International Journal of Computer Trends and Technology Volume 68 Issue 9, 45-52, September 2020
ISSN: 2231-2803 / https://doi.org/10.14445/22312803/IJCTT-V68I9P107 © 2020 Seventh Sense Research Group®

Original Article

Cloud Computing Model for Enhanced Resource Usage in Multi-Tenant Application Environment

Michael Okumu Ujunju[1], Solomon Ogara[2] Kelvin Omieno[3]

[1]*School of Computing and Informatics, Masinde Muliro University of Science and Technology*
[2]*School of Informatics and Innovative Systems, Jaramogi Oginga Odinga University of Science and Technology*
[3]*School of Computing and Information Technology, Kaimosi Friends University College*

Received Date: 16 August 2020
Revised Date: 22 September 2020
Accepted Date: 25 September 2020

Abstract - *Cloud computing technology is built on the concept of virtualization to facilitate resource sharing among many cloud tenants. Due to this attractive feature of the cloud, many organizations and institutions are adopting this technology. Cloud users entrust their resources, hardware, and software to cloud service providers to facilitate sharing, which is the main objective of cloud technology. However, fair allocation of these resources is considered a problem as users don't realize optimal use of these resources, ensuring equitable sharing and allocation of resources. The purpose of this paper is to simulate results built and test a model for resources used in cloud computing multi-tenant application environments. Experimental and design science research designs were used. Data was extracted via simulated results and outputs. The complementary data was collected through focus group discussion, think-aloud protocol, and questioning protocol, which was used in the model validation process. The developed model is expected to help in improving resource usage control for cloud users to enable cloud service providers to enhance the quality of service delivery to their many customers.*

Keywords - *Cloud Computing, Resource Usage, Multi-Tenant, Model*

I. INTRODUCTION

Cloud computing technology is built on the concept of virtualization to facilitate resource sharing among many cloud tenants. Due to this attractive feature of the cloud, many organizations and institutions are adopting this technology. Cloud users entrust their resources, both hardware and software, to cloud service providers to facilitate sharing, which is the main objective of cloud technology. A resource demand specifies the processing time for a single request at a dedicated resource. Given that the resource demands may dynamically change during system runtime, a re-run of the estimation in regular intervals is needed to get updated values. Methods based on general optimization, such as [1] [2], cannot be considered due to their computational complexity, which limits their applicability during system runtime. Development requirements comprise programs development tools that are encapsulated and offered as a service, upon which other higher levels of the service can be built. It is noted that a CSP provides its clients with a development environment that aids in the creation of software that operates on the cloud provider's hosted infrastructure. Similarly, [3] argues that the client has the choice to build his/her programs, which run on the provider's infrastructure. Both Crowe et al. and Harris point out that the customer becomes the key reference in terms of provisions of cloud services, enabling him/her the flexibility of scaling the resources according to his/her requirements. PaaS usually describes an additional level of services layered on top of IaaS. It sometimes confuses some clients, hence the need to properly coordinate the two services to enhance security. According to NIST [4], PaaS is whereby a customer can deploy, manage and run applications using a programming language and execution environment, while IaaS is whereby a customer can provision and use processing, storage, or networking resources. Therefore, PaaS is, in general, a higher level of abstraction than infrastructure. Some well-known PaaS is Google's App Engine, Force.com, LAMP platform (Linux, Apache, MySQL, and PHP), Ruby, among others. Unlike SaaS and IaaS, PaaS rely upon a protected and dependable network as well as a safe web browser. The PaaS application safety measures constitute of security of the PaaS platform itself (that is, runtime engine) and the security of client applications hosted on it [5]. Therefore, the question is this layering of 15 resources within the cloud from SaaS, PaaS, and IaaS creates confusion on who is responsible for the correct integration of these assets so that it does not act as a weak link? Thus, like SaaS, PaaS also brings data security issues due to its nature of sharing resources, which raises privacy and confidentiality concerns.

IaaS Model: In this model, a service provider avail to customers the use of processing power, storage, and networking capabilities over a network. Cloud providers offer virtual data centres of resources, for instance, networking, computing resources, and storage resources. Similarly, the provider configures and manages their own operating systems and software and, to a great extent, scales their programs and provides all the services required to run them. As [3] notes, servers, storage systems, networking equipment, data Centre space, among others, are pooled and readily accessible to handle workloads. Examples of this are Amazon EC2, GoGrid, IBM cloud, and 3 Tera, among others. Thus, both Crowe et al. and Harris argues that CSP provides virtual data centres and is responsible for configuring and maintaining the operating system. In this pooling of resources, the question remains, who provides/guarantees security? PaaS is ideal for application servers and databases because it will require integration with some web services as well as the use of common databases. This layer is also good because, at some point, the applications will require the processing of real-time data. Similarly, the API gateway has been employed to make it easy to be deployed across a wide range of platforms. Consequently, in addition to resolving security and privacy threats in the cloud, the application can also be extended to offer 33 authentication using biometric features, especially for clients who may require enhanced identity management mechanisms.

II. METHODOLOGY

This study adopted a mixed research design that involved experimental design and design science research. The design followed six steps: problem identification and motivation; definition of the objectives for a solution, design, and development; demonstration; evaluation; and communication [6]. On the other hand, the experimental design method was used during the data simulation process, where a controlled experimental factor was subjected to special treatment for comparison. The experimental design consisted of the following steps: Identifying the experimental unit, identifying the types of variables, defining the treatment structure, and finally defining the design structure. The sample units involved (50) simulated cloud service providers (virtual machines) and (1500) users or tenants utilizing cloud services.

III. TOOLS

The study used the following during data collection from the identified sample of the study Software: Simulation software is a program used for the user to observe an operation through the simulation without actually operating [7]. Normally, it is based on a modelling process according to mathematical formulas or an algorithm. This enables the algorithm simulators to independently for testing and simulates various algorithms. By using these algorithm simulators, the study was able to visualize the data structure and computation process, and as a result, the effect and benchmark of different algorithms were seen. For obtaining the desired results, Parameters Monitoring Web Server, Vision Simulation Software, and Rapidly Random Exploring Tree Simulator simulation software.

The simulation process was implemented using java programming enabled CloudSim simulator. The server runs on a window operating system Microsoft® Windows Server® 2016 and Microsoft Windows Server with Hyper-V® VMware® vSphere® ESXi®. For programming, java compilers (JDK) and NetBeans IDE was installed to test the algorithm.

a) Hardware

The simulation was done using Dell EMC PowerEdge R230, which is an excellent first server or replacement server for driving applications in SMB. The R230 delivers greater memory capacity, more hard drives, and I/O slots and accelerates data throughput and IOPs performance. The R230 also supports the full-featured, remote management of the integrated Dell Remote Access Controller (iDRAC8) with Lifecycle Controller, making it highly attractive for ROBOs of large institutions. The server had 1 processor from the following product families:• Intel® Xeon® processor E3-1200 v6 product family, Intel Pentium®, Intel CoreTM i3, and Intel Celeron®. The server Architecture: Up to 2400MT/s DDR4 DIMMs, Memory type: UDIMMs.

b) Memory module sockets

4 Maximum RAM of 64GB. Storage modules of 2.5" SATA SSDs 3.5" Enterprise SATA 7.2k HDDs and 3.5" nearline SAS 7.2k HDDs. Besides, the servers have Systems management, Remote management, Dell OpenManage Connections, and Dell OpenManage Integrations: Dell OpenManage Integration Suite for Microsoft System Center and Dell OpenManage Integration for VMware vCenter®. The server also had device access for 5 total USB: Rear: 2 x USB 3.0 ports, Front: 2 x USB 2.0 ports and Internal: 1 x USB 3.0 port. These hardware components made it easier for the simulation process.

IV. FINDINGS AND DISCUSSION

To achieve the objective of this study, the researchers classified Virtual machines in different classes of 1 VM, 5VMs, 10 VMs, 20 VMs, 30 VMs, 40 VMs, and 50 VMs. Since data was simulated, the response rate was 100%. This included 1500 tenants and 7 classes of VMs. VMs was used to represent the server and the number of tasks subjected to the server as the number of connected tenants. Table 1 represents running time on different VMs and the number of tasks or clients' requests.

Table 1. Simulated Data

	1 VM	**5 VM**	**10 VM**	**20 VM**	**30 VM**	**40 VM**	**50 VM**
Tasks	**Time**	**Time**	**Time**	**Time**	**Time**	**Time**	**Time**
1	160	160	160	160	160	160	160
100	16000	3200	1599.98	800	479.99	320	320
200	32000	6400	3199.94	1600	959.98	800	800
300	47999.2	9599.84	4799.9	2399.96	1600	1280	1280
400	64000	12800	6399.94	3200	2079.95	1759.96	1759.96
500	80000	16000	7999.9	4000	2560	2240	2240
600	95998.4	19199.68	9599.78	4799.92	3200	2719.94	2719.94
700	111997.6	22399.52	11199.74	5599.88	3679.91	3039.93	3039.93
800	128000	25599.68	12799.78	6400	4159.9	3519.92	3519.92
900	143996.8	28799.36	14399.66	7199.84	4799.92	4000	4000
1000	160000	32000	15999.7	8000	5279.87	4319.9	4319.9
1100	175996	35199.2	17599.58	8799.8	5759.87	4799.92	4799.92
1200	192000	38399.36	19199.62	9599.84	6400	5279.87	5279.87
1300	207996	41599.04	20799.5	10399.76	6878.83	5759.87	5759.87
1400	223996	44799.04	22399.46	11199.76	7359.82	6079.86	6079.86
1500	240000	47999.2	23999.5	11999.8	8000	6559.84	6559.84

Table 1 depicts the response time (t), the number of requests from tenants (tasks), and Virtual Machines (VM), which provide the host of the resources. Virtualizing the memory space into multiple storage spaces enabled the study to rename the servers or Virtual Machine (VM) as 1 VM, 5 VMs, 10 VMs, 20 VMs, 30 VMs, 40 VMs, and 50 VMs. The number of tasks was equally distributed in these virtual machines, and their corresponding time was recorded. When simulation results were generated on 50 VM, it was realized that the time taken for each set of inputs had no significant difference from that of 40 VM. The outputs were identical. If the number of tasks is denoted as K, then the time taken by K process/tasks or tenants to access a resource from 40 VM and any other more virtual machines would be the same. Furthermore, the experiment demonstrated that the time to access a resource for one running task or tenant was 160 ns. Nanoseconds were used as a time of measure during the simulation process. A Nanosecond can be defined as a thousandth of a millionth of a second or a billionth of a second (1.0 * 10-9). Increasing the number of tenants on VMs against time demonstrated that the process does not depend on the memory space or the amount of RAM. For the simulation process, the memory size was held at 512 MB.

Since the access to the resource involve a tenant sending a request, a server (VM) receiving the request, acknowledging the receiving, processes the request and sending a reply message to the tenant. All these processes were viewed in turns of time taken. Table 1 indicates that a single tenant request, regardless of the number of VMs, will be serviced by a single VM and hence the time taken for the process is 160 ns, which is constant. In the case of 1 VM, subjecting 100 tasks or tenants will result in the sharing of the resource in that VM, resulting in a time of 160*100(16000)ns; therefore, increasing tasks or tenants to say 500 will result in 80,000 ns. This means that processing time is N*160, where N denotes the number of tenants (tasks) subjected to a server (VM). This implies that time taken is directly proportional to N. increasing N increases the amount of time taken.

The study also collected and analyzed data to test the effect of time taken to process the client/ tenant tasks when the VM is increased to more than one. To analyze the effects, the simulation was carried out involving 5 VMs, 10 VMs, 20 VMs, 30 VMs, 40 VMs, and 50 VMs. In this case, the memory location simulated was demarcated into 5, 10, 20, 30, 40, and 50 slots, respectively, to represent the number of server machines. These simulated memory slots were subjected to 1-1500 task or tenants requests, and their corresponding time taken for each was as in Figure 1.

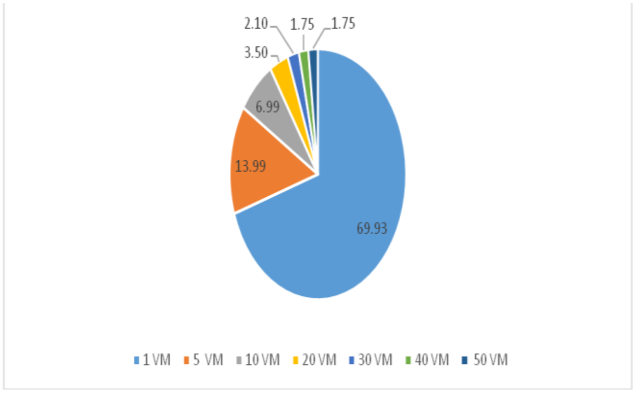


Fig. 1 Proportionality of time taken across VMs

In this case, Figure 1 indicates that 1 VM takes the worst-case time in accessing the resource, which is at 69.93%. When the VMs are increased to say 5 VMs, the time of access is reduced to approximately 14%, increasing VMs to 10, the access time percentage is reduced to approximately 7%, 3.5% for 20 VMs, 2.1% for 30 VMs, and 1.75% for any number of VMs 40 or above, meaning with 40 VMs the time of access to a resource is constant at a specific number of tasks or tenants. This comparison excluded the comparison for a single tenant or one task since the time is taken for accessing a resource is constant across the number of servers or Virtual machines.

A. Implication on Analysis (Number of Tasks vs Access Time)

The statistical analysis enabled the study to have a basis that informed the core elements of the analysis, which included: An element of input (number of tasks), the number of VMs that are at the processing side and, time is taken for processing as the core elements of the algorithm. In the event of multiple tenants accessing multiple servers (VMs), there is no guarantee that all the tenants send for similar requests to the server, and since a single server can use up to 160ns to serve a client request on a certain shared resource, it means that if two clients using the same server send each different request (resources request), then the processing can be done concurrently since tenants specific data needs are isolated from each other.

The deviation in the time taken as the number of VMs increases give a justification that some tasks or request clients make may be heterogeneous (different or unique) tasks to heterogeneous VMs or servers (different servers). Therefore the representations in Figures 1 only considered a case where the said resource is shared amongst the tenants; hence, the time indicated defines the time quantum or slice of the sharing algorithm where each tenant is served at a constant time quantum of 160ns.

When the VMs are more than one or multiple servers, then the resources are shared or distributed in the said VMs. Therefore any VM allows the resource to be distributed in any way. I say (k) tasks (tenants) send their requests to (m) VMs such that (k) is less than (m), then the time taken is constant 160ns. This holds when m>40 VMs. When the resource request is shared or unique tasks from the tenants, that is, a task having a unique request, the time quantum is reached in some way since some VMs will be servicing requests in a condition similar to that of the pigeonhole principle. Meaning that there is a possibility of having more than one client request being handled by one server. This also gives a constructive implication when it comes to the overall time taken for (n) task requests. Therefore, this argument holds for 5Vms, 10VMs, 20 VMs, 30 VMs, and 40 VMs, as depicted in the simulated results

For optimum resource allocation, the simulation algorithm is expected to classify and determine the best way the tasks are handled or responded to within the shortest time possible and also make sure that each VMs is engaged at any time in case the requests outnumbers the VMs. So that 1 VM may not be overwhelmed when others having similar resources are idle. The classification takes into consideration: Number of requests or tasks (N), the number of VMs denoted as (M), shared resources request denoted as (L), and unique resource request denoted as (K). The algorithm role is to balance resource allocation (M) to the domain of the request of (N), which may comprise K and L. the optimization is achieved when each tenant is satisfied, and no tenant compromises into other's quota, all VMs being remitted and request serviced in the shortest time possible (t). Therefore the relationship between t, N, M, K, and L are the basic representation and manipulation of the algorithm as depicted on the growth curves.

B. Model Constructs

The parameters for the model were extracted from the findings above. These were the variables that represented the practices that were related to the phenomena underpinning the study. These include the response time (t), the number of requests from tenants (tasks), and Virtual Machines (VM), which provide the hosting services for the resources. Virtualizing the memory space into multiple storage spaces enables the study to rename them as Virtual Machine (VM), 1 VM, 5 VM, 10 VM, 20 VM, 30 VM, 40 VM, and 50 VM. The number of tasks was equally distributed in these virtual machines, and their corresponding time was recorded.
Fig. 2 Summarizes the Nearest Neighbor Analysis

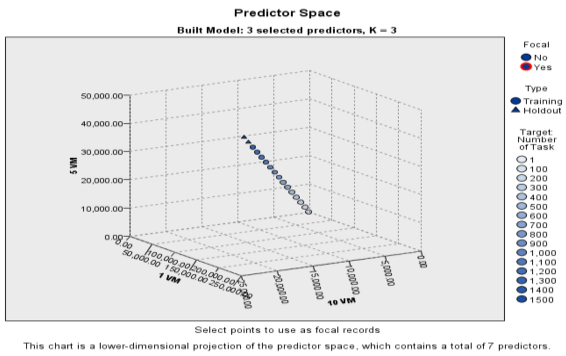


Fig. 2 Nearest neighbor analysis

The constructs for the model included:

- Number of requests or tasks (N)
- The number of VMs denoted as (M)
- Shared resources request denoted as (L)
- Unique resource request denoted as (K)

Thus, the use of indicator variables allowed the study to be expressed in ANOVA procedures as a special case of regression analysis. Both the number of quantitative predictor variables and the number of distinct groups represented in the data by indicator variables were increased.

C. Algorithmic Relationship and Programming

The algorithm defines a set of a step followed to solve a certain task. The role of the algorithm is to balance the resource allocation (M) set of numbers (1,2,3,…M) to the domain of the request of (N) as a set of numbers (1,2,3,…N) and may comprise (K) set of numbers (1,2,3,…K) and (L) set of numbers (1,2,3,…L). The optimization is achieved when each tenant is satisfied, and the tenant does not compromise into other's quota, all VMs being remitted and request services in the shortest time possible (t). Therefore the relationship between t, N, M, K, and L are the basic representation and manipulation of the algorithm. Table 2 Coefficients and Number of VMs

Table 2. Coefficients of different Numbers of VMs

Number of VMs		Unstandardized Coefficients		Standardized Coefficients	T	Sig.
		B	Std. Error	Beta		
1 VM	Number of Task	159.998	.001	1.000	195855.452	.000
	(Constant)	.065	.719		.091	.929
5 VMs	Number of Task	31.971	.016	1.000	1962.779	.000
	(Constant)	29.286	14.340		2.042	.060
10 VMs	Number of Task	15.968	.018	1.000	870.534	.000
	(Constant)	32.836	16.148		2.033	.061
20 VMs	Number of Task	7.966	.019	1.000	411.463	.000
	(Constant)	34.664	17.044		2.034	.061
30 VMs	Number of Task	5.295	.037	1.000	144.234	.000
	(Constant)	-11.774	32.319		-.364	.721
40 VMs	Number of Task	4.375	.039	.999	112.761	.000
	(Constant)	8.402	34.156		.246	.809
50 MVs	Number of Task	4.375	.039	.999	112.761	.000
	(Constant)	8.402	34.156		.246	.809

The independent variable is the number of tasks.

For 1 VM

t1= $\sum_{i=1}^{i=K} 159.998i$……………………………………………….. (i)

t2= $\sum_{i=1}^{i=L} 159.998i$……………………………………………….. (ii)

t= (i) + (ii)

But, $\sum_{i=1}^{i=K} 159.998i + \sum_{j=1}^{j=L} 159.998i = \sum_{i=1}^{i=N} 159.998i$ and 159.998 is approximately 160ns

$\sum_{i=1}^{i=K} 160i = (\sum_{i=1}^{i=L} i + \sum_{j=1}^{j=N} j)\ 160$

t can also be represented as

t= $\sum_{i=1}^{i=N} 160i$…………………………………….………... (iv)

Therefore, t=(N*160) ns for N tenants

For 5 VMs

t= $(\sum_{i=1}^{i=K} i + \sum_{j=1}^{j=L} j)$ 31.97i …………………………………...(v)

t= 31.971(K+L) ns, where (K+L=N)

For 10 VMs

t= $(\sum_{i=1}^{i=K} i + \sum_{j=1}^{j=L} j)$ 15.968 ……………………………….. (vi)

t= 15.968 (K+L) ns, where (K+L=N)

For 20 VMs

t= $(\sum_{i=1}^{i=K} i + \sum_{j=1}^{j=L} j)$ 7.966 ………………………………… (vii)

t= 7.966(K+L) ns, where (K+L=N)

For 30 VMs

t= $(\sum_{i=1}^{i=K} i + \sum_{j=1}^{j=L} j)$ 5.295 …………………………......(viii)

t= 5.295 (K+L) ns, where (K+L=N)

For 40 VMs

t= $(\sum_{i=1}^{i=K} i + \sum_{i=1}^{i=L} j)$ 4.375 …………………….…………. (ix)

t= 4.375(K+L) ns, where (K+L=N)

For 50 VMs

t= $(\sum_{i=1}^{i=K} i + \sum_{j=1}^{j=L} j)$ 4.375 ……………………………........(x)

t= 4.375(K+L) ns, where (K+L=N)

The relation between the number of tasks, VMs and the respective time taken is denoted by equations (iv) to (x) and Table 2. It is realized that, as the number of VM (M) increases, there are corresponding decreases of the constant. The sig. values, on the other hand, increase, indicating that there is a strong positive correlation within the variables under consideration. When the variance of the jth regression coefficient is increased due to the linear association of Xj with other predictor variables relative to the variance, that will result if Xj were not related to them linearly. As Rj tends toward 1, indicating the presence of a linear relationship in the predictor variables, the VIF for hj tends to infinity.

D. Cloud Computing Resource Allocation and Optimization (CCRAO) Model

The technique for developing models is based on the concept/the idea modelled from a given domain. Grounded theory is adequate for model development due to its principal characteristics of model development. This approach builds a context-based, process-oriented description and explanation of the phenomenon, rather than an objective, static description expressed strictly in terms of causality [8]. The architecture of the model provides a blueprint of how the constructs can be integrated to achieve the optimization process. It offers a unified approach for integrating various core components towards achieving an optimized cloud environment resource allocation and access. To test for the goodness of fit of the model, the study adopted (Curve Fit). The multiple Linear Regression model predicts the dependent variable, Analysis of Variance (ANOVA) was used, and the results were as in Table 3. These provide a way of deciding if the evidence is strong enough to support the study.

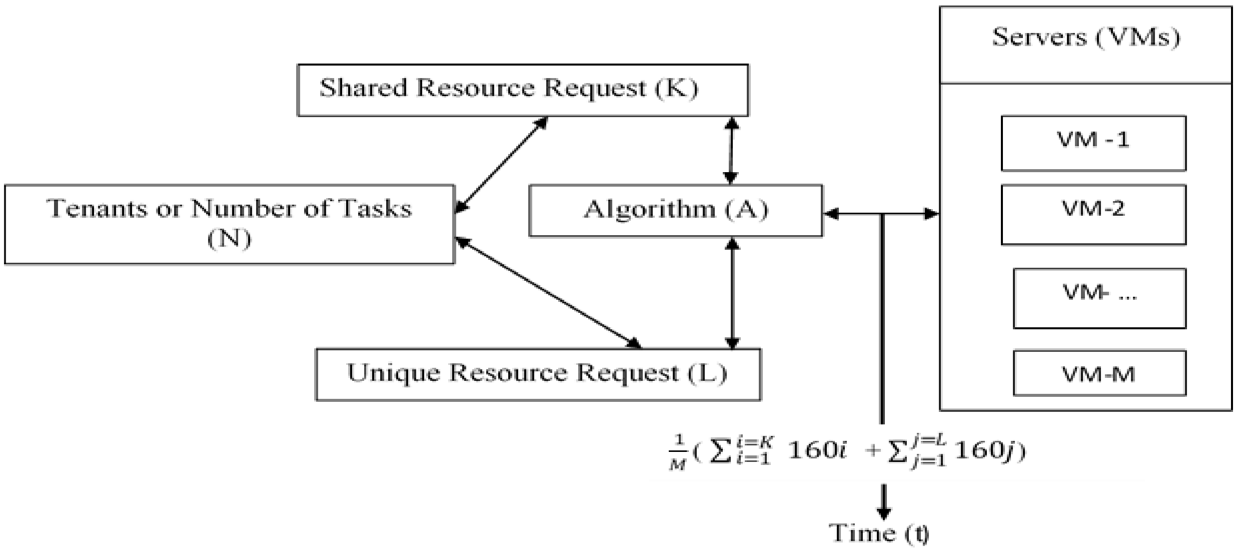
Table 3. Model Summary

No of VMs	R	R Square	Adjusted R Square	Std. The error of the Estimate
1 VM	1.000	1.000	1.000	1.506
	.741	.549	.517	1.240
5 VMs	1.000	1.000	1.000	26.525
	.807	.651	.626	.874
10 VMs	1.000	1.000	1.000	29.852
	.841	.707	.686	.721
20 VMs	1.000	1.000	1.000	35.692
	.877	.768	.752	.573
30 VMs	1.000	.999	.999	67.678
	.901	.812	.798	.490
40 VMs	.999	.999	.999	74.139
	.904	.818	.805	.479
50 VMs	.999	.999	.999	74.139
	.904	.818	.805	.479

The independent variable is the number of tasks.

The values of R can be used to predict how the constructs can be matched by the algorithm. R^2 is used to measure how well the model fits the data. The Model Summary results as in table 5. Indicate that 1 VM has Adjusted $R^2$of .517(51.7%), 5 VMs .626 (62.6%), 10 VMs .686 (68.6%), 20 VMs .752 (75.2%), 30 VMs .728 (72.8%), 40 VMs .805 (80.5%) and 50 VMs .805 (80.5%). All these values are above 50%, with the highest 40 or 50 VMs having .805 (80.5%), which is relatively closer to 1. The p-values present significance in ANOVA results of table 5.1 ($p < 0.05$) for all the sets of the corresponding data; an indication that the model was a good fit to the study data and that the independent variables were good predictors of the study response variable achieve the optimization of resource usage. The significance level tells us how strong the evidence is - with the lower levels indicating stronger evidence.

The study also assumes the dynamism of resources (L and K), Virtual Machines (VMs), (M), and number of Tenants (N) that may affect the allocation process, which may, in turn, affect the processing type or the availability of the resources at the time of request resulting to sharing the resource in the cloud environment as in Figure 3. Summarizes the architecture of the model.


Servers (VMs)
Shared Resource Request (K)
VM -1
VM-2
VM- ...
VM-M
Tenants or Number of Tasks (N)
Algorithm (A)
Unique Resource Request (L)
$\frac{1}{M}(\sum_{i=1}^{i=K} 160i + \sum_{j=1}^{j=L} 160j)$
Time (t)

Source: Author (2020)

Fig. 3 CCRAO model

The model is used to predict resource utilization by various tenants. The role of the algorithm (A) is to allocate or schedule ready task requests from Tenants to the server or VMs. Using the data in the algorithmic analysis in section 5.3, the relation between N, K, L, A, M, and t of the model can be written as equation (1);

$$t = \frac{160(N)}{M} \quad \ldots\ldots (1)$$

Where:

i. N is the number of Tenants having a combination of K (shared resource requests) and L (Unique resource requests). Having a set of numbers 1, 2, 3,. . . N.

ii. A is the Allocation and Optimization Algorithm.

iii. M denoting server machine or Virtual Machine, numbered 1, 2, 3, . . ., M and;

iv. t is the time taken to access a resource in nanoseconds

E. Model Testing and Validation

Validation is perceived as a process and evidence for building the right model. It is considered an important activity being part of the process and model development. This step was undertaken to ensure the model developed is sufficiently accurate for the purpose it was meant for. Validating a model is a process that starts with the researchers, who then seek validation among "outsiders." Presenting an evolving theory at a conference, a seminar, or some other type of academic field provides an excellent opportunity for researchers to discuss and receive feedback [9]The concept of validation ascertains that the research participants determine whether the researcher's interpretation of the meaning and events with their own. This method is used to check on biasness and the quality of research [10]. The model was first programmed using java programming language; the test data were subjected to a simulation process using a cloud simulator on a java Integrated Development Environment (NetBeans IDE Version 8.0.2). The results were later presented to the experts through an organized Focus Group Discussion and thereafter allowed to interrogate the model and indicate their level of agreement and satisfaction with the model and its constructs. The validation was also done through simulated data results and collaboration with other studies on a similar population. The main metrics considered were: number of Tenants, cloud computing technologies (server or VMs), memory space, and the time taken to access the resource by the tenant. These constructs were modelled and presented to experts in three seminars and two conferences to determine whether 1) the constructs provide a clear reflection that underpins the study, 2) whether the constructs represent the domain under the study, 3) whether the constructs under consideration can be modelled in the real world, and 4) whether the model developed would be accepted. Therefore question asking technique was used to get experts' concerns and feedback about the model. The questions of interest were: is the model representative of the real world? Is it an accurate mapping of the concepts underpinning the study? Is it easy to use or apply to the real world, and can it be acceptable?

a) Test data

To validate the model, the study used the following test data for simulation. CloudSim 8 Version 3.0 with 20 VM, RAM was set to 8000mb, and the number of jobs or tasks set to 5. When the simulation was performed, the following was the output:

```
CloudInformationService: Notify all CloudSim entities for
shutting down.
Datacenter_0 is shutting down...
Datacenter_1 is shutting down...
Broker_0 is shutting down...
GlobalBroker_ is shutting down...
Simulation completed.
Simulation completed.
========== OUTPUT ==========
Cloudl   STATUS    Dat  cent    VM   Tim   Start    Finish
et ID              a    er ID   ID   e     Time     Time

0        SUCCESS   3    0       160  0.2   160.02   160.04

1        SUCCESS   3    1       160  0.2   160.02   160.04

2        SUCCESS   3    2       160  0.2   160.02   160.04

3        SUCCESS   3    3       160  0.2   160.02   160.04

4        SUCCESS   4    4       160  0.2   160.02   160.04

6        SUCCESS   3    100     160  200.  160.02   360.04
                                     2

7        SUCCESS   3    101     160  200.  160.02   360.04
                                     2

8        SUCCESS   3    102     160  200.  160.02   360.04
                                     2

9        SUCCESS   3    103     160  200.  160.02   360.04
                                     2

10       SUCCESS   4    104     160  200.  160.02   360.04
                                     2
CloudSim8 finished!
BUILD SUCCESSFUL (total time: 1 second)
```

The test results indicate 20VM, and with a ramset to 8000 Mb, the results indicate that the model is workable.

V. CONCLUSION

The paper has addressed the modelling process of the cloud computing resource optimization model. It has looked at model constructs, which are the parameters of interest in the model construction. These include Tenants denoted as tasks, Cloud Servers denoted as Virtual Machines (VMs), and time is taken to access a requested cloud resource. It also examined the mathematical relationship between the algorithms and the simulated data using ANOVA. The paper thereafter combined the constructs to come up with a cloud computing resource optimization model that can enable cloud environment users to have optimized access and full usage of their resources in their cloud quotas. The study has also looked at the choice of tools, the objectives of the model, the design components, architectural design, development, and validation testing. The simulation model classifies and determines the best way the tasks need to be handled or responded to within the shortest time possible and also make sure that each VMs is engaged at any time in case the requests outnumbers the VMs.

VI. RECOMMENDATION

The cloud environment is an enormous environment composite of both public and private entities as resource providers, the regulations and policies may differ, and this also may affect the tenants. Research is to be carried out to develop frameworks on the cost, time, value, and utilization techniques as a recommender for a specific cloud service in a cloud environment.

VII. REFERENCES

[1] Liu Z., Wynter L., Xia C. H., and Zhang F. Parameter inference of queueing models for it systems using end-to-end measurements, Performance Evaluation, 63(1) (2006)36-60.

[2] Kraft S., S. Pacheco-Sanchez S., Casale G., and Dawson S. Estimating service resource consumption from response time measurements. ICST Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering. ,(2009).

[3] Harris D. Why 'Grid' Doesn't Sell. On-Demand Enterprise blog, (2008). http://www.on-demandenterprise.com/blogs/26058979.html.

[4] NIST Cloud computing Standards Roadmap, Special publication (2) (2013)500-29.

[5] Hashizume et al. An Analysis of Security issues for cloud computing. Journal of Internet Services and Applications. (2013).

[6] Matsui, R. D. The Art of Data Science: A Guide for Anyone Who Works with Data. Retrieved from http://leanpub.com/artofdatascience. (2015).

[7] Bin, F., and Yuan G., Development of Strategy Software and Algorithm Simulators for Multi-Agent System in Dynamic Environments, Technology, and Communication. (2013).

[8] Andersson, A., Hallberg, N., and Timpka, T. Model for interpreting work and information management in process-oriented healthcare organizations, International Journal of Medical Informatics, 72 (2003)47–56.

[9] Jabareen, Y. Building a Conceptual Framework: Philosophy, Definitions, and Procedure. International Journal of Qualitative Methods, (2009)48-62.

[10] Anselemo, P. Technical Factors and User Personality Characteristics as Indicators of Smart Phone Integration into Self-Directed Learning in University. Research Thesis, MMUST. (2014).