_0}):

Response time values of Case B (i.e. OSS) web browsers are equivalent to the Case A web browsers.

Alternative hypothesis (H_{1_1}):

Response time values of Case B (i.e. OSS) web browsers are not equivalent to the Case A web browsers.

7.1.1.1.2 Performing normality test

Table 27 - Normality test results for Response time - All scenarios

Normality test of web browsers	P-values (K-S test)	Result (at level of 0.05)	Overall result	Method selected
Uncached HTML (0.3 Mbps)	0.0348	P < 0.05	Non parametric test	Mann Whitney - U test
Cached HTML (0.3 Mbps)	0.0411	P < 0.05	Non parametric test	Mann Whitney - U test
Uncached HTML (0.8 Mbps)	0.0056	P < 0.05	Non parametric test	Mann Whitney - U test
Cached HTML (0.8 Mbps)	0.0339	P < 0.05	Non parametric test	Mann Whitney - U test

7.1.1.1.3 Performing Non-Parametric tests

From the above results it is evident that for all the scenarios the results are not normal and hence non parametric test (i.e. Mann Whitney U-test) was performed in order to test the hypothesis. The results are shown in the below table:

Table 28 - Non - Parametric test result for Response time scenarios

Scenario	|Z| -value (U - test)	Significant (Z) value	Overall result	Hypothesis result
Uncached HTML (0.3 Mbps)	-12.215	1.96	|Z| > 1.96	Reject Null hypothesis
Cached HTML (0.3 Mbps)	-8.301	1.96	|Z| > 1.96	Reject Null hypothesis
Uncached HTML (0.8 Mbps)	-12.151	1.96	|Z| > 1.96	Reject Null hypothesis
Cached HTML (0.8 Mbps)	-12.183	1.96	|Z| > 1.96	Reject Null hypothesis

7.1.1.1.4 Result of hypothesis testing

For all the four scenarios, Mann Whitney U- test was performed and it is observed that the normalized Z scores for each scenario is greater than 1.96 [22] (i.e. $|Z| > 1.96$). Hence for these scenarios null hypothesis is rejected. Finally it is proven that the *response time values of OSS web browsers are not equivalent to Case A web browsers* for all the four scenarios.

7.1.1.1.5 Comparison with EBC values

In this section, Case B results are compared with EBC values and are presented in the below table. From the below table also it is proven that response time values of OSS web browsers are not equivalent to EBC values for all the four scenarios.

Table 29 - Comparing Case B results with EBC values

Scenario	Case B web browsers				EBC value
	Midori	Sea Monkey	Qupzilla	Arora	
Uncached HTML (0.3 Mbps)	22.170	22.061	24.646	20.694	12.488
Cached HTML (0.3 Mbps)	4.266	4.824	5.146	4.224	2.961
Uncached HTML (0.8 Mbps)	18.856	19.925	21.443	18.935	10.232
Cached HTML (0.8 Mbps)	3.220	3.836	3.949	3.578	1.580

7.1.1.2 Hypothesis testing for Throughput values

7.1.1.2.1 Hypothesis selected

Null hypothesis ($H2_0$):

Throughput values of Case B (i.e. OSS) web browsers are equivalent to the Case A web browsers.

Alternative hypothesis ($H2_1$):

Throughput values of Case B (i.e. OSS) web browsers are not equivalent to the Case A web browsers.

7.1.1.2.2 Performing normality test

Table 30 - Normality test results for Throughput

Normality test of web browsers	P-values (K-S test)	Result (at level of 0.05)	Overall result	Method selected
20 Fish (1Mbps)	0.000006	P < 0.05	Non - Parametric test	Mann Whitney - U test
100 Fish (1Mbps)	0.000001	P < 0.05	Non - Parametric test	Mann Whitney - U test
250 Fish (1Mbps)	0.000001	P < 0.05	Non - Parametric test	Mann Whitney - U test

7.1.1.2.3 Performing Non-Parametric test

From the above results it is evident that for all the scenarios the results are not normal and hence non parametric test (i.e. Mann Whitney U-test) was performed in order to test the hypothesis. The results are shown in the below table:

Table 31 - Non - Parametric test result for Throughput

Scenario	|Z| -value (U - test)	Significant (Z) value	Overall result	Hypothesis result
20 Fish (1Mbps)	-8.904	1.96	|Z| > 1.96	Reject Null hypothesis
100 Fish (1Mbps)	-8.486	1.96	|Z| > 1.96	Reject Null hypothesis
250 Fish (1Mbps)	-5.911	1.96	|Z| > 1.96	Reject Null hypothesis

7.1.1.2.4 Result of hypothesis testing

Initially Mann Whitney U- test was performed for the above scenario and observed that the normalized Z score is greater than 1.96 [22] (i.e. |Z| > 1.96). Hence for the above all the scenarios null hypothesis is rejected. Finally it is proven that the *throughput values of OSS web browsers are not equivalent to Case A web browsers*.

7.1.1.2.5 Comparison with EBC values

In this section, Case B results are compared with EBC values and are presented in the below table. From the below table also it is proven that throughput values of OSS web browsers are not equivalent to EBC values for all the three scenarios.

Table 32 - Comparing Case B results with EBC values

Scenario	Case B web browsers				EBC value
	Midori	Sea Monkey	Qupzilla	Arora	
20 Fish (1Mbps)	43	14	27	56	59
100 Fish (1Mbps)	21	12	20	52	59
250 Fish (1Mbps)	19	9	16	49	58

7.1.1.3 Hypothesis testing for Memory utilization values

7.1.1.3.1 Hypothesis selected

Null hypothesis (H_{3_0}):

Memory utilization values of Case B (i.e. OSS) web browsers are equivalent to the Case A web browsers.

Alternative hypothesis (H_{3_1}):

Memory utilization values of Case B (i.e. OSS) web browsers are not equivalent to the Case A web browsers.

7.1.1.3.2 Performing normality test

Table 33 - Normality test results for Memory utilization - All scenarios

Normality test of web browsers	P-values (K-S test)	Result (at level of 0.05)	Overall result	Method selected
Empty browser	0.00021	P < 0.05	Non - Parametric test	Mann Whitney - U test
Single HTML tab	0.00050	P < 0.05	Non - Parametric test	Mann Whitney - U test
Multiple HTML tabs	0.00408	P < 0.05	Non - Parametric test	Mann Whitney - U test

7.1.1.3.3 Performing Non-Parametric test

Table 34 - Non - Parametric test result for Memory Utilization for all scenarios

Scenario	|Z| -value (U - test)	Significant (Z) value	Overall result	Hypothesis result
Empty browser	-9.102	1.96	|Z| > 1.96	Reject Null hypothesis
Single HTML tab	-9.130	1.96	|Z| > 1.96	Reject Null hypothesis
Multiple HTML tabs	-5.419	1.96	|Z| > 1.96	Reject Null hypothesis

7.1.1.3.4 Result of hypothesis testing

As the data is not normal in all the three scenarios, we had performed Mann Whitney U- test (i.e. Non-parametric test). From the above results, we had observed that the normalized Z scores are greater than its significant value 1.96 [22] (i.e. $|Z| > 1.96$). As a result we had rejected null hypothesis, hence it is proven that *Memory utilization values of OSS web browsers are not equivalent to the Case A web browsers* for all three scenarios.

7.1.1.3.5 Comparison with EBC values

In this section, Case B results are compared with EBC values and are presented in the below table. From the below table also it is proven that memory utilization values of OSS web browsers are not equivalent to EBC values for all the three scenarios.

Table 35 - Comparing Case B results with EBC values

Scenario	Case B web browsers				EBC value
	Midori	Sea Monkey	Qupzilla	Arora	
Empty browser	87337	88270	59572	94434	42141
Single HTML tab	106266	106861	84977	114890	67832
Multiple HTML tabs	118033	111950	104337	131160	84391

8 VALIDITY THREATS

In general, validity of research is concern about, how the conclusions or results might go wrong. By discussing validity threats, one can assess the merits and demerits of the study with respect to validity of the outcome.The term validity threat is defined as *"It describes a particular way in which one might be wrong while performing the research"* [35]. In this section, we discuss validity threats related to Literature Review and experiment. Threats can be mainly classified in to four categories [35] [36] [7]. They are

- **Internal Validity: "***This validity concerns whether the treatment actually causes the outcome*".
- **External Validity: "***This validity concerns whether one can generalize the results outside the scope of the study".*
- **Construct Validity: "***This threat concerns on relation between the theory behind the treatment and outcome. Even though if we maintain a causal relationship between treatment and outcome, there can be a chance that treatment may not correspond to the cause and outcome may not correspond to the effect*".
- **Conclusion Validity: "***This threat concerns the treatment is really related to the actual outcome".*

8.1 Literature Review

Identification of primary studies

The main motivation behind conducting LR is to extract as many related primary studies from the databases. In any LR there is a possible threat of missing some relevant studies (i.e. relevant articles may not be retrieved during automated search). Therefore to mitigate this general threat, we had initially performed manual search in the selected journals and conferences in order not to miss any articles that are related to our study.

Reliability of the collected data

Another threat that is to be considered is the information collected from the retrieved articles is reliable or not. In order to confirm that all information gathered in the study is reliable and correct, we had included only peer reviewed articles in the study and excluded gray literature (i.e. non-peer reviewed articles) even though the information in it is related to our study.

Agreement on selection of primary studies

While selection of primary studies there may be some agreement and disagreement between the two researchers. Therefore there is a threat of not selecting primary studies that are most relevant to the study. In order to prevent this threat both researchers had performed article selection criteria separately. Finally after a thorough discussion both the authors came to a conclusion on the set of primary studies that are to be included in the study and the kappa values of 0.84, suggest us that there is almost perfect agreement on the articles selected by two researchers.

8.2 Experiment

Variations in bandwidth speed

Generally while calculating response time of web browsers many external factors will impact the end results. One of the factors is response time values will tend to change as the speed of the bandwidth varies. Therefore variations in bandwidth can be considered as important threat in our study. This threat arises only during calculation of response time values and not in other cases like memory utilization because these outcomes are not based on the bandwidth. In order to overcome this threat we need to shape or control the bandwidth speeds at either end (i.e. client end or server end). But it is impossible to control the bandwidth speeds at server end, we had chosen an option of controlling it at client end. This can be achieved with the help of Net limiter tool, which shapes or reduces the speed of bandwidth to required values at client end. This tool can only delimit the speed but cannot boost the bandwidth speed. There are many other tools available online which are related to delimiting of bandwidth speed, therefore it is the choice of user, when it comes to the selection of these tools.

Variations in web page size

It is known fact that the values of response time and memory utilization depends on size of the page. For example consider two web pages with 1000 KB and 10MB in size. The response time for 1000 KB web page will be low when compared to 10 MB, since 10 MB page takes more time to load in the web browser and similarly with the case memory utilization, more the size of web page more memory is consumed because more elements will be loaded in to memory. Therefore variations in page sizes can cause a serious threat to our study. In order to eliminate this threat, we had considered a stable website where there are no updates frequently done to it and the page size is always constant while conducting experiment. This is can be checked by using any web traffic analysis tools that are available online for free and confirm that there are no updates done to the page size and other way is to check the size of the page from time to time.

Variations in server execution speed

It is observed that the response time and throughput values are affected by the server execution speed because, the RT & TH values depend on the server load during execution of our experiment. For example if the particular server has more user requests to handle then the server execution speed will be low, while in contrast if there are no requests in the server then the execution speed of server will be high. Hence server execution speed will be a serious threat to our experiment. We tried to mitigate this threat by running the experiment 25 times at different time intervals, such that we can overcome this threat and nullify the effect of server execution speed in our experiment results.

9 CONCLUSION

In this thesis, initially we had studied the efficiency quality attribute in different software products and finally identified 6 software efficiency measures like Response time, Throughput, Turnaround time, Memory utilization, Input or Output utilization and Transmission utilization. The primary studies for the identified efficiency measures are extracted by using the literature review (LR). As a part of LR, ISO 9126 model is studied with emphasis on efficiency quality attribute. The output of LR is that a total of 6 scenarios are identified that are sufficient to analyze efficiency quality attribute in web browsers. They are RT - uncached webpage, RT - cached webpage, Throughput, MU - empty web browser, MU - single HTML webpage, MU - multiple HTML webpage. Later an experiment was organized in order to calculate efficiency values for these identified measures in LR. Further web browsers are categorized as Case A and Case B, where Case A contains CSS and proprietary web browsers and Case B contains OSS web browsers. Efficiency values are calculated for Case A web browsers and then EBC values are proposed based on the values of Case A. These EBC values can be used by OSSD community in order to evaluate the efficiency standards of their web browsers. Later Case B results are calculated and then hypothesis testing was performed. Initially K-S test was performed and results suggest choosing non-parametric test (i.e. Mann Whitney U-test). Later Mann Whitney U-test was performed for all the scenarios and the normalized Z scores are more than 1.96, further suggested rejecting null hypothesis for all the cases. Also EBC values are compared with Case B results and these also suggest us that efficiency standard of OSS web browsers are not up to the mark when compared to other web browsers. Finally we conclude by saying that researchers need to focus more on increasing efficiency standards of OSS web browsers. One of the facts that we conclude from this experiment is that the efficiency standards of OSS web browsers are not equivalent when compared to other web browser categories.

10 FUTURE WORK

In this study we mainly concentrated on efficiency standards of different web browsers and finally proposed EBC values for OSSD community. In the same way research can be performed to other software quality attributes. For example researchers can study on usability, maintainability, portability, security, functionality and reliability attributes and can propose baseline criteria for different software products. As a result, these research works can be used by OSSD community to test their products and proper measures can be undertaken in order to improve the quality standards of the product.

In our thesis, we had studied efficiency quality attribute and observed the efficiency standards of OSS web browsers, similarly various other OSS products like word processing tools, video players, data management tools etc. can also be considered and research can be performed in order to study and improve the efficiency standards of these selected products.

In our thesis, we had proposed an Efficiency Baseline Criteria, but these values are not industrially validated and accepted. Therefore further research work can be performed in this area, where the EBC results can be evaluated by conducting interviews with the industry people and collect their feedback. Finally a set of guidelines can be proposed to OSSD community by improving our EBC results.

11 REFERENCES

[1] R. Roets, M. Minnaar and K.Wright. Open Source: Towards Successful Systems Development Projects in Developing Countries. In Proceedings of the 9th International Conference on Social Implications of Computers in Developing Countries.São Paulo, Brazil, May 2007.

[2] H. Hedberg, N. Iivari, M. Rajanen and L. Harjumaa. Assuring Quality and Usability in Open Source Software Development. First International Workshop on Emerging Trends on FLOSS Research and Development (FLOSS '07), IEEE, vol. no 2, pp. 20-26, May 2007.

[3] I. Samoladas and I. Stamelos. Assessing Free/Open Source Software Quality. Department of Informatics, Aristotle University of Thessaloniki, Greece, 2003.

[4] C. DiBona , S. Ockman , M. Stone , B. Behlendorf , S. Bradner , J. Hamerly, K. McKusick , T. O'Reilly , T. Paquin , B. Perens , E. Raymond , R. Stallman , M. Tiemann , L. Torvalds , P. Vixie , L. Wall , B. Young. Open Sources: Voices from the Open Source Revolution. O'Reilly Media. first edition, Jan 1999.

[5] C. M. Schweik and A. Semenov. The institutional design of open source programming: Implications for addressing complex public policy and management problems, 2003 [Online] Available:
http://www.firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/%20view/1019/2426.
[Accessed: 13-May-2012].

[6] N. Jorgensen. Putting it All in the Trunk Incremental Software Development in the FreeBSD Open Source Project. Information Systems Journal, 11(4): 321-336, 2001.

[7] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B.,Wesslén, A. "Experimentation in software engineering", Kluwer Academic Publishers Norwell, MA, USA., 2000.

[8] H. W. Jung, S. G. Kim and C .S. Chung. Measuring software product quality: a survey of ISO/IEC 9126. IEEE Software , 21(5): 88- 92, 2004.

[9] H. Al-Kilidar, K. Cox and B. Kitchenham. The use and usefulness of the ISO/IEC 9126 quality standard. In International Symposium on Empirical Software Engineering. IEEE, 2005.

[10] J. Feller and B. Fitzgerald. A Framework analysis of the Open Source Software Development paradigm. In Proceedings of the International Conference on Information Systems (ICIS).University College Cork, Brisbane, December 2000.

[11] A. Fuggetta. Controversy Corner: Open source software– an evaluation. Journal of systems and software, 66(1):77-90, 2003.

[12] B. Fitzgerald and P.J. Ågerfalk. The Mysteries of Open Source Software: Black and White and Red All Over?. In Proceedings of the 38 th Hawaii International Conference On System Sciences. IEEE, 2005.

[13] R. T. Selvi, N. Sudha and V. Balasubramanian. Performance Analysis of Proprietary and Non-Proprietary Software. In Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS `08), 2008.

[14] Y. Ye and K. Kishida. Toward an Understanding of the Motivation of Open Source Software Developers. In Proceedings of the International Conference on Software Engineering (ICSE `03).IEEE, 2003.

[15] S.T. Lee, H. Kim and S.Gupta. Measuring open source software success. Omega, 37: 426-438, 2009.

[16] A.Khanjani and R.Sulaiman. The Process of Quality Assurance under Open Source Software Development. In Proceedings of the International Symposium on Computer and Informatics . IEEE, 2004.

[17] D. Taibi, L. Lavazza and S.Morasca. OpenBQR: a framework for the assessment of OSS. In Proceedings of the International Conference on Open Source Systems (OSS `07), 2007.

[18] M. Aberdour. Achieving Quality in Open Source Software. IEEE computer society, 24(1): 58-64, 2007.

[19] D. Spinellis and V. Giannikas. Organizational adoption of open source software. Journal of Systems and Software, 85(3): 666-682, 2012.

[20] S. Liang and C. Lien. Selecting the optimal ERP software by combining the ISO 9126 standard and fuzzy AHP approach. Journal of Contemporary Management Research, 3(1):23–44, 2007.

[21] J.McCall. Quality Factors. Encyclopedia of Software Engineering. Vol. 2, New York, Wiley, 1994.

[22] Statistical distributions [Online] Available:http://www.umm.uni-heidelberg.de/inst/zmf/affymetrix/pdf/statistik.pdf [Accessed: 25-Dec-2014].

[23] W. Afzal, R. Torkar, and R. Feldt. A systematic review for search-based testing for non-functional system properties. Information and Software Technology, 51: 957-976, 2009.

[24] J. L. Fleiss. Measuring nominal scale agreement among many raters. Psychological Bulletin. 76(5): 378, 1971.

[25] A. J. Viera and J.M. Garrett. Understanding Interobserver Agreement: The Kappa Statistic. Family Medicine, 37(5): 360–363, May 2005.

[26] B. Kitchenham and S.L. Pfleeger. Software quality: the elusive target. IEEE Software,13(1):12-21, 1996.

[27] S.A.Shaikh and A. Cerone. Towards a metric for Open Source Software Quality. In Proceedings of the 3rd International Workshop on Foundations and Techniques for Open Source Software Certification. Electronic Communications of the EASST, 20, 2009.

[28] D.Spinellis, G.Gousios, V. Karakoidas, P.Louridas, P.J. Adams, I. Samoladas and I. Stamelos. Evaluating the Quality of Open Source Software. Electronic Notes in Theoretical Computer Science, 233, 5-28, 2009.

[29] V. del Bianco, L. Lavazza, S. Morasca, D. Taibi and D. Tosi. An Investigation of the Users' Perception of OSS Quality. In Proceedings of the 6th International Conference on Open Source Systems (OSS`10), 2010.

[30] G. Zhang, H.Ye and Y.Lin. Quality Attributes Assessment for Feature-Based Product Configuration in Software Product Line. In Proceedings of the 17th Asia Pacific Software Engineering Conference (APSEC). 137–146, 2010.

[31] K. Ven, J. Verelst and H. Mannaert. Should you adopt open source software?. Software, IEEE, 25(3): 54-59, 2008.

[32] A. Porter, C. Yilmaz, A. Memon, A.S. Krishna, D. C. Schmidt and A. Gokhale. Techniques and Processes for Improving the Quality and Performance of Open-Source Software. Software Process—Improvement and Practice, 11(2):163-176, 2006.

[33] J. Liu , L. Wang , P. Sun and H.Ni. A Distributed Sharing Model of the Browser Cache. In Proceedings of the International Conference on Computational Intelligence and Software Engineering (CiSE`09). IEEE, 1-4, 2009.

[34] Speed Battle tool. [Online] Available: http://www.speed-battle.com/speedtest_e.php [Accessed: 31-Dec-2014].

[35] R.Feldt and A. Magazinius. Validity Threats in Empirical Software Engineering Research - An Initial Survey. In Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering (SEKE `10), 2010.

[36] H. K. Wright, M. Kim and D.E. Perry. Validity Concerns in Software Engineering Research. In Proceedings of the Workshop on the Future of Software Engineering Research, November, 2010.

[37] Valentin Wimmer, Statistics and Data analysis using R. Lehrstuhl fur Panzenzuchtung, TUM. December 2010.

[38] M. Jørgensen. Software quality measurement. Department of Informatics, 30:907–912, 1999.

[39] K. Crowston, K.WEI, J. HOWISON and A. WIGGINS. Free/Libre open-source software development: What we know and what we do not know. ACM Computing Surveys (CSUR), 44(2):7, 2012.

[40] Numion stop watch. [Online] Available: http://www.numion.com/stopwatch/ [Accessed: 05-Nov-2012].

[41] Test drive tool. [Online] Available: ie.microsoft.com/testdrive/Performance/ParticleAcceleration [Accessed: 25-Dec-2014]

[42] IE fish tank test link [Online] Available: http://ie.microsoft.com/testdrive/performance/fishietank/ [Accessed: 08-Nov-2012].

[43] Net Limiter Tool [Online] Available: http://www.netlimiter.com/ [Accessed: 11-Nov-2012].

[44] Speedtest.net [Online] Available: http://speedtest.net/ [Accessed: 11-Nov-2012].

[45] HTML Link of the website [Online] Available: http://www.bth.se/eng/prospectivestudents.nsf/pages/bthlife [Accessed: 25-Dec-2014].

[46] Acid Test tool. [Online] Available: http://acid3.acidtests.org/ [Accessed: 31-Dec-2014].

[47] R tool download link of the website [Online] http://www.statmethods.net/ [Accessed: 08-Sep-2014].

[48] Mann Whitney U-test excel spreadsheet link [Online] https://twopaces.com/ESC2007datasets/Mann-Whitney_U_Test.xls [Accessed: 08-Sep-2014].

[49] Peacekeeper tool. [Online] Available: http://peacekeeper.futuremark.com/ [Accessed: 31-Dec-2014].

[50] H.Kim, H. Jo and J. Lee. XHive: Efficient Cooperative Caching for Virtual Machines. IEEE Transactions on Computers,60(1): 106-119, 2011.

[51] X. Lei, K. Xue and Y. Jia. A comprehensive metric to evaluating software aging. In Proceedings of the International Conference on Computer Application and System Modeling (ICCASM` 10). IEEE, 2010.

[52] J. Choi, S.H.Noh, S. L. Min, E. Ha and Y. Cho. Design, implementation, and performance evaluation of a detection-based adaptive block replacement scheme. IEEE Transactions on Computers, 51(7):793-800, Jul 2002.

[53] J.Palmer. Designing for Web site usability. Computer. 35(7):102-103, Jul 2002.

[54] K. Velmurugan and M.A.M. Mohamed. An Empirical Performance Metrics Measurement and Analysis of Software Platforms for Implementation of Web Services. In Proceedings of the International Conference on Computer and Information Science (ICIS 2009). IEEE, 2009.

[55] P. Valente and F. Checconi. High Throughput Disk Scheduling with Fair Bandwidth Distribution. IEEE Transactions on Computers, 59(9):1172-1186, Sept. 2010.

[56] I. Ozkaya. L. Bass, R.L. Nord and R.S. Sangwan. Making Practical Use of Quality Attribute Information. IEEE Software, 25(2) : 25-33, March-April 2008.

[57] M. Yoon, C. Lee and J. Han. Migrating from Per-Job Analysis to Per-Resource Analysis for Tighter Bounds of End-to-End Response Times. IEEE Transactions on Computers, 59(7):933-942, July 2010.

[58] A. Pereira, L. Silvaa, W. Meira Jr. and W.Santos. Assessing the impact of reactive workloads on the performance of Web applications. In Proceedings of the International Symposium on Performance Analysis of Systems and Software. IEEE, 2006.

[59] L.Zheng-wu; J.Guo-song and W.Long-juan. A New Method to I/O Combination on RAID Controller. In Proceedings of the International Conference on Computer Science and Software Engineering. IEEE. 2008.

[60] D.A. Menasce. Performance and availability of Internet data centers. IEEE Internet Computing, 8(3): 94- 96, May-June 2004.

[61] D.A. Menasce. QoS issues in Web services. IEEE Internet Computing, 6(6): 72- 75, Nov/Dec 2002.

[62] J.W. Palmer. Web site usability, design, and performance metrics. Information systems research, 13(2): 151-167, 2002.

[63] S. Eyerman and L. Eeckhout. System-Level Performance Metrics for Multiprogram Workloads. IEEE Micro, 28(3): 42-53, May-June 2008.

[64] X. Franch and J.P. Carvallo. Using quality models in software package selection. IEEE Software, 20(1): 34- 41, Jan/Feb 2003.

[65] C. Lee, C. Chiang and M. Horng. Collaborative Web computing environment: an infrastructure for scientific computation. IEEE Internet Computing, 4(2):27-35, Mar/Apr 2000.

[66] I. Gorton and A. Liu. Evaluating the performance of EJB components. IEEE Internet Computing, 7(3):18- 23, May-June 2003.

[67] H.Wu, H. Chen, Q. Zhou and C. Gao. Throughput Performance of Marine STDMA Ad-hoc Network. In Proceedings of the 8 th International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (ACIS` 07). IEEE, 2007.

[68] R. Henjes, M. Menth and C. Zepfel. Throughput performance of java messaging services using websphere MQ. In Proceedings of the 26 th International Conference on Distributed Computing Systems Workshops (ICDCSW`06). IEEE, 2006.

[69] Z.Liu, R. Ma, F. Zhou, Y. Yang, Z. Qi and H. Guan. Power-aware I/O-Intensive and CPU-Intensive applications hybrid deployment within virtualization environments. In Proceedings of the International Conference on Progress in Informatics and Computing (PIC` 2010). IEEE, 2010.

[70] Y.Yuan. Efficiency metrics model for component-based embedded application software. In Proceedings of the 2nd International Conference on Embedded Software and Systems. IEEE, 2005.

[71] R. Racu, L.Li, R. Henia, A. Hamann and R. Ernst. Improved response time analysis of tasks scheduled under preemptive Round-Robin. In Proceedings of the 5th International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS` 07). IEEE/ ACM/IFIP, 2007.

[72] J. T. Hicks and J.Q. Walker II. A quick check of network performance. International Journal of Network Management, 11(1): 65-72, 2001.

[73] X.Liu, J.Heo, L. Sha and X.Zhu. Adaptive Control of Multi-Tiered Web Applications Using Queueing Predictor. In Proceedings of the 10th International Conference on Network Operations and Management Symposium(NOMS `06). IEEE/IFIP, 2006.

[74] Z. Ding, H. Shen and A. Kandel. Performance Analysis of Service Composition Based on Fuzzy Differential Equations. IEEE Transactions on Fuzzy Systems, 19(1):164-178, Feb. 2011.

[75] D.Liu, K. H. Law and G. Wiederhold. Analysis of integration models for service composition. In Proceedings of the 3rd international workshop on Software and performance, ACM, 2002.

[76] Y. Lu, T.Nolte, J.Kraft and C.Norström. A Statistical Approach to Response-Time Analysis of Complex Real-Time Embedded Systems. In Proceedings of the 16th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA `10).IEEE, 2010.

[77] S. Shahbazi, A. Harwood and S. Karunasekera. An analytical model for performance evaluation in sparse mobile ad hoc networks. In Proceedings of the 2nd International Conference on Wireless Days (WD `09). IFIP, 2009.

[78] J. Dabrowski and E. V. Munson. 40years of searching for the best computer system response time. Interacting with Computers, 23(5): 555-564, September, 2011.

[79] A. Bogardi-Mészöly and T. Levendovszky. A novel algorithm for performance prediction of web-based software systems. Performance Evaluation 68(1): 45-57, 2011.

[80] A. Alvaro, E. S. Almeida and S. R. L. Meira. Quality attributes for a component quality model. In 10th International Workshop on Component-Oriented Programming, WCOP Glasgow, Scotland, July 2005.

[81] M.F. Bertoa and A.Vallecillo. Quality attributes for COTS components. In Proceedings of the 6th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2002), June 2002.

[82] J.Wan, P.Shan and X. Xu. A Utility-Based Adaptive Resource Allocation Policy in Virtualized Environment. In Proceedings of the 8th International Conference on Dependable, Autonomic and Secure Computing (DASC '09). IEEE, 2009.

[83] N. Tomov, E. Dempster, M. H. Williams, A. Burger, H. Taylor, P. J. B. King, and P. Broughton. Analytical response time estimation in parallel relational database systems. Parallel Computing, 30(2): 249-283, 2004.

[84] M. J. Basavaraj and K. C. Shet. Estimating and Prediction of Turnaround Time for Incidents in Application Service Maintenance Projects. Journal of Software, 3(7): 12-21, 2008.

[85] D. Kim, H. Lim, J. Yoo, and S. Kim. Empirical Efficiency Testing for OSS. In Proceedings of the International Conference on Software Engineering Research and Practice, 2008.

[86] V. Rahimian and J. Habibi. Performance evaluation of mobile software systems: Challenges for a software engineer. In Proceedings of the 5th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE `08). IEEE, 2008.

[87] Y. Huang and B. Chao. A priority-based resource allocation strategy in distributed computing networks. Journal of Systems and Software, 58(3): 221-233, 2001.

[88] N. Roy, A. Dubey, A. Gokhale and L.Dowdy . A Capacity Planning Process for Performance Assurance of Component-Based Distributed Systems . In Proceedings of the 2nd International Conference on Performance Engineering (ICPE `11). ACM/ SPEC, 2011.

[89] L.Lu, H. Zhang, G. Jiang, H. Chen, K. Yoshihira, and E. Smirni. Untangling mixed information to calibrate resource utilization in virtual machines. In Proceedings of the 8th international conference on Autonomic computing. ACM, 2011.

[90] S. S. Craciunas, C.M. Kirsch and H. Röck. I/O resource management through system call scheduling. ACM SIGOPS Operating Systems Review, July 2008.

[91] T. Kaldewey, T.M.Wong, R. Golding, A. Povzner, S.Brandt and C. Maltzahn. Virtualizing Disk Performance. In Proceedings of the international conference on Real-Time and Embedded Technology and Applications Symposium (RTAS '08). IEEE, 2008.

[92] L. Watkins, W. H. Robinson and R. Beyah. A Passive Solution to the Memory Resource Discovery Problem in Computational Clusters. IEEE Transactions on Network and Service Management, 7(4): 218-230, December 2010.

[93] J.H. Hill, D. C. Schmidt and J. M. Slaby. Evaluating quality of service for enterprise distributed real-time and embedded systems. Designing Software-Intensive Systems: Methods and Principles. Idea Group, 2007.

[94] B. B. Chua and L. E. Dyson. Applying the ISO 9126 model to the evaluation of an elearning system. In Proceedings of the ASCILITE. 2004.

[95] I. Padayachee, P. Kotze and A. van Der Merwe. ISO 9126 external systems quality characteristics, sub-characteristics and domain specific criteria for evaluating e-Learning systems. The Southern African Computer Lecturers Association, University of Pretoria, South Africa , 2010.

[96] Werner Dirlewanger. Measurement and Rating of Computer Systems Performance and of Software Efficiency. University of Kassel, Germany, 2006.

[97] Mark B. Friedman. Measuring Processor Utilization in Windows and Windows applications. Demand Technology Software.

[98] S. Becker. Performance-related metrics in the ISO 9126 Standard. In Proceedings of the international conference on Real-Time and Embedded Technology and Applications Symposium (RTAS '08). IEEE, 2008.Dependability metrics (2008): 204-206.

[99] Y. Li, V. Suhendra, Y.Liang, T. Mitra and A. Roychoudhury. Timing Analysis of Concurrent Programs Running on Shared Cache Multi-Cores. In Proceedings of the 30th Real-Time Systems Symposium (RTSS`09).IEEE, 2009.

[100] X. E. Chen and T. M. Aamodt. Modeling Cache Contention and Throughput of Multiprogrammed Manycore Processors. IEEE Transactions on Computers, 61(7): 913-927, July 2012.

[101] A. Dandalis, V.K. Prasanna and J.D.P. Rolim. An adaptive cryptographic engine for IPSec architectures. In Proceedings of the international Symposium on Field-Programmable Custom Computing Machines(FCCM`00). IEEE, 2000.

[102] K . Boloor, R. Chirkova. T. Salo and Y. Viniotis. Analysis of Response Time Percentile Service Level Agreements in SOA-Based Applications. In Proceedings of the International Global Telecommunications Conference (GLOBECOM `11).IEEE, 2011.

[103] C. Belwal and A.M.K. Cheng. Determining Actual Response Time in P-FRP. In Proceedings of the 14th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC`11).IEEE, 2011.

[104] O.J.Haggarty, W.J. Knottenbelt and J.T. Bradley. Distributed response time analysis of GSPN models with MapReduce. In Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS `08). IEEE, 2008.

[105] C.Ryan and P.Rossi. Software, performance and resource utilisation metrics for context-aware mobile applications. In Proceedings of the 11th International Symposium on Software Metrics (METRICS` 05). IEEE, 2005.

[106] S. Balsamo, M. Marzolla and R. Mirandola. Efficient Performance Models in Component-Based Software Engineering. In Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications . IEEE, 2006.

[107] W. Suryn and F. Laverdière. The applicability analysis of ISO/IEC 9126 series of standards in telecom sector: the example of soft switch category of software. Software Quality Journal, 15(4): 417-448, 2007.

[108] M. Côté, W. Suryn and E. Georgiadou. In search for a widely applicable and accepted software quality model for software quality engineering. Software Quality Journal, 15(4): 401-416, 2007.

[109] A. Stefani and M. Xenos. E-commerce system quality assessment using a model based on ISO 9126 and Belief Networks. Software Quality Journal, 16(1): 107-129, 2008.

[110] M. Ortega, M. Pérez, and T. Rojas. Construction of a systemic quality model for evaluating a software product. Software Quality Journal, 11(3): 219-242, 2003.

[111] X. Zhang, D.Cao, H.Mei and F.Yang. Towards Balancing Determinism, Memory Consumption and Throughput for RTSJ-Based Real-Time Applications. In Proceedings of the 11th International Conference on Quality Software (QSIC`11). IEEE, 2011.

[112] J. J. M. Trienekens , R. J. Kusters, and D. C. Brussel. Quality specification and metrication, results from a case-study in a mission-critical software domain. Software Quality Journal, 18(4): 469-490, 2010.

[113] J.Ras and A.M.K. Cheng. Response Time Analysis of the Abort-and-Restart Model under Symmetric Multiprocessing. In Proceedings of the 10th International Conference on Computer and Information Technology (CIT `10). IEEE, 2010.

[114] S. Chen, Y. Liu, I. Gorton, and A. Liu. Performance prediction of component-based applications. Journal of Systems and Software, 74(1): 35-43, 2005.

[115] J.A. Hoxmeier and C. DiCesare. System response time and user satisfaction: An experimental study of browser-based applications. In Proceedings of the 6th Americas Conference on Information Systems, 2000.

[116] O. Redell and M. Torngren. Calculating exact worst case response times for static priority scheduled tasks with offsets and jitter. In Proceedings of the 8th International Symposium on Real-Time and Embedded Technology and Applications. IEEE, 2002.

[117] G. Pacifici, W. Segmuller, M. Spreitzer, M. Steinder, A. Tantawi, and A. Youssef. Managing the response time for multi-tiered web applications. Technical Report RC 23-65-1, IBM TJ Watson Research Center, 2005.

12 APPENDIX

Appendix A - Primary articles list

Article ID	Article title	Reference
A1	XHive Efficient Cooperative Caching for Virtual Machines	50
A2	A Comprehensive Metric to Evaluating Software Aging	51
A3	Design, Implementation, and Performance Evaluation of a Detection-Based Adaptive Block Replacement Scheme	52
A4	Designing for Web Site Usability	53
A5	An Empirical Performance Metrics Measurement and Analysis of Software Platforms for Implementation of Web Services	54
A6	High Throughput Disk Scheduling with Fair Bandwidth Distribution	55
A7	Making Practical Use of Quality Attribute Information	56
A8	Migrating from Per-Job Analysis to Per-Resource Analysis for Tighter Bounds of End-to-End Response Times	57
A9	Assessing the Impact of Reactive Workloads on the Performance of Web Applications	58
A10	A New Method to I/O Combination on RAID Controller	59
A11	Performance and Availability of Internet Data Centers	60
A12	QoS Issues in Web Services	61
A13	Website Usability, Design and Performance Metrics	62
A14	SYSTEM-LEVEL PERFORMANCE METRICS FOR MULTIPROGRAM WORKLOADS	63
A15	Using Quality Models in Software Package Selection	64
A16	Collaborative WebComputing Environment:An Infrastructure for Scientific Computation	65
A17	Evaluating the Performance of EJB Components	66
A18	Throughput Performance of Marine STDMA Ad-hoc Network	67
A19	Throughput Performance of Java Messaging Services Using WebsphereMQ	68
A20	Power-aware I/O-Intensive and CPU-Intensive Applications Hybrid Deployment within Virtualization Environments	69
A21	Efficiency Metrics Model for Component-based Embedded Application Software	70
A22	Improved Response Time Analysis of Tasks Scheduled under Preemptive Round-Robin	71
A23	A quick check of network performance	72
A24	Adaptive Control of Multi-Tiered Web Applications Using Queueing Predictor	73
A25	Performance Analysis of Service Composition Based on Fuzzy Differential Equations	74
A26	Analysis of Integration Models for Service Composition	75
A27	A statistical approach to response-time analysis of complex embedded real time systems	76
A28	An Analytical Model for Performance Evaluation in Sparse Mobile Ad hoc Networks	77
A29	40 Years of searching for the best computer system response time	78
A30	A novel algorithm for performance prediction of web-based software systems	79
A31	Quality Attributes for a Component Quality Model	80
A32	Quality Attributes for COTS Components	81
A33	A Utility-based Adaptive Resource Allocation Policy in Virtualized Environment	82
A34	Analytical response time estimation in parallel relational database systems	83
A35	Estimating and Prediction of Turn around Time for Incidents in Application Service Maintenance Projects	84
A36	Empirical Efficiency Testing for OSS	85
A37	Performance Evaluation of Mobile Software Systems Challenges for a Software Engineer	86
A38	A priority based resource allocation strategy in distributed computing networks	87
A39	A Capacity Planning Process for Performance Assurance of Component-Based Distributed Systems	88

A40	Untangling Mixed Information to Calibrate Resource Utilization in Virtual Machines	**89**
A41	I/O Resource Management through System Call Scheduling	**90**
A42	Virtualizing Disk Performance	**91**
A43	A Passive Solution to the Memory Resource Discovery Problem in Computational Clusters	**92**
A44	EVALUATING QUALITY OF SERVICE FOR ENTERPRISE DISTRIBUTED REAL-TIME AND EMBEDDED SYSTEMS	**93**
A45	Applying the ISO 9126 model to the evaluation of an elearning system - imp	**94**
A46	ISO 9126 external systems quality characteristics, subcharacteristics and domain specific criteria for evaluating e-Learning systems	**95**
A47	Measurement and Rating of Computer Systems Performance and of Software Efficiency	**96**
A48	Measuring Processor Utilization in Windows and Windows applications	**97**
A49	Performance-Related Metrics in the ISO 9126 Standard	**98**
A50	Timing Analysis of Concurrent Programs Running on Shared Cache Multi-Cores	**99**
A51	Modeling Cache Contention and Throughput of Multiprogrammed Manycore Processors	**100**
A52	An Adaptive Cryptographic Engine for IPSec Architectures	**101**
A53	Analysis of response time percentile service level agreements in SOA-based applications	**102**
A54	Determining Actual Response Time in P-FRP	**103**
A55	Distributed Response Time Analysis of GSPN Models with MapReduce	**104**
A56	Software, Performance and Resource Utilisation Metrics for Context-Aware Mobile Applications	**105**
A57	Efficient Performance Models in Component-Based Software Engineering	**106**
A58	The applicability analysis of ISO/IEC 9126 series of standards in telecom sector: the example of softswitch category of software	**107**
A59	In search for a widely applicable and accepted software quality model for software quality engineering	**108**
A60	E-commerce system quality assessment using a model based on ISO 9126 and Belief Networks	**109**
A61	Construction of a Systemic Quality Model for Evaluating a Software Product	**110**
A62	Towards Balancing Determinism, Memory Consumption and Throughput for RTSJ-Based Real-Time Applications	**111**
A63	Quality specification and metrication, results from a case-study in a mission-critical software domain	**112**
A64	Response Time Analysis of the Abort-and-Restart Model under Symmetric Multiprocessing	**113**
A65	Performance Prediction of Component-based Applications	**114**
A66	System Response Time and User Satisfaction: An Experimental Study of Browser-based Applications	**115**
A67	Calculating ExactWorst Case Response Times for Static Priority Scheduled Tasks with Offsets and Jitter	**116**
A68	Managing the Response Time for Multi-tiered Web Applications	**117**

Appendix B - Experiment Results

Case A - Response time - Uncached HTML - 0.3 Mbps:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	15.32	17.89	12.58	15.93
T2	16.79	18.57	11.98	13.01
T3	14.24	15.34	11.81	13.68
T4	12.21	15.98	13.89	17.09
T5	14.53	16.99	12.78	15.64
T6	13.67	17.95	11.56	14.65
T7	13.45	17.28	10.34	15.35
T8	14.24	15.64	13.88	16.30
T9	15.89	16.89	14.62	13.66
T10	13.26	16.17	17.89	12.69
T11	12.35	15.17	11.77	16.32
T12	11.73	18.96	12.34	13.21
T13	15.98	16.66	11.90	15.68
T14	16.26	17.45	12.23	14.45
T15	11.83	17.27	13.33	13.09
T16	12.34	17.00	11.77	13.06
T17	13.27	16.67	12.67	15.09
T18	14.28	15.89	11.23	14.64
T19	11.35	16.18	10.23	15.69
T20	12.32	16.20	13.45	13.86
T21	14.32	15.37	11.18	14.78
T22	12.44	15.88	12.35	15.45
T23	11.35	17.22	12.68	14.54
T24	14.89	17.89	11.39	13.75
T25	16.47	18.23	12.36	14.87

Case A - Response time - Cached HTML - 0.3 Mbps:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	3.64	3.67	3.38	4.21
T2	3.35	4.14	2.85	3.43
T3	3.78	3.73	2.71	4.36
T4	3.55	3.92	3.31	3.17
T5	3.30	4.56	2.77	3.98
T6	3.15	4.11	2.65	2.99
T7	3.50	3.78	2.92	4.78
T8	3.31	3.49	2.69	3.21
T9	3.77	3.43	2.80	4.60
T10	3.29	4.36	3.10	3.23
T11	3.24	3.68	2.96	4.26
T12	3.52	3.79	3.24	4.15
T13	3.19	3.15	2.79	4.12
T14	3.51	4.50	3.33	3.87
T15	3.32	3.17	3.15	3.74
T16	3.55	4.57	3.00	4.16

T17	3.17	3.20	3.33	3.65
T18	3.53	4.14	2.95	4.24
T19	3.20	3.59	3.08	3.12
T20	3.99	4.50	2.65	4.25
T21	3.20	3.65	3.35	3.22
T22	3.53	3.66	2.70	3.81
T23	3.27	4.69	2.68	3.00
T24	3.48	4.30	2.66	4.98
T25	3.15	4.31	2.98	3.37

Case A - Response time - Uncached HTML - 0.8 Mbps:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	10.30	14.58	10.53	10.39
T2	10.25	13.81	10.96	11.77
T3	11.38	15.52	11.56	11.54
T4	8.64	15.34	10.57	11.31
T5	13.83	11.64	10.12	9.88
T6	7.63	11.54	9.99	10.26
T7	10.28	12.75	9.56	11.90
T8	12.40	15.46	12.23	11.32
T9	11.62	14.77	8.89	9.72
T10	13.28	11.63	10.78	12.78
T11	10.24	11.26	11.67	8.88
T12	7.24	13.72	10.79	11.43
T13	8.31	15.81	9.36	10.71
T14	13.85	12.89	8.67	9.83
T15	12.92	11.09	9.22	8.73
T16	13.11	14.78	10.38	13.56
T17	11.09	15.92	11.29	12.33
T18	10.23	13.89	10.29	11.34
T19	9.89	15.40	9.28	9.78
T20	10.93	12.49	10.38	13.56
T21	11.93	13.56	9.29	12.13
T22	13.89	15.90	8.23	11.88
T23	10.92	11.32	10.29	12.13
T24	9.09	12.37	10.23	8.88
T25	8.98	11.90	11.23	10.18

Case A - Response time - Cached HTML - 0.8 Mbps:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	2.25	2.56	1.95	2.19
T2	2.71	2.23	1.75	2.28
T3	2.93	2.75	1.72	2.22
T4	2.24	2.24	1.35	2.23
T5	2.42	2.43	1.19	1.83
T6	2.13	2.39	2.18	2.12
T7	1.51	2.37	1.93	1.81

T8	2.29	2.47	1.37	2.17
T9	1.67	2.42	1.22	1.83
T10	1.93	2.35	1.18	2.56
T11	1.56	2.44	1.96	1.88
T12	2.44	2.21	1.21	2.11
T13	2.63	2.11	1.03	1.84
T14	2.88	2.11	1.19	2.78
T15	2.31	2.85	1.67	1.85
T16	2.83	2.61	1.96	2.57
T17	2.11	2.25	0.99	2.80
T18	1.92	2.31	1.22	1.75
T19	1.72	2.29	1.93	2.51
T20	1.61	2.44	1.38	2.65
T21	1.70	2.39	1.45	2.49
T22	1.63	2.19	1.46	2.23
T23	2.41	2.22	2.50	1.82
T24	1.44	2.18	1.76	2.20
T25	1.55	2.21	1.95	1.92

Case A - Throughput - 20 Fishes:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	59	36	59	58
T2	59	35	59	59
T3	59	35	59	59
T4	60	36	59	58
T5	59	37	59	59
T6	59	35	59	58
T7	59	36	59	58
T8	59	39	59	58
T9	58	37	58	58
T10	59	36	59	58
T11	59	39	59	58
T12	58	37	59	58
T13	59	38	58	57
T14	60	36	59	58
T15	59	39	59	56
T16	58	39	58	58
T17	59	38	59	58
T18	60	37	59	59
T19	59	34	58	58
T20	58	39	59	58
T21	59	37	55	59
T22	59	36	59	58
T23	57	38	59	58
T24	59	37	59	58
T25	59	39	58	57

Case A - Throughput - 100 Fishes:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	59	18	59	59
T2	58	19	58	59
T3	57	20	58	59
T4	58	18	59	59
T5	58	21	59	59
T6	57	18	58	59
T7	58	19	58	59
T8	58	18	59	58
T9	58	21	58	59
T10	58	21	58	59
T11	58	20	58	59
T12	58	19	59	58
T13	58	21	58	59
T14	58	18	58	59
T15	58	21	59	58
T16	57	19	58	59
T17	58	21	58	59
T18	58	20	59	59
T19	58	21	58	59
T20	56	20	59	59
T21	58	19	59	58
T22	58	21	58	59
T23	58	19	58	59
T24	58	18	59	59
T25	58	21	59	59

Case A - Throughput - 250 Fishes:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	59	12	58	46
T2	58	9	59	48
T3	59	8	58	48
T4	57	10	58	47
T5	58	11	58	48
T6	57	10	58	47
T7	57	11	59	46
T8	58	12	58	48
T9	58	9	58	46
T10	57	12	58	47
T11	56	11	58	48
T12	58	9	59	46
T13	57	11	58	47
T14	58	10	58	45
T15	58	12	58	45
T16	58	9	59	44
T17	58	10	58	46
T18	58	11	58	47
T19	57	12	59	48

T20	58	10	58	44
T21	58	9	58	45
T22	57	12	58	44
T23	58	11	59	46
T24	58	9	58	44
T25	58	12	58	48

Case A - Memory Utilization - Empty browser:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	80496	75996	41696	58464
T2	80768	71992	41884	55368
T3	80764	71820	41884	55672
T4	79856	71296	41994	52060
T5	79156	71436	41980	55032
T6	80456	71528	41788	52112
T7	80436	72128	41576	56748
T8	79808	71840	42340	56672
T9	80676	71804	42288	56092
T10	77588	71100	43204	55680
T11	80836	71248	41572	55072
T12	80200	71372	42320	55664
T13	77884	72444	41976	55352
T14	80096	71736	42464	55420
T15	80032	71460	42204	56628
T16	80819	71932	41892	56734
T17	80515	71898	41932	55234
T18	81234	71112	42561	56833
T19	80832	72091	42891	51932
T20	78416	71191	42892	53322
T21	79672	72233	41445	56921
T22	82224	74211	42272	55829
T23	83567	70091	41823	59919
T24	75292	73910	41821	53818
T25	76893	73561	42833	56901

Case A - Memory Utilization - Single HTML:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	101653	91972	68876	74368
T2	102832	91772	67092	74832
T3	103832	94100	67756	70888
T4	102684	95068	67408	71440
T5	100120	91580	67020	72444
T6	104540	93704	67268	76788
T7	104962	95906	67060	75782
T8	105192	92856	67688	71532
T9	102492	95132	68086	70960
T10	101056	94608	67284	72784

T11	98288	95680	68360	70888
T12	99640	89844	68864	71064
T13	98920	88340	67488	75840
T14	101304	90284	69320	76714
T15	102915	90926	67782	75891
T16	103913	92392	67727	70901
T17	104432	88712	67561	72891
T18	102811	88818	68409	76350
T19	101910	89617	67388	72356
T20	102181	94616	68451	76891
T21	100291	88215	67108	76911
T22	97118	91717	67677	72913
T23	95181	88839	68455	74443
T24	99121	94781	67891	75891
T25	98915	93918	67771	73719

Case A - Memory Utilization - Multiple HTML:

Browser :	Google Chrome	Mozilla Firefox	Internet Explorer	Opera
T1	118564	119424	86016	102516
T2	115496	117592	84172	101784
T3	110712	117640	84016	96048
T4	116480	118716	84392	97776
T5	113948	116124	84356	96948
T6	119204	116544	84032	104472
T7	118444	111552	86552	99208
T8	110352	113888	83508	101350
T9	116420	113532	83720	99316
T10	116806	116676	87296	104224
T11	117116	118132	84216	102088
T12	110796	119060	84104	95732
T13	117792	111860	83904	99328
T14	114108	119688	84068	95000
T15	116192	112264	83648	96188
T16	119991	113732	85474	98393
T17	117892	116282	86292	97234
T18	115671	116117	84287	95223
T19	113367	118993	86213	97344
T20	115282	116282	84234	95345
T21	117272	115112	83134	101992
T22	110191	115566	83322	103234
T23	119782	118229	81345	103393
T24	117829	119113	84292	96893
T25	116188	118238	83181	98564

Case B - Response time - Uncached HTML - 0.3 Mbps:

Browser :	Midori	Sea Monkey	Qupzilla	Arora
T1	20.66	21.65	24.55	21.55
T2	24.33	21.57	24.17	21.18

T3	20.10	20.34	23.30	21.81
T4	24.40	24.98	22.70	19.70
T5	22.56	22.99	25.75	19.75
T6	23.86	23.94	23.12	22.12
T7	24.86	21.28	25.87	18.78
T8	22.21	20.64	23.65	21.62
T9	21.96	22.89	22.99	21.65
T10	20.71	20.16	24.35	19.09
T11	21.18	20.88	26.09	21.38
T12	20.41	24.78	25.68	21.99
T13	24.36	22.66	23.77	19.01
T14	22.91	20.45	22.36	20.86
T15	21.89	21.27	24.33	19.27
T16	20.92	20.00	25.66	20.75
T17	21.72	22.67	25.42	22.23
T18	23.96	20.89	25.61	18.98
T19	20.21	23.18	26.36	21.20
T20	22.90	21.96	26.54	20.01
T21	21.39	23.37	25.56	19.92
T22	21.60	22.88	24.99	21.11
T23	21.81	20.22	24.36	22.01
T24	20.82	21.89	23.18	20.84
T25	22.51	23.98	25.80	20.55

Case B - Response time - Cached HTML - 0.3 Mbps:

Browser :	Midori	Sea Monkey	Qupzilla	Arora
T1	5.15	5.36	5.02	4.25
T2	3.33	4.82	5.26	5.41
T3	3.87	4.27	5.16	3.23
T4	5.10	4.88	4.73	3.09
T5	4.13	4.42	5.31	2.12
T6	5.90	4.86	4.70	3.89
T7	3.83	4.28	5.05	4.78
T8	3.80	5.23	6.29	5.46
T9	5.28	4.33	4.82	6.32
T10	3.12	5.08	5.01	2.79
T11	3.30	4.83	4.75	2.99
T12	3.20	5.30	5.01	2.46
T13	4.87	5.34	4.76	4.87
T14	4.99	4.98	6.46	6.76
T15	5.73	4.61	6.38	5.57
T16	3.50	3.67	6.07	4.65
T17	5.27	4.78	4.71	3.78
T18	3.36	4.60	5.33	5.55
T19	4.81	4.98	4.66	5.98
T20	3.17	5.35	5.00	4.23
T21	4.78	4.83	4.74	2.57
T22	4.41	4.81	5.03	3.23

T23	3.83	5.24	4.69	4.33
T24	4.40	4.62	5.00	3.87
T25	3.54	5.12	4.73	3.45

Case B - Response time - Uncached HTML - 0.8 Mbps:

Browser :	**Midori**	**Sea Monkey**	**Qupzilla**	**Arora**
T1	18.46	18.67	21.31	19.58
T2	17.59	21.38	21.14	18.81
T3	22.45	21.20	19.69	25.52
T4	18.27	22.20	27.40	22.35
T5	19.22	19.39	20.41	16.64
T6	17.84	23.13	19.45	16.54
T7	13.99	17.98	21.89	17.75
T8	22.69	19.37	21.82	20.46
T9	14.86	21.66	21.59	19.77
T10	17.69	21.78	20.74	16.63
T11	14.87	18.84	20.84	16.26
T12	20.19	17.68	21.87	18.72
T13	21.33	17.99	22.31	20.81
T14	25.95	20.10	19.24	17.89
T15	15.55	24.45	22.09	16.09
T16	20.31	18.12	21.02	19.78
T17	19.11	16.12	22.03	22.92
T18	20.32	19.08	21.79	18.89
T19	18.68	22.23	22.59	20.40
T20	19.98	17.35	20.93	17.49
T21	17.98	18.11	21.22	18.56
T22	18.12	21.98	21.89	20.90
T23	21.78	16.84	19.81	16.32
T24	16.98	19.34	21.09	17.37
T25	17.21	23.10	21.92	16.90

Case B - Response time - Cached HTML - 0.8 Mbps:

Browser :	**Midori**	**Sea Monkey**	**Qupzilla**	**Arora**
T1	3.67	4.40	4.68	3.40
T2	2.90	4.43	3.85	4.20
T3	2.92	3.99	4.19	3.43
T4	2.86	3.57	3.79	3.43
T5	3.89	4.16	4.11	3.78
T6	4.03	3.60	3.76	3.39
T7	3.11	4.32	4.18	3.49
T8	2.86	3.58	3.78	4.12
T9	2.94	3.82	4.10	3.50
T10	2.95	3.87	3.10	3.41
T11	3.91	3.61	4.15	3.51
T12	4.44	3.39	3.82	4.34
T13	3.03	3.92	4.10	3.43
T14	2.87	3.64	3.80	3.40
T15	2.95	3.56	4.16	3.18

T16	2.97	3.83	3.84	3.45
T17	4.01	3.59	4.44	3.48
T18	2.90	3.64	3.38	3.39
T19	2.96	3.57	3.35	3.48
T20	3.78	4.18	3.33	3.57
T21	2.95	3.91	3.60	4.11
T22	2.91	4.14	3.36	3.42
T23	2.92	3.91	4.61	3.53
T24	2.86	3.67	4.89	3.52
T25	2.92	3.60	4.37	3.52

Case B - Throughput - 20 Fishes:

Browser :	**Midori**	**Sea Monkey**	**Qupzilla**	**Arora**
T1	45	15	28	55
T2	45	13	26	58
T3	44	14	29	56
T4	43	14	28	54
T5	45	15	27	56
T6	42	16	28	54
T7	43	15	29	55
T8	42	14	26	54
T9	43	16	29	54
T10	42	15	29	54
T11	42	13	26	56
T12	43	14	28	58
T13	44	13	29	56
T14	45	15	26	54
T15	45	14	27	58
T16	43	13	28	56
T17	42	15	27	54
T18	42	15	27	54
T19	45	14	29	57
T20	45	15	26	58
T21	43	16	27	58
T22	42	14	26	54
T23	44	14	28	57
T24	43	13	27	55
T25	42	14	26	54

Case B - Throughput - 100 Fishes:

Browser :	**Midori**	**Sea Monkey**	**Qupzilla**	**Arora**
T1	22	13	21	54
T2	21	13	21	52
T3	23	12	20	54
T4	21	11	19	52
T5	22	13	18	51
T6	19	12	18	50

T7	22	11	21	52
T8	23	12	20	51
T9	20	12	21	53
T10	22	13	22	52
T11	19	11	19	54
T12	20	12	22	52
T13	21	11	21	53
T14	20	12	19	51
T15	22	13	20	50
T16	21	13	19	52
T17	22	11	22	52
T18	21	12	21	51
T19	20	13	19	52
T20	21	12	21	54
T21	22	11	22	50
T22	21	13	19	51
T23	21	13	22	53
T24	20	13	21	52
T25	19	12	22	53

Case B - Throughput - 250 Fishes:

Browser :	Midori	Sea Monkey	Qupzilla	Arora
T1	20	10	17	50
T2	19	8	17	49
T3	18	8	16	48
T4	18	9	15	49
T5	18	8	15	50
T6	18	9	15	49
T7	18	10	16	49
T8	20	10	17	49
T9	18	10	15	49
T10	18	9	16	48
T11	18	9	16	49
T12	19	8	17	49
T13	18	8	17	47
T14	18	8	16	49
T15	20	8	15	49
T16	18	9	17	50
T17	19	9	15	49
T18	18	9	16	49
T19	18	10	16	50
T20	20	10	17	49
T21	19	9	15	50
T22	18	8	16	49
T23	19	10	15	48
T24	18	9	17	49
T25	18	9	16	47

Case B - Memory Utilization - Empty browser:

Browser :	Midori	Sea Monkey	Qupzilla	Arora
T1	97996	88240	59664	100380
T2	86992	88900	59744	93884
T3	86820	88728	59588	98564
T4	86296	88376	59620	90460
T5	86436	88912	59504	94812
T6	86528	88380	59660	95324
T7	87128	88860	59560	94804
T8	86840	88480	59604	95236
T9	86804	88484	59628	92716
T10	86100	88500	59504	90912
T11	86248	88144	59460	95536
T12	86372	88144	59432	93356
T13	87444	88316	59881	90912
T14	86736	88984	59345	92608
T15	86460	89152	59908	90740
T16	86932	87901	59829	94674
T17	86898	86818	59232	94345
T18	86112	88819	59421	95675
T19	87091	87091	59324	96317
T20	86191	85893	59482	96894
T21	87233	87901	59567	94127
T22	89211	88453	59091	93317
T23	85091	88243	59718	92256
T24	88910	87133	59723	95543
T25	88561	89891	59820	97454

Case B - Memory Utilization - Single HTML:

Browser :	Midori	Sea Monkey	Qupzilla	Arora
T1	107708	107036	86657	115324
T2	105628	108420	86408	114876
T3	105724	107378	86384	115040
T4	106068	109108	86292	115140
T5	105716	107123	85680	114692
T6	105940	106931	86023	114728
T7	105988	104213	85728	115484
T8	105432	106183	86656	114628
T9	105892	108092	86188	114892
T10	105988	106944	85617	115001
T11	106004	106840	84324	114822
T12	106356	107982	85124	115221
T13	106064	105667	84555	114922
T14	105868	106675	86343	114980
T15	106887	105881	86728	114474
T16	106728	107881	86829	114328
T17	106262	103392	84282	114272
T18	105893	108281	82892	115839

T19	106929	109282	84228	114322
T20	105463	106373	82292	114737
T21	105898	107933	83282	115721
T22	107892	105829	84111	114667
T23	106782	105820	83228	114892
T24	106821	104432	82282	114636
T25	106710	107829	82282	114617

Case B - Memory Utilization - Multiple HTML:

Browser :	Midori	Sea Monkey	Qupzilla	Arora
T1	114940	111544	108940	131268
T2	118552	110272	109567	130412
T3	116128	111764	101996	130244
T4	118840	110316	101416	130272
T5	118668	112093	101408	131453
T6	118744	113091	106108	131202
T7	118240	110145	101692	131533
T8	118792	111091	101420	130222
T9	118352	113789	102782	132032
T10	118848	110256	103113	131234
T11	118500	113113	102567	132983
T12	118844	113130	105333	131201
T13	118160	110878	103456	130383
T14	118256	110902	101223	131212
T15	118068	111783	104786	131333
T16	118383	110672	107181	130838
T17	114384	113382	108182	130829
T18	117831	112289	102829	132993
T19	119345	111783	103333	130363
T20	117354	111671	109332	131738
T21	118489	110811	100913	132892
T22	117932	113892	102292	131178
T23	118677	111901	106892	130881
T24	118282	112292	105782	130117
T25	118221	115892	105891	130187

Appendix C - Overview of usage of tools

Table 36 - Overview of tools used in the experiment process

Tool Name	**Measure**	**Calculated**	**Units**
Numion Stopwatch	Response time (RT)	Loading time of HTML Page	Seconds (Sec)
IE Fish Tank Test	Throughput (TH)	Number of frames loaded per second.	Frames (FPS)
Resource Monitor option of the Task Manager	Memory utilization (MU)	Space occupied by browser on disk.	Kilobytes (KB)
Net limiter pro	Adjusting Server Speed	Speed at which browser operates.	Megabits per second (Mbps).