Communications and Network, 2018, 10, 196-210
http://www.scirp.org/journal/cn
ISSN Online: 1947-3826
ISSN Print: 1949-2421

Analysis and Simulink Modeling of the Performance of Dynamic Web Server Using JSP and PHP

Fontaine Rafamantanantsoa*, Paulson Ravomampiandra

University of Fianarantsoa, Fianarantsoa, Madagascar
Email: *fontainerafamant@yahoo.fr

How to cite this paper: Rafamantanantsoa, F. and Ravomampiandra, P. (2018) Analysis and Simulink Modeling of the Performance of Dynamic Web Server Using JSP and PHP. *Communications and Network*, **10**, 196-210.
https://doi.org/10.4236/cn.2018.104016

Received: August 14, 2018
Accepted: November 9, 2018
Published: November 12, 2018

Copyright © 2018 by authors and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/
Open Access

Abstract

In recent years, web technology carried on growing and at same time, the number of internet users increased significantly in number. Today, a Web server is capable of processing millions of requests per day, but during the peak period may collapse and becomes critical causing unavailability of the services offered by the servers. That is why Web server performance is a topic of great interest to many researchers. In this paper, we evaluate experimentally the impact of JSP and PHP dynamic content technology: JSP and PHP with access to a database of performance data of Apache Web server. Using the "ApacheBench" performance measurement tool, the approach is to compare the performances of four different configurations of a Web server, such as: Apache Web server implementing JSP technology with access to PostgreSQL database, Apache using PHP technology with the PostgreSQL as database, Apache Web server using the JSP technology with access to MySQL database, finally Apache and PHP with DBMS MySQL. At the end of this article, we also present a Simulink model of Web server performance based on the simple M/M/1 queue. During the modeling, the MATLAB software was used.

Keywords

Analysis, Queue, JSP, Model, Performances Evaluation, PHP, Web Server, Simulink

1. Introduction

In the recent decades, the number of internet users did not cease to increase. According to some studies, statistics of users connected to the internet have risen more than 2.3 billion in 2012 in comparison with the number of inhabitants

of the world close to 7 billion; one third of the population is connected to the Internet. In November 2012, a Netcraft survey counted 625,329,303 active websites worldwide. Even though web servers are now able to handle millions of requests per day, during the peak period its performance can collapse and become critical. This rough location users thus despreciate the quality of service offered by the servers. Therefore, many researchers have conducted studies on the performance of the computer systems, namely: direct measurement, simulation and modeling. Direct measuring is an exact method but it requires a real system to collect specific information about the metrics of system performance. Simulation is the prototype of the real system and can provide performance measurement with more or less accurate results but can be difficult to build. As for the modeling, we used techniques such as: network queue, the Markov model, the Petri Nets and the neural network. Modeling is the cheapest method because now personal computer hardware is more unaffordable.

These three methods are also used by various researchers who have addressed their studies on Web server performance. In [1] and [2], the authors considered respectively the queue M/G/1/K*PS and M/M/1/K to evaluate the performance of the Web server. During their work, they studied the impact of parameters such as: traffic arrivals and the service discipline on performance metrics such as average Web response time, throughput and rate of server rejection. Other researchers such as R. Fontaine *et al.* [3] have modeled the performance of ApacheWeb servers using other methods, namely Neural Networks. For this, three models of neural networks were used to predict the performance metrics of the Web server, such as average response time, the percentage of rejection and flow depending on the optimization of system kernel parameters FreeBSD, Apache and inbound traffic, using the propagation algorithm of the gradient. In 2011, [4] has taken the same work but the operating systems they used was Debian and Ubuntu Linux 9.4. Numerous research focuses on the analysis and design of the Web server to identify the elements that make up the bottleneck. In [5], Rafamantanantsoa and Aussem have used Webstone to analyze performance of the Apache Web Server. They studied the different factors which can influence server performance such as: Apache optimization parameters (Maxclients, MaxUsers) and the FreeBSD operating system (e.g. Somaxconn). At the end of this paper, they proposed a simple model based on the queue that represents the behavior of the saturated Web server using the Mean Value Analysis (MVA) algorithm. Hu *et al.* also measured and analyzed the behavior of Apache. The authors used the benchmark tool SpecWeb96 and Webstone to evaluate the behavior of a Web server on the system uniprocessor and multiprocessor (4-CPU SMP (Symmetric Multiprocessor)).

More recently, Xiao and Dohi [6] evaluated quantitatively Apache performance using the Apache JMeter tool. Specifically, they focus on the relationship between the error rate of the Apache Web server and system parameters that can influence the performance of Apache.

The works we have presented above focus only on the use of static workload,

but the evolution of the internet has led other researchers to consider workload based on dynamic Web pages. Trent *et al.* [7], in their works, they focus on the comparison of dynamic languages PHP and JSP script using SPECWeb2005 on Lighttpd and Apache Web servers. Their results showed that there is only 5% - 10% difference in speed and performance between these two technologies.

In 2012, Aaqib and Sharma [8] measured the performance of Apache implementing PERL, PHP and Java Servlet technology and using a MySQL database. Their goal was to determine which of these three technologies gives better results on Windows and Linux platforms. Their results showed that the performance obtained with PHP is much better than when they used PERL and Java Servlets. A multiple linear regression model they developed was used to predict the performance of the Web server to validate their results. In this paper we will analyze and model the performance of the Apache Web server combined with JSP and PHP technology. This article is divided into four sections. Section 1 is for the presentation of the performance metric of the Web server. The experimental configurations will be given in Section 2. Then the third section will show the results of the various experiments. And it is in that last section model on server performance will be offered.

2. Web Server Performance

A web server is a system processing many types of client requests which progress simultaneously. These queries can be directed to static files of different sizes, dynamic web pages, CGI commands or access to databases through various APIs. The processing of these requests by the server is different with respect to others. So, the processing time of a query depends on the complexity of an application, the document size and the speed of the server. The latter also depends on the hardware (CPU, RAM…) and configuration software that make up the server.

To measure the performances of a web server, the units of measurement known performance metrics or performance indexes are used. These are key indicators that describe the quantitative behavior of a Web server. These indices may vary depending on the type, nature, the Web server configuration studied. Among the performance metric, the base units of measures for the performance of a Web server are:

Response time: this value shows the average execution time of an application, that is to say, the time after the client sends a request until it gets a response from the server;

Number of requests processed per second: is a measure of the number of requests managed by a Web server for a period of time;

Number of errors: applications rejected by the server due to a large number of requests received.

3. Performance Evaluation Methodology

As part of our study, we examine four different configurations of the Web server

for each experiment. Figure 1 show the various configurations during experimentation.

First, we send queries on dynamic Web pages with the extension **.jsp*or **.php*. Each HTTP request causes a SQL SELECT command to retrieve data from the database PostgreSQL or MySQL. Next, we varied the size of data to be retrieved from the database so that we can determine the impact of the size of data on server performance. For each configuration we have shown in Figure 1, we vary the size of data to get 20, 40, 60, 80 and 100 bytes. PostgreSQL and MySQL database each has a single table with four columns.

We notice that we only use the SQL SELECT command during our experiment. We did not use any other SQL commands such as INSERT, UPDATE and DELETE because most of the time, Web users consult only the various documents that the Web services provide them.

The characteristics of the dynamic workload that we used during the experiments are presented in Table 1.

4. Experimental Environment

During the experiment we used two machines, a machine that acts as a server and another client device that generates requests to the server. The server and the client machine are interconnected by a crossover cable RJ45. The rest of this section provides a description of details of this experimental environment. Section 4.1 shows the hardware configuration of our experimental environment and Section 4.2 presents the software we used.

4.1. Hardware Requirements

The client machine that we used is a Dell machine with a single Intel Pentium IV processor, 256 MB of RAM and 20 GB IDE hard drive. As for the server machine, an Acer brand with Intel Core i5 processor with 8 GB of RAM and a 1 TB SATA hard drive. Note that the server machine is not a dedicated Web server; it is a system where Apache is running in dual boot with Windows 7. Table 2 summarizes the configuration of the server and client machine used during the experiment.

4.2. Software Configuration

We used a Pear Linux OS8 distribution as the operating system for server and Debian 6 for the customer. We used the default kernel parameters of the Linux. Also the configuration of the operating systems was left at their default values. Table 3 gives a summary of the various software used lasting the experiments.

For all software used, setup optimizations are left at their defaults values.

4.3. Experiments Setup

Figure 2 shows the experimental setup used to obtain the data used to model Apache Web server performances.

Table 1. Workload during the experiments.

Specification	Workload	
	JSP	PHP
Number of files	16	16
Total size of files	5 Mo	5 Mo
Total data size in PgSQL and MySQL	600 Octets	

Table 2. Characteristics of materials used.

	Server	Client
Processor	Intel Core i5	Intel Pentium IV
Main memory	8 Go	256 Mo
Hard disk	50 Go	20 Go

Table 3. Characteristics of software used.

	Server	Client
Operating system	Pear Linux OS	Debian 6
Web server	Apache2.2	
RDBMS	-PostgreSQL -MySQL	
Tools	Top	ApacheBench

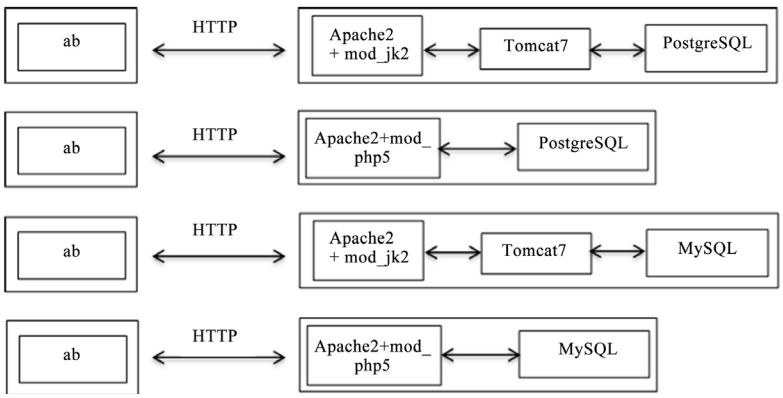

Figure 1. Performance measurement methodology.

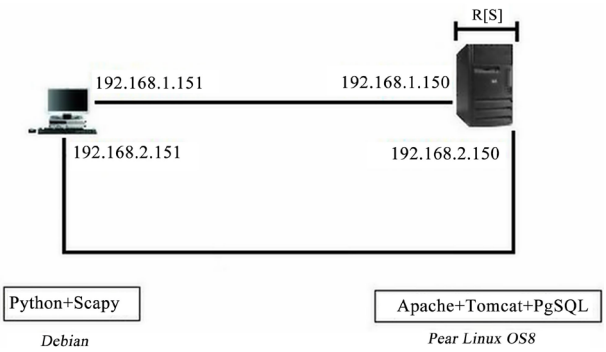

Figure 2. Experimental setup.

E[S] is the average service time of the Web server, that is to say the time between the arrival of a client request to the server and the time the server is ready to send the response to the client. E[S] can be calculated by taking the difference between the output of the response time of a query and the arrival time of the request on the server.

The course of the experiments is as follows: the requests are sent from the machine where the tool developed in Python and Scapy is installed. The request comes on the first network interface with the IP address 192.168.1.151. The Web server receives on its first network interface with 192.168.1.150 as IP address. Once the request has been processed, the response will be returned to the client via the second server network interface with the IP address 192.168.2.150 and the client machine receives this response on its second network interface with the IP address 192.168.2.151.

We used the same materials as in the previous sections. A series of experiments were conducted to collect the performance metrics of Apache Web server with supported JSP technology and with database access. In this section, we have not used the ApacheBench tool, but we developed a performance measurement tool of the Web server using the Python programming language and network packet manipulation software called Scapy.

Figure 3 shows the user interface of the tool that has developed. This GUI was conducted from Qt4 Designer.

Tool description

- *Server IP*: is used to indicate the Web server's IP address to be tested.
- *Competitors*: used to indicate the number of users simultaneously accessing the resources provided by the server.
- *Requests*: it allows specifying the total number of query to run during each performance test.
- *URL*: allows you to indicate the absolute path of the web page to retrieve customers during the test.

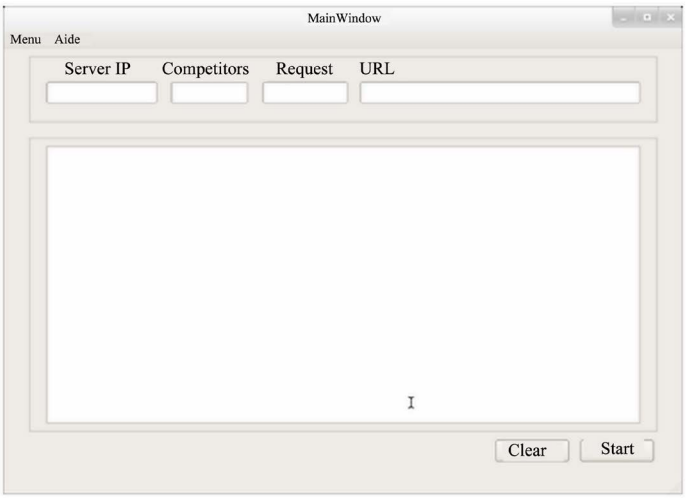

Figure 3. GUI tool.

The Start button is used to start the Web server performance test. And as for the Clear button, it is used to clear the display area of the results of experiments.

We used this tool to run performance tests. The size of the document was varied from 20 to 100 bytes in steps of 20. And for each value, the tool retrieves metrics on Web server performance. As a result of the test, is particularly useful in measuring the time of E[S] of the server.

5. Experimental Results

A series of tests were performed to measure and then examine the performance of the Web server. The various test consisted of ApacheBench running on client machines with the workload shown in Table 1 after each test, the ApacheBench tool collects statistics on various performance metrics such as average response time, the number of errors.

The "top" tool for Linux allows monitoring system resources on the server machine. The monitoring resources are: memory and the CPU.

For each experiment, we will establish the relationship between the data size and the average response time. To properly organize this section we will represent the results of the various tests as follows:

***Experiment* 1:** Access to the first column of the databases table.

***Experiment* 2:** Access to the second column of the database table.

***Experiment* 3:** Access to the third column of the database table.

***Experiment* 4:** Access to the fourth column of the database table.

5.1. Experiment 1: Access to the First of the Database Table

The average response time is an important metric of the performance of a Web server. Figure 4 shows the relationship between the size of the recovered data in the first column and the database medium response time.

It is found that the average response time of the server configuration with *Apache* and *PHP* with the access to MySQL database is reasonably better compared to other configurations that we used during the experiments. In addition Figure 4 shows that when the Apache Web server is coupled with technology generation dynamic content JSP, server is better when using the MySQL DBMS that when using the PostgreSQL DBMS.

Based on these results, we can conclude that the MySQL DBMS is suitable for a small base while the PostgreSQL DBMS is suitable for a much larger base.

For all configurations tested, the average response time increases significantly between data size 0 and 20 bytes and it becomes stable between 20 and 80 bytes. This stability can be explained by the fact that use of the system is stabilized between 20 and 80 bytes.

During the experiment the top Linux tool also shows that JSP technology uses more memory than PHP resource. This is due to the use of the Java Virtual Machine (JVM).

The equations of the curves of the average response time according to the size

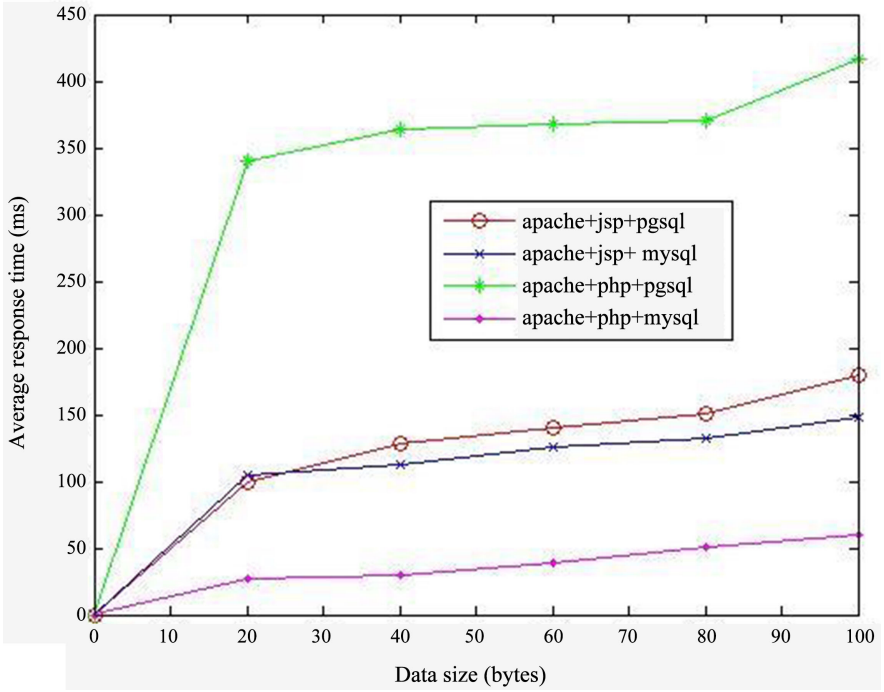

Figure 4. Average time response depending on the size of the retrieved data in the 1st column.

of data for Apache and JSP with PostgreSQL, Apache and JSP with the access to MySQL database, Apache and PHP with the DBMS PostgreSQL, Apache and PHP with MySQL obtained from MATLAB are respectively as follows:

$$y = 0.00056x^3 - 0.1x^2 + 6.5x + 1.4 \quad (1)$$

$$y = 0.00057x^3 - 0.11x^2 + 6.3x + 3.5 \quad (2)$$

$$y = 0.00011x^3 - 0.019x^2 + 1.4x + 1.4 \quad (3)$$

$$y = 0.0021x^3 - 0.39x^2 + 22x + 10 \quad (4)$$

5.2. Experiment 2: Access to the Second Column of the Database Table

Figure 5 shows the results of experiments in the second column of the table of databases PostgreSQL and MySQL. The recovered data size varies in databases 20, 40, 60, 80 and 100 bytes.

As in the first experiment the configuration *Apache* Web serverimplementing *PHP* with the access to MySQL database gives us better performance than other configurations. In addition, there is no major difference between the response times on access to the first column of the table in the database. One can see this small difference on the equations of the curves obtained from the second experiment (the order of the equations is the same as in the first experiment):

$$y = 0.00052x^3 - 0.1x^2 + 6.8x + 2.9 \quad (5)$$

$$y = 0.00052x^3 - 0.11x^2 + 6.4x + 3.2 \quad (6)$$

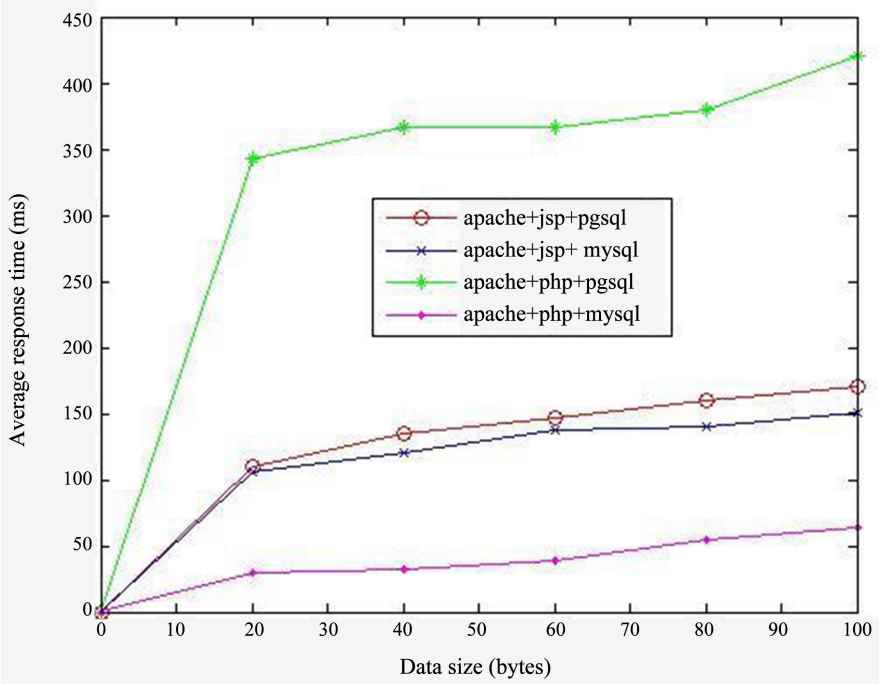

Figure 5. Average time response depending on the size of the retrieved data in the 2nd column.

$$y = 0.0021x^3 - 0.39x^2 + 22x + 11 \quad (7)$$

$$y = 0.00014x^3 - 0.022x^2 + 1.5x + 1.8 \quad (8)$$

5.3. Experiment 3: Access to the Third Column to the Database Table

In this third experiment, we access the third column in the database table. Configurations are always the same tests, first Apache Web serverimplementing JSP with the access to PostgreSQL database, Apache with JSP and MySQL, finally Apache implement the technology PHP with the DBMS PostgreSQL and Apache implement PHP with the access to MySQL database (**Figure 6**).

Between 0 and 20 bytes, there is a significant increase in the average response time for each configuration tested. But the average response time stabilizes between 40 and 80 bytes. The small difference in the average response time compared to previous experiments can be seen in the equations of the curves:

$$y = 0.00053x^3 - 0.11x^2 + 7x + 3.6 \quad (9)$$

$$y = 0.00052x^3 - 0.1x^2 + 6.2x + 3.8 \quad (10)$$

$$y = 0.0021x^3 - 0.39x^2 + 22x + 11 \quad (11)$$

$$y = 0.00011x^3 - 0.019x^2 + 1.4x + 1.8 \quad (12)$$

5.4. Experiment 4: Access to the Fourth Column to the Database Table

The fourth and final test is accessing the fourth column of the database table

(Figure 7).

Comparing the curves of equations in the above three experiments, there are no considerable differences in mean response time. But in configuration Apache implemented PHP with the access to MySQL database gives better performance compared to the other configurations.

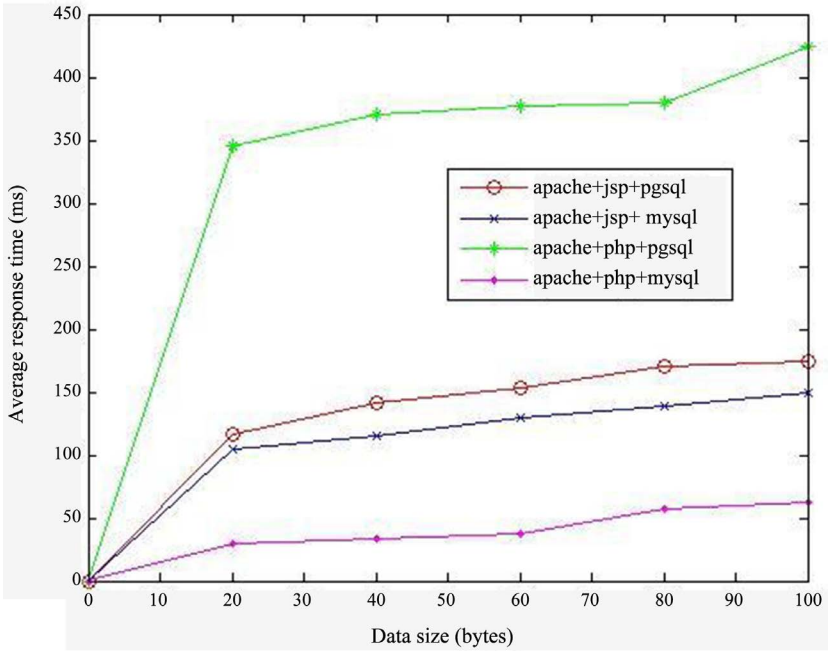

Figure 6. Average time response depending on the size of the retrieved data in the 3th column.

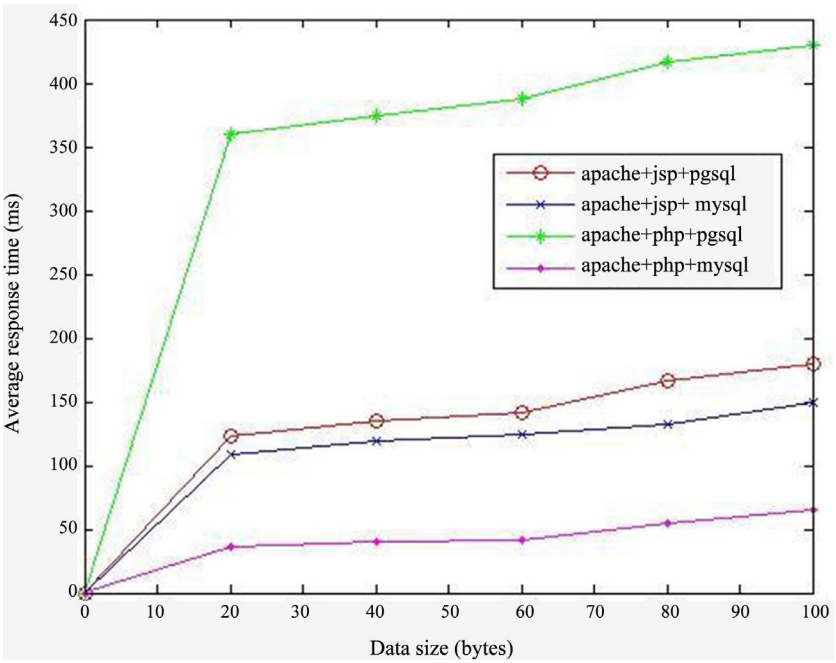

Figure 7. Average time response depending on the size of the retrieved data in the 4th column.

$$y = 0.00065x^3 - 0.12x^2 + 7.3x + 5 \quad (13)$$

$$y = 0.00064x^3 - 0.12x^2 + 6.8x + 3.3 \quad (14)$$

$$y = 0.002x^3 - 0.37x^2 + 22x + 14 \quad (15)$$

$$y = 0.00021x^3 - 0.036x^2 + 2.1x + 1.5 \quad (16)$$

6. Modelling Web Server Performance

We present in this section a simple model based on the M/M/1 queue representing the performance of the Apache Web server. During the modeling, we use data obtained during the various tests.

6.1. The M/M/1 Queue

This is the simplest and most widely used for modeling of computer systems design. An M/M/1 is formed by infinite capacity of queue and a unique server. The clients arrived in the system according to a Poisson process with the rate λ and the inter-arrival $A(t)$ is an exponential distribution. The service time is distributed to an exponential law with μ parameter. The order of service is FIFO.

The solution of M/M/1 queue is:

Server utilization: There is a higher probability that the server is occupied. We define the rate of use of the server as the intensity of the traffic (or load) noted ρ.

$$\rho = \frac{\lambda}{\mu}$$

Average response time: represents the average time of stay of a client in the system. This time consists of the waiting time and the service time.

$$w_s = \frac{1}{\mu - \lambda}$$

6.2. Simulink Model

We modeled the performance of the Web server using the standby M/M/1 queue. Inbound traffic is a process of rate λ fish. The service time follows the exponential distribution with rate μ.

In **Figure 8**, the block *Event-Based Random Number* 1 generates in a random manner based on a number events. It specifies the value of λ and the arrival of law traffic arrival. Λ following the law is the number of requests sent per second and during the simulation, we varied the value of λ from 20 to 100. This block is connected to the block *Time-Based Entity Generator* which acts as a generator of entities which satisfy the criteria we specified. The block Time-Based Entity Generator is connected to the IN port of the *Schedule Timeout* block that determines the timing of events which happen for each entity.

The OUT port of the Schedule Timeout block liaises with the IN port of the FIFO Queue block representing the queue with infinity length. FIFO means that

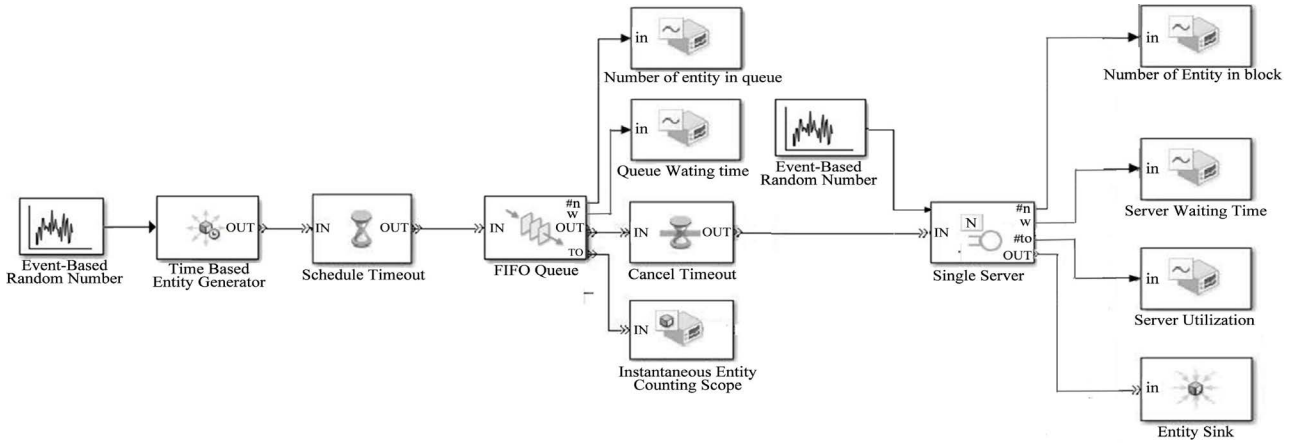

Figure 8. Simulink model.

the first entity to come in this block is the first to come out. How long an entity remains in this block can't be determined in advance depending on the number of entities in this block. The *#n* ports, *#w* and *to* are connected to respective blocks queue length which shows the curve of the entities in queue instantaneous and counting entity scope curve gives the number of entities in expectations.

The OUT port of the FIFO queue block is connected to the IN block Cancel Timeout cancels a timer event that the Schedule Timeout block has already provided for the entity arrival in the FIFO queue block. The cancel Timeout block is connected to the single server block representing our Web server. The w use of single server block port is connected to a respective block Waiting time Server that provides visualization of the curve of Waiting time in the server and using the curve that shows the utilization of the server.

Using t block single liaises with the block Event-Based Random Number that specifies the distribution of the service time of the server. In this study, the service time follows the exponential distribution. Here, the value of the service time is the average value of E[S] we obtained during the experiments. The last block, *i.e.* the block Entity Sink provides us a way to end the way entities.

6.3. The Mean Square Error

The squared error often called Mean Square Error (MSE) is a measure of the average error, weighted by the square of the error. It answers the question, "What is the magnitude of the error of prediction", but does not indicate the direction of errors. Because it is a quantity squared, the MSE is influenced more by large errors than smaller errors. Its range is 0 to infinity, a score of 0 being a perfect score.

The MSE calculation using the mathematical equation:

$$MSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(F_i - O_i\right)^2}$$

where:

F_i values prediction parameter,

O_i is the value corresponding verification (Observed or analyzed),

N is the number of check points (grid points or observation points) in the area of verification.

In our work, the squared error is used to calculate the difference between the curves obtained from the simulation and the analytical model obtained by M/M/1 queue.

6.4. Results and Discussion

In this section, we will present various simulation results and the analytical model obtained. The vector entry consists of trafficking arrived λ.

6.4.1. Server Utilization Rate

The server utilization is a metric that reflects the extent of the server's occupation. It varies between 0 and 1 according to the number of entities that use resources offered by the server.

Figure 9 shows the utilization of the Web server during the simulation measured and calculated from the model queue M/M/1. It is observed that logically, the curve server utilization increases with the number of applications processed. The curve increases to the maximum value that is equal to 1 when the server reaches the point of congestion, there is a release of packages and should be constantly close to the maximum value. Congestion can be explained by the increase in the length of the queue.

From the curve of the server utilization, the results of the M/M/1 are reasonably good. This is confirmed by the value of the squared error calculated between the two curves. The value of the mean squared error is equal to 0.033.

6.4.2. Average Response Time

The average response time shows the average time a client stays in the system. It's a very important metric of the performance of a Web server. Figure 10 shows the curves (gotten by the analytic method and the simulation method) of the response time according to the number of requests.

Considering Figure 10, we note that logically, the two curves of the response time increase gradually with the size of the data. It can be explained by the increase of the length of the queue when the size of data increases. When the size of data becomes big, more time is necessary for the server to handle it. The two curves of the analytic model and the simulation are nearly identical; we can affirm it by the value of the Mean Square Error (MSE) between the two curves that is very small. The value of this MSE is equal to: 0.004.

7. Conclusions

In this paper we conducted an analysis of Web server performance. Several experiments were conducted to examine the performance of the Web server. During the experiments, we used the performance evaluation tool ApacheBench. During

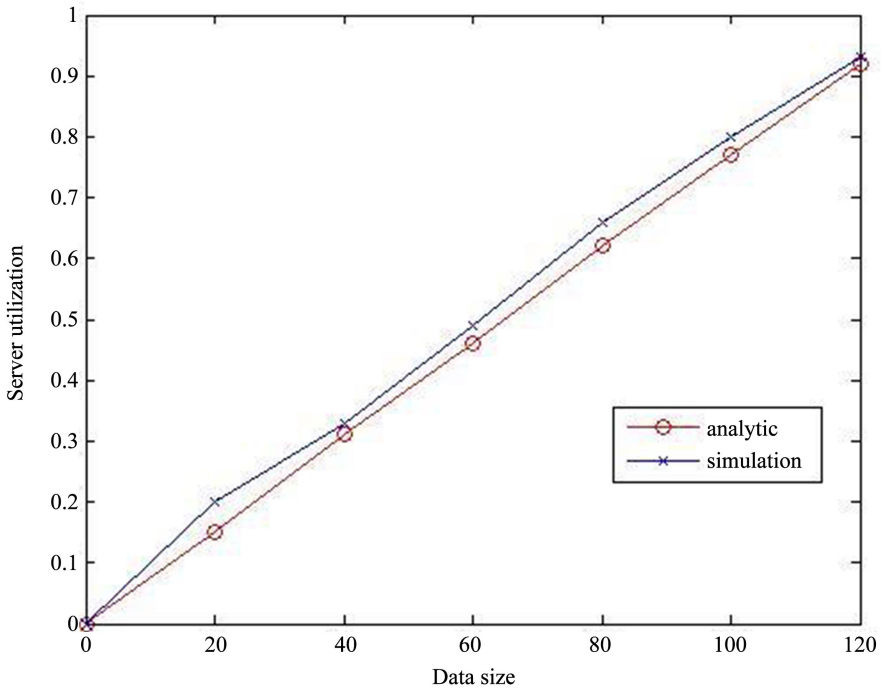

Figure 9. Server utilization.

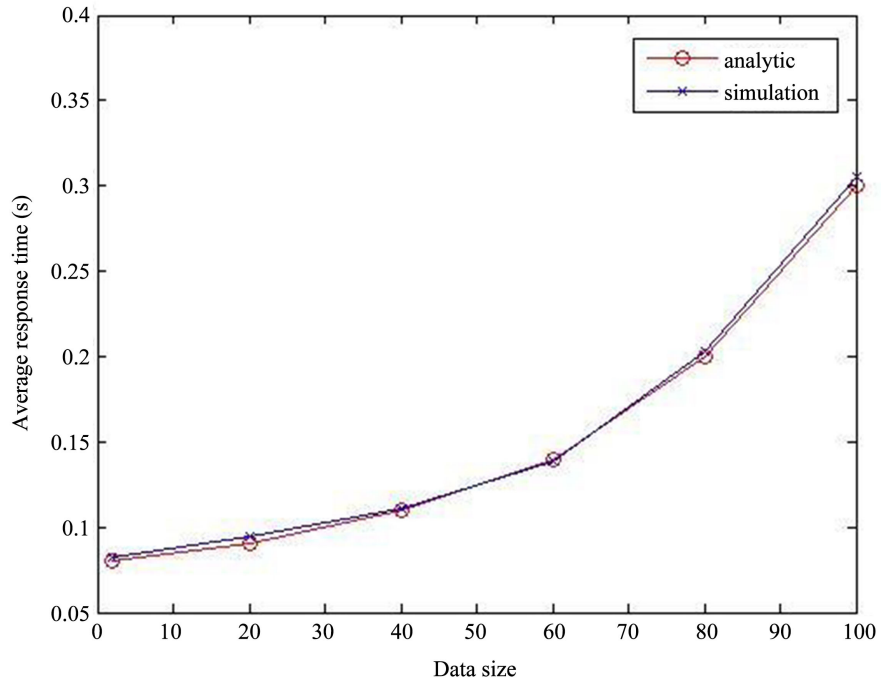

Figure 10. Web server average response time.

these experiments, we examined different server configurations: Apache and JSP with the DBMS PostgreSQL, Apache Web server implement PHP with the DBMS PostgreSQL, Apache and the technology JSP with the access to MySQL database, Apache and PHP with MySQL. For these configurations, the relationship between the average response time and the data size retrieved from data-

bases is considerable. The experimental results showed that: compared to other configuration that we used, the configuration Apache Web server implemented PHP with the access to MySQL database gives us the best performance. As for system resources, configuration using JSP technology consumes lots of memory resources due to the use of the Java Virtual Machine.

At the end of this paper, we presented a simple model of the performance metrics of Web servers based on the M/M/1 queue. For this, we used data obtained during the experiments. Two methods of resolutions queue were used: the analytical method based on the theory of queues and simulation. During the simulation, we used the simulation tool Simulink MATLAB. During the modeling phase, we focused on performance metrics such as: utilization of the server, the average waiting time in the server, the average waiting time in the queue, and the number of entities in the queue. The comparative results of the analytical model and simulation are reasonably good.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Cao, M. Anderson, Nyberg, C. and M.K. (2003) Web Server Performance Modelling Using an M/G/1/K*PS Queue. *ICT*, 10*th International Conference on Telecommunications*, Vol. 2, Papeete, 23 February-1 March 2003, 1501-1506.

[2] Acosta, R. and Waena, F. (2009) Queuing Theory: Performance Modeling of Computer Networks. Strathmore University, Nairobi.

[3] Fontaine, R., Laurencot, P. and Aussem, A. (2007) Performance Learning, Real Time Monitoring and Admission Control of a Web Server Using Neural Technique. *Proceedings of* 4*th International IEEE of Science of Electronics*, *Technologies of Information and Telecommunication*, Aubière, 2.

[4] Aid, L., Loudin, M. and Hidouci, W.-K. (2011) An Admission Control Mechanism for Web Servers Using Neural Network. *International Journal of Computer Applications*, 15.

[5] Fontaine, R., Laurencot, P. and Aussem, A. (2009) Analyse des performances et modélisation d'un serveur Web. *Proceedings of* 5*th International IEEE of Science of Electronics. Techonologies of Information and Telecommunication*, Aubière, March 2009, 7.

[6] Xiao, X. and Dohi, T. (2010) Estimating the Error Rate in an Apache Web Server System. *International Journal of Software Engineering and Its Applications*, **4**, 19-28.

[7] Trent, S., Tatsubori, M., Suzumura, T., Tozawa, A. and Onodera, T. (2009) Performance Modeling of Computer Networks. Strathmore University, Nairobi.

[8] Aaqib, S.M. and Sharma, L. (2012) Analysis of Compute vs Retrieve Intensive Web Applications and Its Impact on the Performance of a Web Server. *IJANA International Journal of Advanced Networking and Applications*, 236 p.