ISSN 2229-5518

EVALUATING PERFORMANCE OF MULTIPLE SYSTEM INNOVATIONS

Atabong, T. Agendia-Abanda. and Nnadi, Hillary.

[1]. Department of Computer Science
Madonna University Elele, Nigeria.
Phone:+2347035085016; +23777797929
Email:t.atabong@madonnauniversity.edu.ng; tagendia@tibasr.net;tagendia@yahoo.com

ABSTRACT

At first taught, an undergraduate student of computer science or other computer users will appreciate the working of their computer system as good if their systems specifications are within some acceptable range. Vendors and users of computer systems will want to know the processor model, processor speed, memory size and rating, hard disk size and rating and the operating system they are using, as the determining factors of their new system. The measurement and evaluation of the performance of a computer system to compare alternatives in Madonna university laboratory is carried out in this research. Thirty (30) systems were analysed for performance using Response time performance matrix. During this analysis, Systems tuning was done with 20 factors (variables) and the different levels of the respective response time was obtained with a highly sensitive stop watch aimed at ensuring that a minimum amount of effort, expense and waste is incurred in getting the exact elapsed time. Both software and hardware were considered during the tuning. The geometric, arithmetic, harmonic and combined means were calculated during the evaluation. Missing levels of factors were equally calculated in two ways; from the recorded values of the other factors of the same systems by taking the harmonic mean; and by taking the harmonic mean values of that factor when all the 30 systems are considered. A comparison of all the 30 systems was done and presented accordingly. The architecture of the software system as the highest level of abstraction where upon useful analysis of system properties is possible was discussed from the presented overall performance of the software. Our presented graph can be used to predicts the throughput and the average response time of software system under varying workloads and also identifies bottlenecks in the system, suggesting possibilities for their removal.

keywords: System Tuning, Response time, throughput, harmonic mean, geometric mean, arithmetic mean.

1: INTRODUCTION

Many professionals and researchers have gone for a system with the most recent specifications but which fail to give them the desire results. They got a system with the best processor in the market, the highest hard disk capacity, the biggest memory coupled with the most recent operating system which fail to do the job their old system was performing at the fly [5].

Presently, computer system users, administrators, and designers are all interested in performance evaluation since their goal is to obtain or provide the highest performance at the lowest cost[1][48][66]. This goal has resulted in continuing evolution of higher performance and lower cost systems leading to today's proliferation of workstations and personal computers, many of which have better performance than earlier supercomputers[6]. As the field of computer design matures, the computer industry is becoming more competitive, and it is more important than ever, to ensure that, the alternative selected, provides the best cost-performance trade-off[1][7].

Performance evaluation is required at every stage in the life cycle of a computer system, including its design, manufacturing, sales/purchase, use and upgrade[6]. Program Performance Evaluation is key in the decision making process[9][14][45]. A performance evaluation is required when a computer system designer wants to compare a number of alternative designs and find the best design[1][10]. It is required when a system administrator wants to compare a number of systems to decide which system is best for a given set of applications[11][46][47]. Even if there are no alternatives, performance evaluation of the current system helps in determining how well it is performing and whether there is need for improvements[6][2]. Unfortunately, the types of applications of computers, are so numerous that it is not possible to have a standard measure of performance, a standard measurement environment (application), or a standard technique for all cases[3][12]. Before commencing a performance evaluation experiment, it is necessary to select the right measures of performance, the right measurement environments, and the right techniques[48].

Performance is a key criterion in the design, procurement, and use of computer systems[13]. As such, the goal of computer systems engineers, scientists, analysts, and users is to get the highest performance for a given cost[14]. Therefore, to achieve this goal, computer systems professionals need, at least, a basic knowledge of performance evaluation terminology and techniques[5]. Anyone associated with computer systems should be able to state the performance requirements of their systems and should be able to compare different alternatives to find the one that best meets their requirements [1][47].

Performance measurement approaches has been seen as *complementary* to program evaluation [8][2]. Analysts in the evaluation field as seen in [6][14][20][30][15][16][21][23] have generally recognized this complementation. In some jurisdictions, efforts to embrace performance measurement have eclipsed program evaluation [22][15]. There is growing evidence that the promises that have been made for performance measurement as an accountability and performance management tool [9], have not materialized [16][26].

Performance evaluation are done at all levels in the development of a computer hardware or computer software. In most software the performance is evaluated at the architectural level [8][17][18]. Two main issues involved in the evaluation of performance at the architectural level as opined by Semonetta etal, are; first the designer may need to choose among several alternative software architectures for the system, with the choice being driven especially by performance considerations; and secondly, for a specific software architecture of the system, the designer may want to understand whether its performance can be improved and, if so, it would be desirable for the designer to have some diagnostic information that guide the modification of the software architecture itself [3].

Computer performance is characterized by the amount of useful work accomplished by a computer system compared to the time and resources used[48], in this light, Atabong *etal* [23] observed that, depending on the context, good computer performance may be computed using one or more performance metric. This metric has been referred to as the criteria used to evaluate the performance of the system [1][4]. Referred to as time for service of a request, the response time metric is amongst the performance metrics which has been widely used for evaluating performance of software and hardware systems[40]. This metric has been used as a metric to compare two timesharing systems[49].

Similarly, the throughput metric measured in transactions per second two has been used to compare the performance of two or more transaction processing systems[51]. In a Human computer interface system, response time is taken to be the time it takes for the system to react to a given input while in a data processing system, the response time is perceived by the end user in the interval between, (a) the instant at which an operator at a terminal enters a request for a response from a computer and (b) the instant at which the first character of the response is received at a terminal[48][53]. Response time in data transmission system is the interval between the receipt of the end of transmission of an inquiry message and the beginning of the transmission of a response message to the station originating the inquiry[23]. In real-time systems the response time of a task or thread is defined as the time elapsed between the dispatch to the time when it finishes its the discharge of one job [19].

Like response time, throughput has been widely used for evaluating performance of a computer system. The system throughput or aggregate throughput is the sum of the data rates that are delivered to all terminals in a network[36]. The throughput can be analyzed mathematically by means of queuing theory, where the load in packets per time unit is denoted arrival rate λ, and the throughput in packets per time unit is denoted departure rate μ. Throughput is essentially synonymous to digital bandwidth consumption[23][24].

Utilization of computing resources(s) is a metric which measure the how effective a system makes use of its limited resources[54]. A resource, or system resource, is any physical or virtual component of limited availability within a computer system, this include; every device connected to a computer system and every internal system component[6]. Both virtual and physical system resource have been used for various performance evaluation of systems[23]. Virtual system resources widely involved in performance evaluation include files, network connections and memory areas. Physical resources on most performance evaluation include: CPU time, Random access memory and virtual memory, Hard disk space, Network throughput, Electrical power and External Devices[34].

Data compression and decompression programs are used for evaluating the performance of a system in which data transmission is involved. In such cases, the bandwidth performance metric has been proven to be very suitable for the evaluation[58]. Data compression or source coding is the process of encoding information using fewer bits than an unencoded representation would use through use of specific encoding schemes[43]. Compression is useful because it helps reduce the consumption of expensive resources, such as hard disk space or transmission bandwidth[48][6]. Also called, data transmission time, digital bandwidth, network bandwidth or just bandwidth is a measure of available or consumed data communication resources expressed in bit/s or multiples of it (kbit/s, Mbit/s etc)[23]. It may equally refers to bandwidth capacity or available bandwidth in bit/s, which typically means the net bit rate, channel capacity or the maximum throughput of a logical or physical communication path in a digital communication system[36].

In Madonna University Elele, Computer users and professional have encounter various problems based on the performance of their systems. This low performance has triggers a lot of questions on the suppliers of these digital products and network providers. A good number of performance evaluation has been carried out at the university for various objectives. Most of the performance evaluation was directed towards improving the web accessibility[23][40][41][42][43]. Complimenting these researches with an evaluation of the software and hardware supplied in the laboratory and offices in 2014 will be a way forward towards improving the overall performance of computer systems as well as users performance. It a proven facts, in the history of technological advancement that, performance evaluation conducted on both software and hardware of a computer system has resulted in great improvement of previous systems[28][29][31][34].

Memory Management, Power mode management, Processor Management and I/O managements are some of the elements of computing hardware which have undergone significant improvement over their history as a result of continuous performance evaluation[50]. This improvement has triggered worldwide use of the technology, performance has improved and the price has declined[32]. Computers are accessible to ever-increasing sectors of the world's population. Computing hardware has become a platform for uses other than computation, such as automation, communication, control, entertainment, and education in which performance cannot be dropped out[38]. Like computer science most disciplines haves imposed its own requirements which has evolved in response to those requirements[33][66].

Since digital computers rely on digital storage, and tend to be limited by the size and speed of memory, the history of computer data storage is tied to the development of computers based on continuous performance evaluation[60]. This major advancement technology has implemented reduced abstractions in input, output, memory, and processor. The complexity of a processor control and data path have greatly increased and with significant improvement in its processes[48]. Looking at the original von Neumann architecture, control of the data path is stored in memory which allows control to become an automatic process and the data path under software control in response to events. Beginning with mechanical data paths such as the abacus and astrolabe, the hardware first started using analogs for a computation, including water and even air as the analog quantities: analog computers have used lengths, pressures, voltages, and currents to represent the results of calculations. Eventually the voltages or currents were standardized, and then digitized. Digital computing elements have ranged from mechanical gears, to electromechanical relays, to vacuum tubes, to transistors, and to integrated circuits, all of which are currently implementing the von Neumann architecture [60][61]. This shows that the art of performance evaluation existed from the time the first use of the word "computer" as was recorded in 1613.

The components connected to the computer has been shown to have significant effects of the performance of the system[27][60]. Based on the performance metric considered, performance of old and new systems have not changed significantly because the architectural design of modern systems still relies heavily on von Neumann [6][61].

A common approach to assessing performance is to use a numerical or scalar rating system whereby managers are asked to score an individual against a number of objectives/attributes. In some companies, employees receive assessments from their manager, peers, subordinates and customers while also performing a self assessment. This is known as 360° appraisal. Forms good communication patterns. The most popular methods that are being used in the performance appraisal process include: Management by objectives, 360 degree appraisal, Behavioral Observation Scale, Behaviorally Anchored Rating Scale[33].

Network performance refers to the level of quality of service of a telecommunications product as seen by the customer. It should not be seen merely as an attempt to get improve network accessibility[41]. For example, in circuit switched networks, network performance is synonymous with the grade of service. The number of rejected calls is a measure of how well the network is performing under heavy traffic loads. Other types of performance measures can include noise and echo[59]. In an Asynchronous Transfer Mode (ATM) network, performance can be measured by line rate, quality of service (QoS), data throughput, connect time, stability, technology, modulation technique and modem enhancements[37][67].

There are many different ways to measure the performance of a network, as each network is different in nature and design. Performance can also be modeled instead of measured; one example of this is using state transition diagrams to model queuing performance in a circuit-switched network[24][25].

Millions of instruction per second (MIPS)[48][66] performance metric have been used to measure the performance of high level code compilation process of a programming language while, performance evaluation of executable high level programs have been measured with the Millions of Floating point per second performance metric[61]. Whatever the metric may be, SMART Test, is frequently used to produce the quality of a performance metric used in performance evaluation. This is also referred to as the specific, measurable, attainable and realistic time test[66][1].

The components of a loosely coupled system are typically designed to operate by generating and responding to asynchronous events. An event notification service is an application-independent infrastructure that supports the construction of event-based systems, whereby generators of events publish event notifications to the infrastructure and consumers of events subscribe with the infrastructure to receive relevant notifications. The two primary services that should be provided to components by the infrastructure are notification selection and notification delivery. Numerous event notification services have been developed for local-area networks, generally based on a centralized server to select and deliver event notifications[41]. Therefore, they suffer from an inherent inability to scale to wide-area networks, such as the Internet, where the number and physical distribution of the service's clients can quickly overwhelm a centralized solution[40]. The critical challenge in the setting of a wide-area network is to maximize the expressiveness in the selection mechanism without sacrificing scalability in the delivery mechanism[53]. The event notification service have been designed and implemented to exhibit both expressiveness and scalability. The service's interface to applications, the algorithms used by networks of servers to select and deliver event notifications, and the strategies used have been discussed accordingly [54].

Quality of Service (QoS) performance metric has been used to evaluate system architecture for self-adaptation. This is done in the Performance Evaluation (Research) Laboratory (PERL) which was opened in July 1993 and started from scratch. It was proved that, Performance of a computer is determined not only by the hardware but also by the workload [36]. This is a confirmation of the fact that performance is determined by the interaction between hardware and software [35]. Today's high-performance workstations, let alone main frame computers, are very complex in design to get improve performance [39]. They commonly employ such techniques as single or multiple CPU pipelines, branch target buffer, instruction prefetching, and single- or two-level cache memories [67]. These techniques take advantage of particular aspects of program behavior [35]. Cache performance has been greatly improved as a result of this performance evaluation researches. Some of these improve caches techniques include: locality of reference which states that a program tends to concentrate its memory references to a small subset of its data space during a relatively short period of time[55][56]; Static and dynamic caching techniques on networks and web search engines[57][58]. Pipeline interlocks depend on how different instructions are executed sequentially. As such, performance of a computer highly depends on its workload. Workload characterization is therefore essential to study performance of today's computers[36][65].

Other performance evaluation techniques in the exterior of experimental measurements, used for evaluating performance of systems can be classified under analytical, simulation and mathematical modeling models [65]. Some of these empirical highly cost effective techniques such as queueing network techniques and the petri net techniques, are not detailed enough to study performance at all levels. Simulation and mathematical modeling techniques generally requires high performance work stations which are capable of running large simulation programs efficiently [66]. Experimental measurements have been conducted on multiple work stations is the past and have yielded good results which has led to significant performance improvement. In one such experiments, a set of benchmark tests were conducted on the IBM RS/6000 model 580 and the HP Apollo 9000 series 700 model 735 [67]. In this light our experiments are design so that we can be able to get the best system specification for running programs in Madonna University Elele Post graduate laboratories. In the past, two major research projects were initiated jointly to evaluate performance; performance evaluation of multiprocessor systems and workload characterization of the database systems[65]. It was found that a major difficulty in evaluating performance of a multiprocessor system is modeling the workload represented by the instruction traces at the register transfer level controlling the hardware[66][67]. In a nutshell, instruction traces cannot be used to evaluate the performance in a multiprocessor environment, because the timing differences in, say, spin locks and even more seriously low level workload. Control flow graph technique has been proposed which can generate different low-level workload in different multiprocessor environments[52].

In a project to characterize a database systems, initially in a uni-processor environment and eventually in a distributed, parallel environment have been proposed. Of particular interest is the performance of instruction and data memory reference behavior. Instruction traces of an existing database system are generated then, the memory reference behavior is studied and a new memory reference technique is proposed. By using both original instruction traces and also newly proposed memory reference behavior technique, the memory subsystem is evaluated for the hardware bottlenecks in both the database and software systems,[64], and for the computer architecture systems[63]. A series of talks have been given around the world to improve of the research on performance evaluation see [67].

In other developments, performance has been carried out for Self-adaptation of systems in which the maximization of the satisfactory requirements under changing environmental conditions was the ultimate goal [62]. In which case, automatically out sourcing of relevant architectural configuration is seen as a key challenge of such systems [35]. Existing performance experiments require a set of adaptive strategies for estimation of noisy effects which lack validation methods leading to erroneous adaptations. Side-effects, is generally a problem in performance evaluation which may not generally be eliminated but controlled in which case,[44] proposed a solution that leverages quality contracts whose accuracy can be separately established and which can be dynamically composed to get a quality prediction of any possible architectural configurations. Equally [39] propose a reactive planning algorithm which exploits quality contracts to dynamically discover unforeseen architectural configurations to support self-adaptation. Validation of their approach to performance evaluation was done using a running HTTP server adapting its architecture with respect to the number and the similarity of incoming requests[66].

Techniques from model-driven software development are useful to analyse the performance of a software architecture during early development stages[64]. Design models of software models can be transformed into analytical or simulation models, which enable analyzing the response times, throughput, and resource utilization of a system before starting the implementation[62].

The Palladio Component Model (PCM) is accompanied by several model transformations, which derive stochastic regular expressions, queuing network models, or Java source code from a software design model. Software architects can use the results of the analytical models to evaluate the feasibility of performance requirements, identify performance bottlenecks, and support architectural design decisions quantitatively[49].
Over the last decade, a lot of research has been directed toward integrating performance analysis into the software development process see [62][63][64]. Traditional software development methods focus on software correctness, introducing performance issues later in the development process[66]. This approach does not take into account the fact that performance problems may require considerable changes in design, for example, at the software architecture level, or even worse at the requirement analysis level. Several approaches were proposed in order to address early software performance analysis [65]. Although some of them have been successfully applied, we are still far from seeing performance analysis integrated into ordinary software development. In this paper, we present a comprehensive review of recent research in the field of model-based performance prediction at software development time in order to assess the maturity of the field and point out promising research directions[51][19].

2. Methodology

Thirty (30) systems were analysed for performance using Response time performance matrix. During this analysis, Systems tuning was done with 20 factors (variables) and the different levels of their response time was obtained with a highly sensitive stop watch aimed at ensuring that a minimum amount of effort, expense and waste is incurred in getting the exact elapsed time. Both software and hardware were considered during the tuning. The geometric, Arithmetic, harmonic and a combine effects of all the means were considered during the evaluation. Missing levels of any factor were calculated in two ways; from the recorded values of the other factors of the same systems by taking the harmonic mean; and by taking the harmonic mean values of that factor when all the 30 systems are considered. A comparison of all the 30 systems was done and presented accordingly using the statistical package for social science (SPSS 17.0).
The computers include laptops, desktops and tablets in the Computer Science Laboratories and offices of the computer science department of Madonna University. The system specifications were recorded before the start of the performance evaluation. The performance of the systems were evaluated without connecting any external hardware and when connected to some external components like external hard disk, flash drives and modems. A database was created to record the factors and levels during system tuning.

3. System Tuning

In order to carryout performance evaluation of the systems the programs and hardware listed in table 1 were used as factors during system tuning.

Table1:Factors used in the evaluation of the sytems

Factor	type	size	decimal	Nominal
SYSTEMS	String	8	0	Nominal
Windows OS Response Time	Numeric	8	2	Scale
MSWORD response time	Numeric	8	2	Scale
MSPPT response time	Numeric	8	2	Scale

MSACC response time	Numeric	8	2	Scale
MSINF response time	Numeric	8	2	Scale
MSOLK response time	Numeric	8	2	Scale
MSNOTE response time	Numeric	8	2	Scale
MSPUB response time	Numeric	8	2	Scale
MSEXC response time	Numeric	8	2	Scale
MSFPG response time	Numeric	8	2	Scale
MSPJT response time	Numeric	8	2	Scale
MSWKSP response time	Numeric	8	2	Scale
IECONN bandwidth	Numeric	8	2	Scale
IENOCONN though put	Numeric	8	2	Scale
DMGER response time	Numeric	8	2	Scale
SCANDIS through put	Numeric	8	2	Scale
DEFGM through put	Numeric	8	2	Scale
MSGRV response time	Numeric	8	2	Scale
MSCLIP response time	Numeric	8	2	Scale
MSPICMG response time	Numeric	8	2	Scale

4. Performance metrics

Performance measurement frequently has its focus to evaluate the effect of changes to a system, such changes might include; Software[8][9][18], Access interface, scheduling parameters, system generation as well as Hardware changes such as amount of main storage, number of channels, Processor rating and power modes. In some instances such evaluation are carried out on fixed benchmark workload[44][48] while in other they may be carried out with real user[23][52]. Purpose of such experiment is, to compare the performance implications of the system changes, to optimize or to tune the system. The selection of the specific performance metric, with the goal of designing experiments on program which could be achieved as quickly and economical as possible while at the same time being able to ensure accuracy and validity of result. In general terms we consider the performance of a complete computer system as a function of: the hardware configurations, the operating system; the associated software support required to run the system and the workload [36].

The performance evaluation of Madonna university computer systems geared towards getting administrators as well as students understand the relationship among constituent systems requirements from which the system was built and to know which factor of the system to change that will improve the system performance. The ideal way of meeting all our objectives would be to have a detailed mathematical model which explicitly displays the nature of the relationship but however, the time required for detail mathematical modeling and the acceptability of the result by all stakeholder. Such a model will then be to study the effect of hardware, software and workload variation on system performance and thus provide a means of predicting and optimizing the system[66]. The experimentation was therefore our best option vis-a-vis the software considered for the performance. The response time, through put and bandwidth metrics were used for the evaluation. Amongst the 20 factors (software) considered for the evaluation, three (3) representing 15% of the software, were linked to the through put performance metric for evaluating the 30 systems, one (1) representing 5% was linked to the bandwidth performance metric, and 16 representing 80% were linked to the response time performance metric.

5. Systems Specifications

The thirty (30) systems evaluated had varied specifications as shown in table 2 below;

Table2: Specifications of the 30 systems in the evaluation experiment.

Sys	LT/DT	CPU MO	CPU RA	MM	HD	OS	OSRA
SYSTEM1	LT	AMD E	2.0GHz	2GB	500	WIN7	64
SYSTEM2	DT	P4 R	3.0 GHz	500MB	250	XP	32
SYSTEM3	LT	I3	1.66 GHz	2GB	300	WIN7	64
SYSTEM4	DT	P 2D	2.3 GHz	500MB	250	XP	32
SYSTEM5	LT	I3	2.9 GHz	2GB	448	WIN8	64
SYSTEM6	DT	P4 R	2.4 GHz	500MB	250	WIN7	64
SYSTEM7	LT	I3	1.6 GHz	2GB	500	XP	32
SYSTEM8	DT	P4	2.2 GHz	896MB	300	WIN7	32
SYSTEM9	LT	I5	3.0 GHz	2GB	500	WIN7	64
SYSTEM10	DT	P4 M	1.7 GHz	500MB	250	XP	32
SYSTEM11	LT	I3	2.88 GHz	2GB	500	WIN8	64
SYSTEM12	DT	P 2DC	2.0 GHz	500MB	250	XP	32
SYSTEM13	LT	I3	2.3 GHz	2GB	500	WIN7	64
SYSTEM14	DT	P4 2DC	2.0 GHz	500MB	250	WIN7	32
SYSTEM15	LT	I3	1.78 GHz	2GB	500	XP	32
SYSTEM16	DT	P4	2.0 GHz	500MB	250	WIN7	32
SYSYEM17	LT	I5	2.3 GHz	2GB	500	WIN8	64
SYSTEM18	DT	P 2DC	2.0 GHz	500MB	250	XP	32
SYSTEM19	LT	I3	1.8 GHz	2GB	500	WIN8	64
SYSTEM20	DT	P4 M	1.6 GHz	500MB	250	WIN7	32
SYSTEM21	LT	I3	1.8 GHz	2GB	500	WIN7	64
SYSTEM22	DT	P4 M	2.0 GHz	500MB	250	XP	32
SYSTEM23	LT	I3	3.0 GHz	2GB	500	WIN7	64
SYSTEM24	DT	P4 M	2.2 GHz	500MB	250	XP	32
SYSTEM25	LT	I3	2.3 GHz	2GB	500	WIN8	64
SYSTEM26	DT	P4 R	1.7 GHz	500MB	250	WIN7	32
SYSTEM27	LT	I3	2.88 GHz	2GB	500	WIN7	64
SYSTEM28	DT	P 2DC	2.0 GHz	500MB	250	XP	32
SYSTEM29	LT	I5	2.4 GHz	2GB	500	WIN7	64
SYSTEM30	DT	P 2DC	2.0 GHz	500MB	250	XP	32

Equal number of laptops and Desktops were used in the investigation. Amongst the desktops in the experiment, 5 random systems where taken each from the General, Programming and internet laboratories to make 15 system. The 15 Laptops used in the experiment came from the post graduate laboratory and the offices. All the systems were running on at least 250mb of main memory and 1.5Ghz processor rating. While all the laptops had windows7 and above, 11 of the desktops had windows XP operating system rated at 2.0 and 32bits. The specification shows that the systems are above average systems and need to perform according to their respective bench mark.

6. Level Recording

In recording the values of the response time for each of the software for which response time was the performance metric, in the evaluation experiment, a database was created in Microsoft

ISSN 2229-5518

Access with fields and attributes been the different levels state above. Figure1 shows a sample input interface for some of the factors. This interface was implemented in Visual basic6.0.

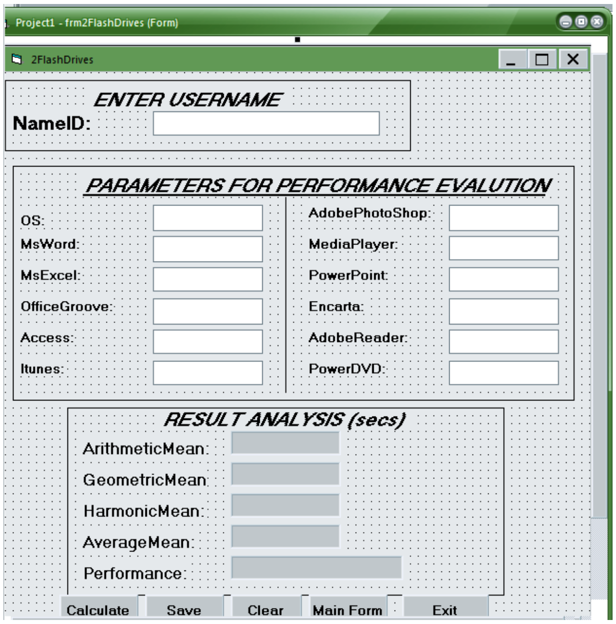

Figure1: Input interface of performance experiment.

This interface was implemented in all the systems and used for the capturing of the values of the selected response time. This program was evaluated with the Millions of floating points operations per second (MFLOPS) metric. The codes were evaluated on the volume of data for which the arithmetic, harmonic and geometric mean were calculated.

7. Results

The capture values of the factors were exported to SPSS for that presentation and analysis. The mean response time for all the factors were calculated as well as their standard deviations and variances. Table 3 presents the mean response time as a functions of all the factors (software) which were ran on the 30 systems.

TABLE 3: Statistics of mean running times of software on windows for different system specs.

	N	Minimum	Maximum	Mean		Std. Deviation	Variance
	Statistic	Statistic	Statistic	Statistic	Std. Error	Statistic	Statistic
MSCLIP	4	1.68	2.32	2.0300	.15722	.31443	.099
MSINF	14	.79	4.30	2.3264	.22814	.85364	.729
MSPJT	8	1.29	3.48	2.5087	.28058	.79361	.630
MSPICMG	4	1.35	4.62	2.5275	.72450	1.44901	2.100
MSEXC	30	1.20	5.42	2.5870	.17316	.94843	.900
MSPPT	28	1.13	9.41	3.2339	.36701	1.94203	3.771
MSWORD	30	1.05	8.20	3.2813	.34189	1.87259	3.507
MSACC	30	1.20	12.02	3.7643	.48794	2.67255	7.143

MSPUB	26	1.75	13.50	3.8131	.55964	2.85363	8.143
MSWKSP	2	3.80	4.42	4.1100	.31000	.43841	.192
IENOCONN	26	1.30	26.22	4.3608	.92172	4.69989	22.089
MSNOTE	22	1.52	6.65	4.4650	.33748	1.58293	2.506
MSOLK	26	1.92	13.80	5.3835	.56594	2.88573	8.327
MSFPG	6	1.51	12.00	5.9767	1.89628	4.64492	21.575
DEFGM	10	2.49	24.23	9.1180	2.46922	7.80836	60.970
IECONN	18	1.06	62.41	11.7239	3.51093	14.89559	221.879
MSGRV	13	2.23	29.63	11.8708	2.77002	9.98745	99.749
DMGER	20	1.35	301.04	28.5825	17.45982	78.08267	6096.903
WINOS	22	3.15	113.25	35.9077	5.15492	24.17874	584.611
SCANDIS	18	.33	1250.27	178.2006	87.09629	369.51825	136543.734
Valid N (listwise)	0						

In Table 4, we presented the calculation of geometric, arithmetic, harmonic means and combined mean of all the program response time running as recorded for the 30 systems. The arithmetic mean was calculated using the formula, $\frac{\sum_1^{30} p_i}{30}$ where p_i is the response time for the i^{th} program. The Geometric mean was calculated using the formula, $\sqrt[30]{\prod_1^{30} p_i}$. The harmonic mean was calculated based on the formula, $\frac{\sum_1^{30}\left(1/p_i\right)}{30}$. The combined mean is the average value of the three means, $\frac{\frac{\sum_1^{30} p_i}{30} + \sqrt[30]{\prod_1^{30} p_i} + \frac{\sum_1^{30}\left(1/p_i\right)}{30}}{3}$. We used the combined mean for performance evaluation to reduced the effects of wrong judgment when only, arithmetic, geometric or harmonic mean is used. For example from the table, if arithmetic mean is used to evaluate program performance for the 30 systems, scandis(132s),winos(33.9s), Dmger(23.1s) are the worse three program. On the other hand, if harmonic mean is used, winos(22s),msgrv(7.2s),Defgm(5.9) are the last three program. Geometric mean project, winos(28.6s),scandis(21.5s) and msgrv(8.5s) are the last three programs.

Table4: Program evaluation by Arithmetic, Geometric, Harmonic and Combine means

SYSTEM	WINOS	MSWORD	MSPPT	MSACC	MSINF	MSOLK	MSNOTE	MSPUB	MSEXC	MSFPG	MSPJT	MS WKSP	IECONN	IENO CONN	DMGER	SCANDIS	DEFGM	MSGRV	MS CLIP	MS PICMG
SYSTEM25	15.25	4.25	5.3	4.2	4.3	2.63	1.52	3.3	4.45	4.7	2.4	4.1	7.6	1.54	1.56	0.33	6.8	2.59	2	2.3
SYSTEM9	3.15	2.52	3.75	3.66	2.2	3	4.1	7.61	2.52	3.65	2.4	4.1	4	3.72	4.3	0.6	8.02	8.7	2	2.3

SYSTEM27	15.02	1.61	2.8	1.2	2.2	4.8	4.1	2.2	1.3	4.7	2.4	4.1	1.06	3.2	1.35	63.6	2.65	8.7	2	2.3
SYSTEM11	16.59	1.71	1.13	1.4	2.2	4.8	4.1	2.22	1.2	4.7	2.4	4.1	1.06	3.83	12.2	63.6	2.88	8.7	2	2.3
SYSTEM28	23.05	1.05	2.8	1.7	2.2	4.8	4.1	2.33	1.6	4.7	2.4	4.1	4.05	2.25	1.45	63.6	2.5	8.7	2	2.3
SYSTEM1	46.26	2.6	3.02	3.17	2.2	7.61	4.1	3.67	3.76	3.8	2.4	4.1	4.29	2.3	7.3	0.6	8.02	8.7	2	2.3
SYSTEM30	28.3	1.87	1.62	1.87	1.68	2.82	2	2.72	1.92	4.7	1.87	4.1	7.6	1.5	12.2	63.6	6.8	2.23	2	2.3
SYSTEM21	25.45	3.15	1.71	2.38	2.11	5.13	2.2	1.75	1.8	4.7	2.4	4.1	7.1	3.24	4.95	6.12	6.8	24.16	2	2.3
SYSTEM24	28.3	2.23	2.75	2.69	2.59	4.19	5.66	3.3	1.99	4.7	2.4	4.1	7.6	3.4	3.76	4.17	6.8	4.61	2	2.3
SYSTEM12	22.05	1.1	1.29	1.63	2.2	4.8	4.1	2.37	1.98	4.7	2.4	4.1	4.03	2.52	12.2	63.6	2.49	8.7	2	2.3
SYSYEM17	113.25	1.82	1.54	1.78	2.2	1.92	4.45	2.33	3.18	4.7	1.29	4.1	3.7	4.5	3.42	2.5	6.8	8.7	1.7	2.31
SYSTEM8	28.3	1.87	1.62	1.87	1.68	2.82	2.6	2.72	1.92	4.7	1.87	4.1	7.6	1.3	12.2	63.6	6.8	8.7	2	2.3
SYSTEM7	28.3	2.79	1.77	3.35	1.76	2.66	3.32	1.8	2.73	4.7	2.51	4.1	7.6	1.89	12.2	63.6	6.8	3.76	2	2.3
SYSTEM29	28.3	2.79	1.77	3.35	1.76	2.66	3.32	1.8	2.73	4.7	2.51	4.1	7.6	1.89	12.2	63.6	6.8	3.76	2	2.3
SYSTEM20	39.6	1.4	3.13	2.56	2.71	8.72	3.23	4.54	2.65	4.7	2.4	4.1	9.75	7.15	1.6	3.74	19.43	8.7	2	2.3
SYSTEM15	49.11	4.11	2.69	4.44	2.2	6.65	4.75	1.81	2.42	4.7	2.4	4.1	7.6	3.88	3.69	3.17	6.8	11.41	2	2.3
SYSTEM16	49	4.02	2.58	4.3	2.2	6.55	6.65	1.79	2.32	4.7	2.4	4.1	7.6	3.82	3.57	3.09	6.8	11.42	2	2.3
SYSTEM18	45.73	2.52	2.62	3.35	2.2	3.35	4.92	3.05	5.42	4.7	3.22	4.1	5.22	3.32	4.22	3.32	6.8	8.7	2.3	4.62
SYSTEM26	19.32	2.15	2.19	2.23	2.44	4.75	5.66	3.3	1.9	4.7	2.4	4.1	7.6	3.6	12.2	63.6	6.8	8.7	2	2.3
SYSTEM23	28.3	4.15	2.48	4.45	0.79	6.68	4.96	3.3	2.62	4.7	2.4	4.1	7.6	3.6	12.2	63.6	6.8	8.7	2	2.3
SYSTEM19	39.39	3.47	6.67	2.96	2.8	2.76	4.62	3.72	2.96	4.7	2.4	4.1	10.18	6.58	3.77	6.45	24.23	8.7	2	2.3
SYSTEM10	11.55	2.76	2.8	11.22	2.2	4.6	4.1	13.5	2.76	12	2.4	4.1	4.39	26.22	3.89	18.09	6.04	8.7	2	2.3
SYSTEM6	28.3	7.64	9.41	3.87	2.2	3.2	4.1	3.62	3.61	1.51	2.4	4.1	7.6	3.6	12.2	63.6	6.8	6.4	2	2.3
SYSTEM22	30.16	8.2	5.46	6.37	1.87	10.6	6.56	4.07	2.49	4.7	2.4	4.1	15.02	5.24	2.33	20.44	6.8	29.63	2	2.3
SYSTEM2	26.02	2.8	4.5	12.02	2.2	13.8	4.1	11.32	2.9	11.76	2.4	4.1	18.09	4.39	6.01	3.89	14.92	8.7	2	2.3
SYSTEM5	28.3	4.6	5.46	8.2	2.2	4.44	3.62	4.46	3.94	3.14	2.4	4.1	7.6	3.6	12.2	63.6	6.8	9.12	2	2.3
SYSTEM13	25.66	3.58	1.34	2.22	2.2	5.24	4.81	2.23	1.2	4.7	2.4	4.1	10.13	5.56	6.2	942.05	6.8	15.6	1.9	1.35
SYSTEM4	56.7	3.4	2.76	2.18	3.12	7.26	6.56	3.56	2.36	4.7	3.48	3.8	62.41	2.59	203.1	451.42	6.8	8.7	2	2.3
SYSTEM3	72.18	4.18	3.55	1.94	2.96	8.26	6.36	3.65	2.88	4.7	3.32	4.42	34.08	3.83	301	487.36	6.8	8.7	2	2.3
SYSTEM14	45.48	8.1	5.64	6.37	2.2	7.68	6.36	4.3	2.1	4.7	2.4	4.1	12.47	3.72	4.13	1250.27	6.8	29.63	2.3	1.83
GMEAN	28.6	2.9	2.8	3.1	2.2	4.7	4.1	3.2	2.4	4.7	2.4	4.1	7.2	3.5	6.9	21.5	6.7	8.5	2	2.3
AMEAN	33.9	3.3	3.2	3.8	2.3	5.3	4.4	3.7	2.6	5	2.4	4.1	10.1	4.3	23.1	132.4	7.6	10.1	2	2.3
HARMEAN	22	2.5	2.5	2.7	2.1	4.3	3.9	2.9	2.3	4.4	2.4	4.1	5.2	3	4.3	3.3	5.9	7.2	2	2.3
CMEAN	28.2	2.9	2.8	3.2	2.2	4.8	4.1	3.3	2.4	4.7	2.4	4.1	7.5	3.6	11.4	52.4	6.7	8.6	2	2.3

Table 5 presents the geometric, arithmetic, harmonic and combined means of the 30 systems based on the programs ran on them. The combined mean was taken to be the harmonic average of the three means to reduce the complexity of the figures.

Table 5: Comparing 30 system using geometric, arithmetic, harmonic and combined means

SYSTEM	GEOMETRIC	ARITHMETIC	HARMONIC	COMBINED

	MEAN	MEAN	MEAN	MEAN
SYSTEM25	90673.2	4.1	2.1	4.17
SYSTEM9	158179.2	3.8	2.7	4.74
SYSTEM27	119882.5	6.6	2.4	5.28
SYSTEM11	294820.4	7.2	2.5	5.57
SYSTEM28	268894.4	7.1	2.6	5.71
SYSTEM1	752124.7	6.1	2.9	5.9
SYSTEM30	419981.3	7.7	2.7	6
SYSTEM21	772076.9	5.7	3.1	6.02
SYSTEM24	713305	5	3.4	6.07
SYSTEM12	613431.3	7.5	2.8	6.12
SYSYEM17	252847.8	8.8	2.7	6.2
SYSTEM8	880511.2	8	2.8	6.22
SYSTEM7	1505703	8	3.1	6.7
SYSTEM29	1505703	8	3.1	6.7
SYSTEM20	2473838.3	6.7	3.4	6.77
SYSTEM15	2043984.4	6.5	3.5	6.82
SYSTEM16	2145029.1	6.6	3.5	6.86
SYSTEM18	2406673.6	6.2	3.8	7.07
SYSTEM26	4721576	8.1	3.6	7.48
SYSTEM23	8852773.2	8.8	3.5	7.51
SYSTEM19	8803137.1	7.2	4	7.71
SYSTEM10	18919622.2	7.3	4.2	8
SYSTEM6	13694691	8.9	3.9	8.14
SYSTEM22	42655094.2	8.5	4.3	8.57

SYSTEM2	45451050	7.9	4.5	8.6
SYSTEM5	26032010.8	9.1	4.3	8.76
SYSTEM13	12784455.2	52.5	3.1	8.78
SYSTEM4	602488380.7	42	4.2	11.45
SYSTEM3	1104910401	48.2	4.4	12.1
SYSTEM14	345074804.4	70.5	4.4	12.42

Table 6, below presents the 30 systems with the mean response rate of the 20 programs executed during the experiments.

Table6:Program response rate

SYSTEM	WINOS	MSWORD	MSPPT	MSACC	MSINF	MSOLK	MSNOTE	MSPUB	MSEXC	MSFPG	MSPJT	MS WKSP	IECONN	IENO CONN	DMGER	SCANDIS	DEFGM	MSGRV	MS CLIP	MS PICMG
GMEAN	16.7	2.9	2.7	3.1	2.7	4.2	3.7	3.1	2.4	3.2	3	3.1	4.8	3.4	4.1	5.8	4.5	4.7	2.9	2.9
AMEAN	28	3.3	3.1	3.8	2.9	4.9	4.1	3.6	2.6	3.7	3.2	3.3	8.2	4.2	19.9	107.9	6.8	7	3.1	3.2
HARMEAN	8.1	2.5	2.4	2.7	2.4	3.6	3.3	2.8	2.3	2.9	2.7	2.9	3.4	3	2.9	2.1	3.6	3.6	2.7	2.7
CMEAN	17.6	2.9	2.7	3.2	2.7	4.2	3.7	3.2	2.4	3.3	3	3.1	5.5	3.5	9	38.6	5	5.1	2.9	2.9

Table 7 below represent the response rates of all the 30 systems calculated from their geometric, arithmetic, harmonic and combined means response time of the 30 systems.

Table 7: Systems Response Rate

SYSTEM	GMEAN	AMEAN	HMEAN	CMEAN
SYSTEM1	463557.5	5.8	2.8	5.67
SYSTEM2	478626217	8.8	6.3	11.01
SYSTEM3	1.56E+09	48.2	4.8	13.1
SYSTEM4	684129692	41.8	4.4	11.94
SYSTEM5	6417175.8	4.9	4.7	7.2
SYSTEM6	905648.6	4.2	3.7	5.9
SYSTEM7	7716.3	2.5	2.4	3.67
SYSTEM8	737	2	1.9	2.92
SYSTEM9	75217.1	3.4	2.6	4.42
SYSTEM10	123863262	7.9	5.5	9.73
SYSTEM11	947.8	2.6	1.8	3.19
SYSTEM12	4604.2	3.1	2.1	3.75
SYSTEM13	7576577.8	52.2	3	8.51
SYSTEM14	697254259	70.7	4.9	13.75
SYSTEM15	2740400.2	6.4	3.8	7.15

ISSN 2229-5518

SYSTEM16	1475291.4	6.6	3.8	7.23
SYSYEM17	45052.7	8.1	2.4	5.55
SYSTEM18	1265479.8	5.8	3.7	6.78
SYSTEM19	19917093	7.4	4.6	8.51
SYSTEM20	2465417.2	6.6	3.5	6.86
SYSTEM21	1973849.3	6.6	3.3	6.6
SYSTEM22	376413543	10.2	5.4	10.59
SYSTEM23	16591.9	2.9	2.4	3.94
SYSTEM24	114463.2	3.3	3.1	4.8
SYSTEM25	3975.9	3.1	1.7	3.29
SYSTEM26	94582.8	3.8	2.9	4.93
SYSTEM27	804.5	2.5	1.8	3.14
SYSTEM28	3221.3	3.1	2	3.65
SYSTEM29	7716.3	2.5	2.4	3.67
SYSTEM30	4332.3	3.4	2.1	3.89

Table 7 displays the systems with their corresponding arithmetic, geometric, harmonic and combined mean response rate. The table is aimed at comparing the response time rate for the systems in order to determine the best among the systems.

8. Discussion

From the results, presented in table3, all the 30 systems were evaluated for performance using the response time metric with MSEXC, MSWORD and MSACC. MSEXC performed better on all the 30 systems with a mean value (μ) of 2.59s, standard deviation (δ) of 0.95s and variance (σ) of 0.900s. The worse run of the program recorded a response time of 5.42s while the best run of the program 1.2s. Amongst the three programs, MSACC performed the worse with μ of 3.76s, δ of 2.6s and σ of 7.1s. The worse run of MSACC responded at 12.0s while the best run responded at 1.2s. MSWORD responded best in one of the system with a response time of 1.05s in which case if that system was the only system under consideration, then MSWORD will record the best performance. This goes to confirm the fact that performance varies according to the factors and design objective of the systems[6]. Twenty-eight (28) systems were evaluated only with the MSPPT program for which the best run of the program was 1.13s and worse run was 3.2s. The MSPPT program however, performed better than MSWORD with μ of 3.23s, δ of 1.94s and σ of 3.77s as against μ of 3.28s, δ of 1.87s and σ of 3.5s. The table also shows that 26 systems were evaluated with MSPUB, IENOCONN, and MSOLK and among these programs, MSPUB perform best with μ of 3.8s, δ of 2.85s and σ of 8.14s while MSLOK performed the worse with μ of 5.38, δ of 2.89 and σ of 8.3. From the table, it is shown that 22 systems were evaluated with MSNOTE and WINOS, 20 systems were evaluated with DMGER, 18 systems were evaluated with IECONN and SCANDIS, 14 systems were evaluated with MSINF, 13 systems were evaluated with MSGRV, 10 system were evaluated with DEFGM, 8 systems were evaluated with MSPJT, 6 systems were evaluated with MSFPG, 4 systems were evaluated with MSCLIP and MSPICMG and finally, 2 systems were evaluated with MSWKSP.

Table 4 shows the performance of the programs as well as the systems using the response time, through put, bandwidth and the rate metrics. All the system were evaluated with all the programs with the exception that some of the values of the programs were evaluated using the rate metric as a function of the harmonic mean values of the program performance. The results presented reveal that Microsoft Clip (MSCLIP) was the best performing program among the 20 software used responding at a mean time of 2.0s. Microsoft Infor (MSINF) was the second best performing program (2.2s) while three programs, MSPICMG, microsoft Project (MSPJT) and MSPPT were the third, fourth and fifth respectively, and all responding at 2.3s, 2.4s and 2.8s. The recent performance evaluation technique implemented at the software architecture level[44] is certainly the reason why the first two performing software are MSCLIP and MSINF. These two software have not changed significantly from their initial architecture, the mode of operation and their uses have also not had any significant change in recent years. In table3 we saw that the first 5 best responding programs were MSCLIP, MSINF, MSPJY, MSPICMG and MSEXC. This is because all the systems were not considered in table 3 as in table 4. The mean used in the comparison in table 3 was the arithmetic mean only while the combined mean involving the average of the arithmetic, geometric and harmonic mean is the subject of comparison in table4. This confirms the fact that using only arithmetic or geometric or harmonic mean may not give the best evaluation of performance [66][67]. According to table3, Microsoft Access (MSPUB) was 8th in performance ITNOCONN was 9th in performance as oppose to table 4 where both Access and publisher performance were the same and occur in the 6th position of the performance chart.

Table 5 shows program performance of the systems with regards the response time. The combine mean response time is presented. Figure 2 below is a chart that summaries our performance result as drawn from table5. As the chart show the smallest combined mean response time was recorded for systems 25 whose specification as shown in table 2 indicate that the system is a core I3 laptop with 2.3GHz processor rating, 2Gb main memory, 500Gb Hard disk, 64bits windows8 operating system. the second, third and fourth systems are system9, system27 and system 11. All these system being laptops computers. our of the 30 systems, four (4) desktops computers make up the top ten (10) performance list. These desktop systems are system 28, system 30, system 24 and system 12 which are respective in positions 5, 7, 9 and 10 on the list. A look at the specification of system 28 which is 5th on the performance list shows that it is a Pentium Dou core, rated 2.0GHz, 500Mb Main memory, 250Gb hard disk and 32 bit windows Xp operating system. This goes to confirm the question many professionals have asked about their old system performing better than their new system even though in terms of cost and specifications the new system had higher values. Figure 4 and figure 5 shows the specification of two systems, whose performance does not reflects their specifications as envisage by Littlecloud in 2010[57]. The

ISSN 2229-5518

explanation of old system performing more than the new system, or a system of lower specification performing more than a system of higher specification is the benchmark programs. The comparison was done with respect to software or hardware that were considered during the system design process. These software or hardware will run at their benchmark performance when installed on the computer system they were designed for. As a result, the program used in comparing the performance of the system in figure4 and figure5 are programs that were considered when designing the old system (figure4) but assumed when designing the new system (figure5).

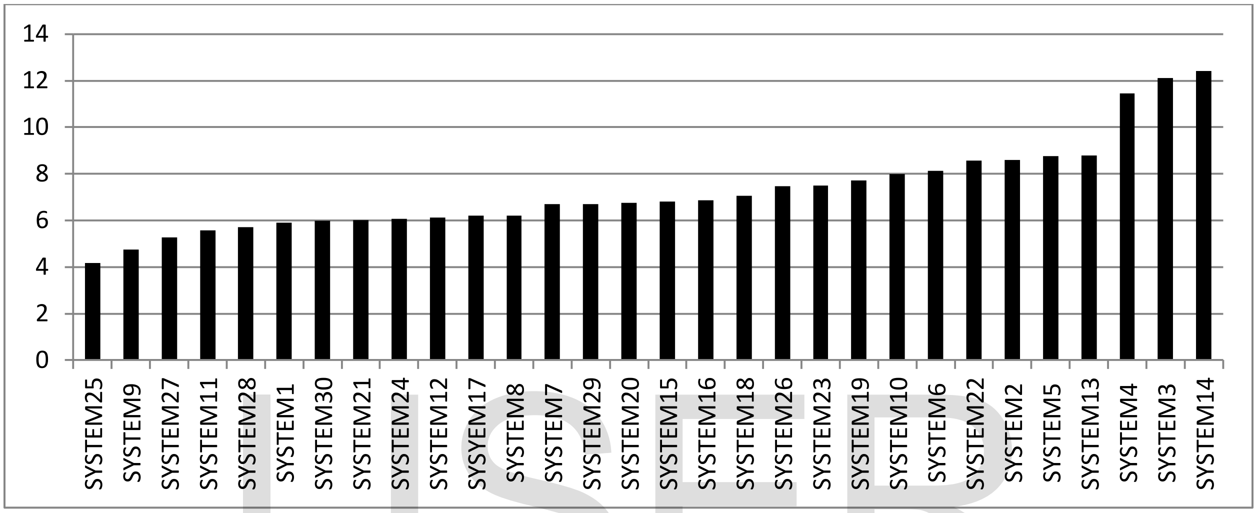

Figure 2: Performance Chart of the 30 systems on combined mean response time.

As a matter of fact, figure2 shows that the worse of the 30 systems was system 14 whose combined response time was above 12 seconds. The laptop represented by system3 was close in performance to system 14 coming in the 29th position. The specification of this system shows that it is suppose to perform better in which case we can only architectural level performance evaluation was not done in the construction of the system or the low rating of the processor (1.66GHz) may be responsible for the low performance. Wrong installations of software have been associated to performance of this nature[11].

Table6 presents the program response rate performance comparison. The rate of response was calculated from the ratio of response time of program to the bench mark response time which was taken to be the global average of response time (the sum of all the combined mean's of programs divided by 20, the number of programs). According to the table, MSEXC was the best performing program in terms of rate. MSINF and MSPPT were second in performance while MSWORD, MSCLIP and MSPICMG were the third in the line of performance. The worse performing program was SCANDIS and WINOS. Like for the response time performance metric as discussed in table4, Microsoft excel (MSEXC) was the best performing program among the 20 software used responding at a mean rate of 2.4 similar to the 2.4 using response time metric. Microsoft Power point (MSPPT) was the second best performing program (2.7) while three programs, microsoft word (MSWORD), microsoft clip (MSCLIP) and MSPIC all responding at 2.9. The recent performance evaluation technique implemented at the software architecture level[44] is certainly the reason why the first two performing software are MSEXC and MSPPT. These two software have not changed significantly from their initial architecture, the mode of operation and their uses have also not had any significant change in recent years. In table3 we saw that the first 5 best responding programs were MSCLIP, MSINF, MSPJY, MSPICMG and MSEXC. This is because all the systems were not considered in table3 as in table 4 and table6. The mean used in the comparison in table 3 was the arithmetic mean only while the combined mean involving the average of the arithmetic, geometric and harmonic mean is the subject of comparison in table4 and table6. This confirms the fact that using only arithmetic or geometric or harmonic mean may not give the best evaluation of performance [66]. According to table3, Microsoft Access (MSACC) was 8th in performance while microsoft publisher was 9th in performance as oppose to table 4 where both Access and publisher performance were the same and occur in the 6th position of the performance chart.

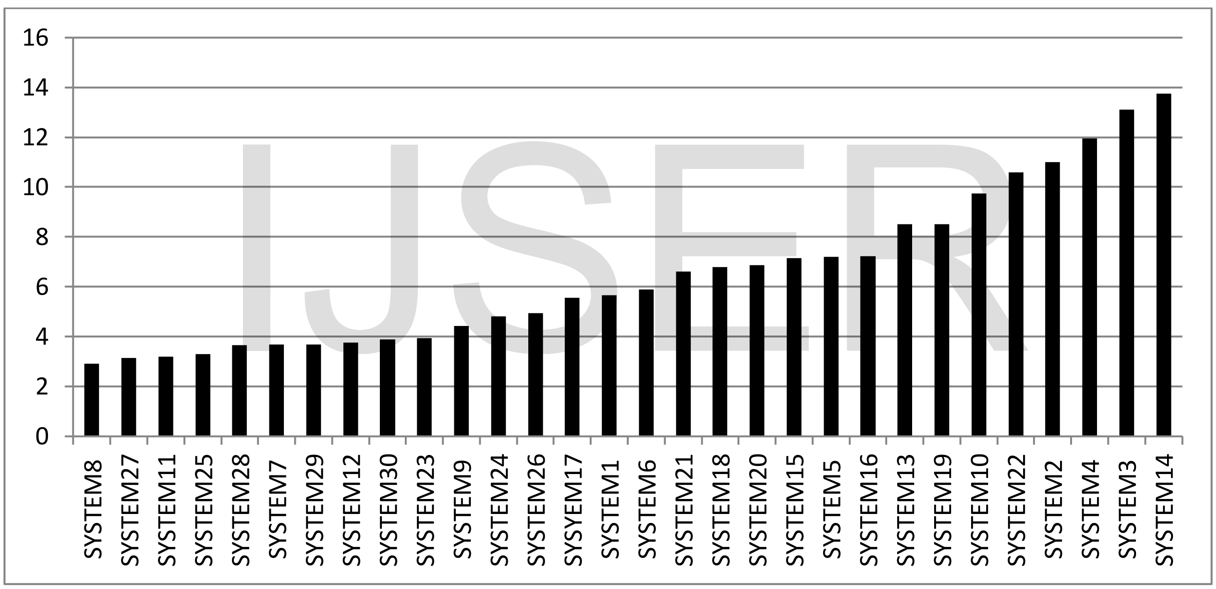

Figure 3: Performance chart of the 30 systems on combined mean response rate.

Table7 unlike table5 discusses the performance evaluation of the 30 systems with respect to the rate metric. From the table the arithmetic, geometric, harmonic and combined means are calculated for all the 30 systems response rate. The summary of the systems performance validated from the combined mean response rate is presented in figure3. The bar chart shows that the desktop system8 is the best performing system, consequently, it can be inferred that most of the programs executed on the systems were compatible with the specification of the system. Using the response time metric, the best performing system was system 25 while using the response metric, the best performing system was system8. This goes to confirm one of the aim of performance evaluation vis-a-vis to select the best possible performance metric for the evaluating any system[65][66]. However, system25, system11 and system28 all make the first 5 systems on the list as was the case with the response time metric shown in figure2. Amongst the systems that made the first 10 in the list using the response time performance metric, four (4) were not in the top 10 performance list using response rate performance metric these four systems were; system1, system24, system21 and system9 with 3 of them being laptops and one desktop. Worth noting is the fact that the systems; system4, system3 and system14 are the last three performing system using both metric. These experiments and results will go a long way to explain to professionals and users of computer systems that the specification of a system shouldn't be considered as the exclusively as the performance measure of that system. The programs installed in the systems must be compatible with the system specification for maximum system performance. An old computer system will perform best with its complimentary software so as much as a new computer system will perform best with its complimentary software. Therefore, starcraft [57] software which runs better on the system represented in figure4 which is lower in specification to the system represented on figure5 is more compatible with that system at the present state of the system.

Acknowledgement

Thanks for support received from Abondem Atabong leke during the conducting the experiments and the management of Madonna University Computer Laboratories.

References

[1] Raj, J. (1991). Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling, Wiley Computer Publishing, John Wiley & Sons, Inc. U.S.A.

[2] Mcdavid J. C. Irene, H. and Laura, R. L. H. (2013). Program Evaluation and Performance Measurement :An Introduction to Practice Second Edition, Sege Publication Inc. Canada.

[3] Lyord G. W. and Connie, U. S.(1998) Performance evaluation of software architectures In Proceedings of the 1st international workshop on Software and performance, New York:164-177.

[4] Hatry, H. P. (2006). Performance measurement: Getting results (2nd ed.). Washington, DC: Urban Institute Press.

[5]P.C.C.(2014). Comparing new computer and Old Computers, Performance Computer Community, Retrieved 23-12-14 from http://www.overclockers.com/forum/showthread/653896-new-computer-slower-than-old-one.

[6]Atabong, T. A., Agu, S., Nitah, E. etal. (2012). Complete Computer Science:A Reference to digital passover, Aba, Ark of Wisdom Publisher.

[7]Chen, H.-T. (1996). A comprehensive typology for program evaluation. Evaluation Practice, 17(2), 121–130.

[8]Mattson,M., Grahn, H. and Frans, M.(2014). Software Architecture Evaluation methods for performance maintainability, Testability and Portability.

[9]CMU(2014). Evaluate and Improve software architectures relative to quality attribute goals, Batimore, Software Engineering institute.

[10]Bakvis, H., & Juillet, L. (2004). The horizontal challenge: Line departments, central agencies and leadership. Ottawa, Ontario, Canada: Canada School of Public Service.

[11]Feller, I. (2002). Performance measurement redux. American Journal of Evaluation, 23(4), 435–452.

[12]Chelimsky, E. (1997). The coming transformations in evaluation. In E. Chelimsky & W. R. Shadish (Eds.), Evaluation for the 21st century: A handbook (pp. ix–xii). Thousand Oaks, CA: Sage.

[13]Government Social Science Research Unit. (2007). Magenta book: Guidance notes on policy evaluation. Retrieved from http://www.civilservice.gov.uk/wp-content/uploads/2011/09/magenta_book_complete_sep_2007.pdf

[14]Campbell, S., Benita, S., Coates, E., Davies, P., & Penn, G. (2007). Analysis for policy: Evidence-based policy in practice. London, England: Government Social Research Unit, HM Treasury.

[15] McDavid, J. C., & Huse, I. (2006). Will evaluation prosper in the future? Canadian Journal of Program Evaluation, 21(3), 47–72.

[16]Moynihan, D. P. (2008). The dynamics of performance management: Constructing information and reform. Washington, DC: Georgetown University Press.

[17]Lloyd, G. W. and Connie, U. S.(2002). PASASM:A method for the performance assessment of software architectures, Colorado, Software Engineering Research and Performance Engineering Services.

[18]Simonetta, B., Roberto, M., Moreno, M.( ). Performance Evaluation of software architectures with queuing network models,

[19] Perrin, B. (1998). Effective use and misuse of performance measurement. American Journal of Evaluation, 19(3), 367–379.

[20] Mayne, J. (2006). Audit and evaluation in public management: Challenges, reforms and different roles. Canadian Journal of Program Evaluation, 21(1), 11–45.

[21] Newcomer, K. E. (1997). Using performance measurement to improve public and nonprofit programs. In K. E. Newcomer (Ed.), New directions for evaluation, (Vol. 75, pp. 5–14). San Francisco, CA: Jossey-Bass.

[22] McDavid, J. C. (2001). Program evaluation in British Columbia in a time of transition: 1995–2000. Canadian Journal of Program Evaluation, 16(Special Issue), 3–28.

[23]Atabong, T., Eleanu, O. F. etal(2010). Performance Evaluation of madonna University Wireless network, Trop. Journal of Biomedical and Allied Sci. Res. 4:(1):

[24]Reiser, M., and Lavenberg, S. S.(1980). Mean value analysis of closed multichain queuing networks, Journal of the ACM, 27:313-322.

[25] Ching-Tarng, H. and Simon, S. L.(1987). "Two Classes of Performance Bounds for Closed Queueing Networks," Performance Evaluation, vol. 7, no. 1, pp. 3-30, 1987.

[26] McDavid, J. C., & Huse, I. (2012). Legislator uses of public performance reports: Findings from a five-year study. American Journal of Evaluation, 33(1), 7–25.

[27] Derek, L. E. and Kenneth, C. S.(1983)."Performance Bound Hierarchies for Queueing Networks," Transactions On Computer Systems, 1 (2): 99-115

[28] Smith, C. U. and Williams, L. G. (1998). "Performance Engineering Evaluation of CORBA-based Distributed Systems with SPEED," In Computer Performance Evaluation, Lecture Notes in Computer Science, vol. 1469, R. Puigjaner, N. N. Savino and B. Serra, ed., Berlin, Springer- Verlag,: 321-335.

[29] Smith, C. U. and Williams, L. G. (1998). "Performance Engineering Evaluation of Object-Oriented Systems with SPEED," In Computer Performance Evaluation: Modelling Techniques and Tools, Lecture Notes in Computer Science, vol. 1245, R. Marie, B. Plateau, M. Calzarossa and G. Rubino, ed., Berlin, Springer- Verlag: 135-154.

[30] Mayne, J. (2008). Building an evaluative culture for effective evaluation and results management. Retrieved from http://www.cgiar-ilac.org/files/publications/briefs/ILAC_Brief20_Evaluative_Culture.pdf

[31] Smith,C. U. (1990). Performance Engineering of Software Systems, Reading, MA, Addison-Wesley.

[32] Klein, M. and Kazman, R.(1999). "Attribute-Based Architectural Styles," Technical Report No. CMU/SEI-99-TR-022, Software Engineering Institute, Carnegie-Mellon University, Pittsburgh, PA, October, 1999.

[33] Mayne, J. (2011). Contribution analysis: Addressing cause and effect. In K. Forss, M. Marra, & R. Schwartz (Eds.), Evaluating the complex: Attribution, contribution, and beyond: Comparative policy evaluation (Vol. 18, pp. 53–96). New Brunswick, NJ: Transaction.

[34] Smith, C. U. and Williams, L. G. (2002). Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software, Reading, MA, Addison-Wesley.

[35] Pooley, R. and King, P.(1990). "The Unified Modeling Language and Performance Engineering," *IEE Proceedings-Software*, **146** (1):2-10.

[36]Johannes, L., Shikharesh, M., Gabriele, K. and Günter, H.(1997)."Performance Bounds for Distributed Systems with Workload Variabilities and Uncertainties," Parallel Computing, 22(13):1789-1806.

[37] Castaldi, M., Inverardi, P., and Afsharian, S.(2002). "A case study in performance, modifiability and extensibility analysis of a telecommunication system software architecture," *Proc. 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems:*281-290.

[38] Smith, C. U. and Williams, L. G."Performance Characteristics of Common Architectural Styles: Pipe-and-Filter and Client-Server," manuscript in preparation

[39] Hennessy, J. L. and Patterson, D. A. (2003). *Computer Structures and Architecture*, 3rd Ed., Morgan Kaufmann Publishers, Inc.

[40] Perkins, C. E., Royer, E. M., Das, S. R. and Marina, M. K. (2001). "Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks," *IEEE Personal Communications*, 16-28.

[41] Garcia-Luna-Aceves, J. J. and Fullmer, C. L. (1998). "Performance of floor acquisition multiple access in ad-hoc networks ," *Proc. IEEE Symp. on Computers and Communications (ISCC '98), 30 Jun-2 July*,63 -68.

[42] Fullmer, C. L. and Garcia-Luna-Aceves, J. J. (1997), "Complete single-channel solutions to hidden terminal problems in wireless LANs," *Proc. ICC*, 575-579.

[43] Cote, G. and Winger, L. (2002)."Recent Advances in Video Compression Standards," *IEEE Canadian Review*, Spring.

[44] Franck, C., Hui, S., Xiangping, C., Gang, H., Hong, M. (2010). Quality of Software Architectures in Conference Proceedings of QOSA:102-118.

[45] Funnell, S., & Rogers, P. (2011). Purposeful program theory: Effective use of theories of change and logic models. San Francisco, CA: Jossey-Bass.

[46] Osborne, D., & Gaebler, T. (1992). Reinventing government: How the entrepreneurial spirit is transforming the public sector. Reading, MA: Addison-Wesley.

[47]Auditor General of British Columbia and Deputy Ministers' Council. (1996). Enhancing accountability for performance: A framework and an implementation plan—Second joint report. Victoria, British Columbia, Canada: Queen's Printer for British Columbia.

[48]Mostafa, A., and Hesham E. (2005). Fundmentals of computer organisation and architecture, John Wiley and Sons publishers, New Jessey.

[49] Heiko, K., Jens, H., Steffen, B., Ralf, R.( ). Evaluating Performance of Software Architecture Models with the Palladio Component Model

[50]Abanda, T. A. and Eberendu-Ogu, A.C.(2010). Philosophy of Computer Science and Formal Languages, Aba, Ark of Wisdom Publishers.

[51] Simonetta, B., Antinisca, D., and Marta, S. (2004). Model-Based Performance Prediction in Software Development: A Survey , *IEEE Transactions on Software Engineering - TSE* , **30**(5):295-310.

[52] Makoto Kobayashi, Shietung Pen and Sergey V. Ten. Professors Peng and Ten belong to the Distributed Parallel Processing Laboratory.

[53] Katz, R. (1996). "The Case for Wireless Overlay Networks," Invited talk at the ACM Federated Computer Science Research Conferences, Philadelphia.

[54] Lin, Y. B. (1997). "Performance Modeling for Mobile Telephone Networks," *IEEE Network*, 63-68.

[55] Carvalho, M. M. and Garcia-Luna-Aceves, J. J. (2003). "Delay analysis of IEEE 802.11 in single-hop networks," *Proc. IEEE Int. Conf. on Network Protocols*,146-155.

[56] Guba, E. G., & Lincoln, Y. S. (1989). *Fourth generation evaluation*. Newbury Park, CA: Sage.

[57] LittleClooud (2010). New computer slower than old one, Overclockers, Retrieved 22 December, 2014 from www.overclockers.com/forum.

[58] Bianchi, G. (1998). "Throughput Evaluation of the IEEE 802.11 Distributed Coordination Function," *Proc. of 5th International Workshop on Mobile Multimedia Comm*.

[59] McKenney, P. E. and Bausbacher, P. E.(1991). "Physical- and Link-Layer Modeling of Packet Radio Network Performance," *IEEE J. on Selected Areas in Comm.*, **9:** 59-64.

[60] John, L. H. and David A. P. (2006). *Computer Architecture: A Quantitative Approach*, 4th Ed., New York, Morgan Kaufmann Publishers, Inc.

[61] Hennessy, J. L. and Patterson, D. A. (2003). *Computer Structures and Architecture*, 3rd Ed., Morgan Kaufmann Publishers, Inc.

[62]Charles W. A. (1993). *Systems Engineering Fundamentals:Defense Acquisition. Oxford*, University Press.

[63] Gomaa, H. and Menasc´e, D. A.(2001). Performance engineering of component-based distributed software systems. In *Performance Engineering – State of the Art and Current Trends*, volume 2047 of *Lecture Notes in Computer Science*, pages 40–55. Springer.

[64] Grassi, V. and Mirandola, R..(2004). Towards automatic compositional performance analysis of component-based systems. In Proceedings of WOSP 2004 :59–63.

[65] Varsha, A.(2005). Performance Evaluation of Computer Systems and Networks in Lecture notes for computer science, Department of Computer Science and Engineering, IIT BOMBAY.

[66]John C.S. L. (2009). Computer Systems Performance Evaluation Department of Computer Science & Engineering The Chinese University of Hong Kong Retrieved from 13/09/2009 from: www.cse.cuhk.edu.hk/_cslui

67] Richard, J. E., Kelley, P., and William, M. (2009). Performance Monitoring in Advanced Computer Architecture, Department of computer science and engineering, Michigan State University.

Appendix

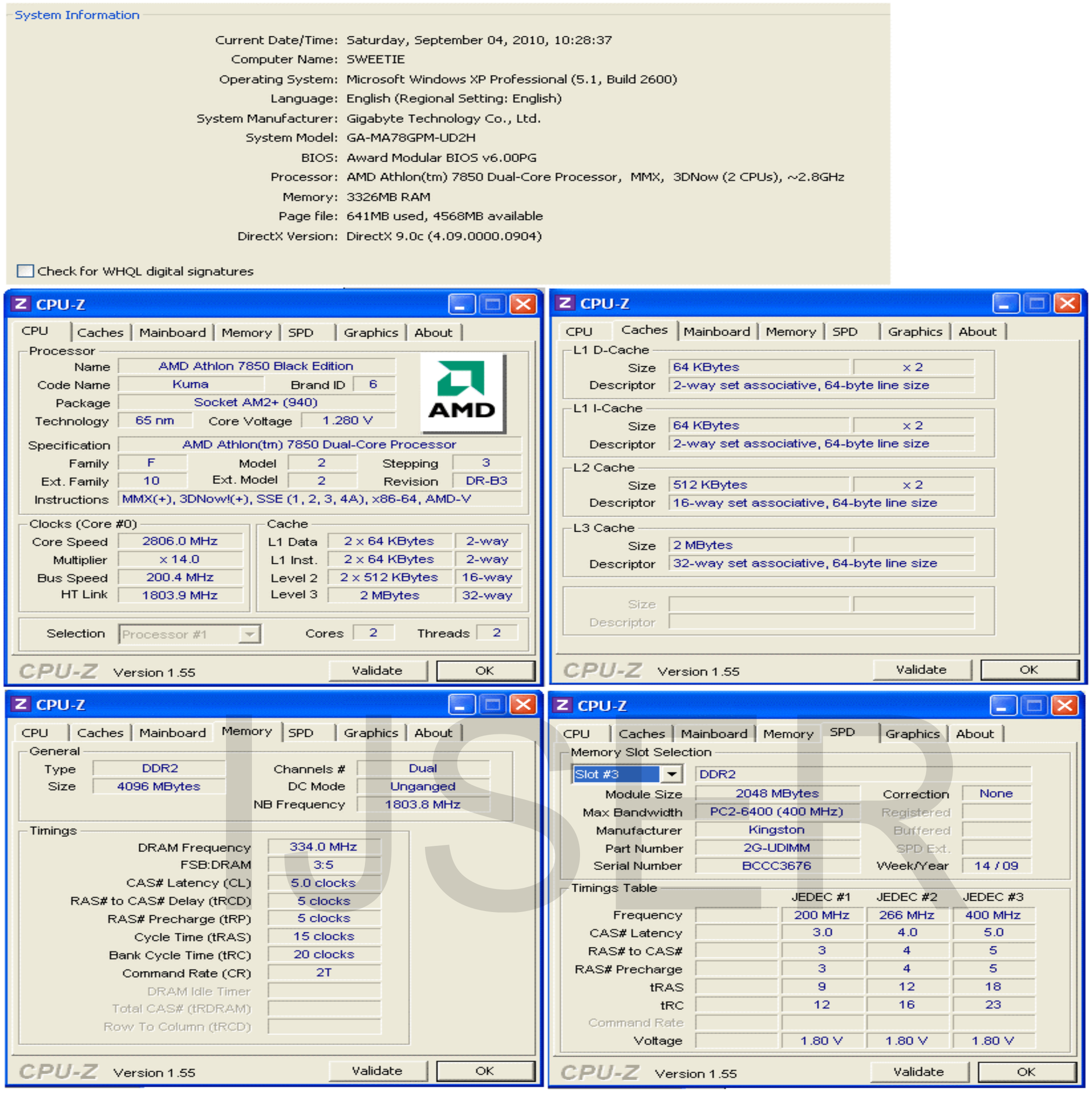

Figure 4: Old desktop specifications [57]

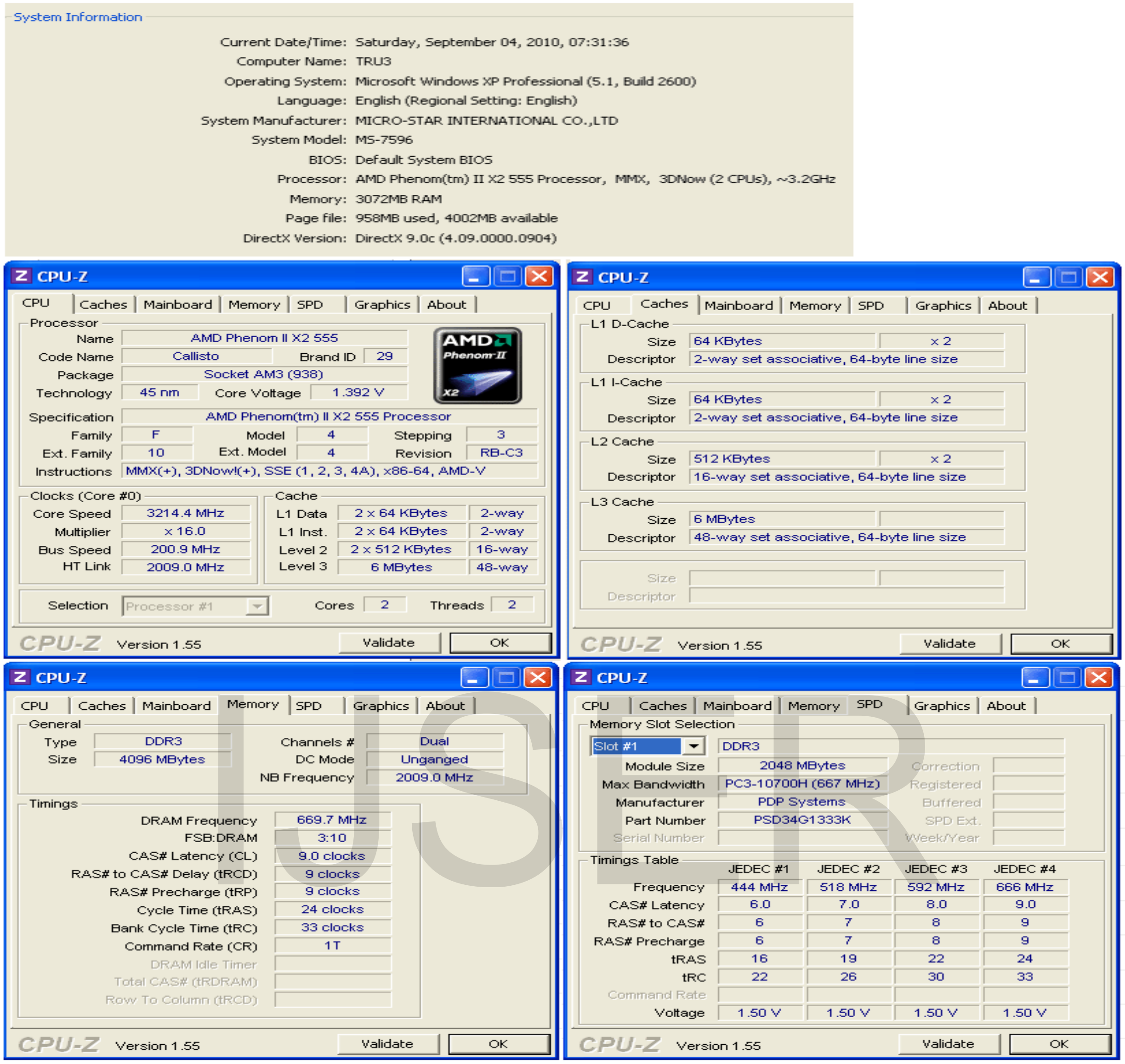

Figure5: The specification of a new system whose performance is worse than the old system [57]