IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Mathematical Modelling of Engineering Problems
Vol. 6, No.1, March, 2019, pp. 85-91
Journal homepage: http://iieta.org/Journals/MMEP

An indigenous tool (NoJavaCloud) to handle virtual nodes to simulate the cloud tasks

Thulasi Bikku

CSE, Vignan's Nirula Institute of Technology and Science for Women, Palakaluru, Guntur 522005, Andhra Pradesh, India

Corresponding Author Email: thulasi.jntua@gmail.com

https://doi.org/10.18280/mmep.060111

Received: 25 July 2018
Accepted: 27 August 2018

Keywords:
CloudSim, cloud computing, data management, NoJavaCloud

ABSTRACT

Now a days, computing in real-time for greater timesaving and ease of access made the cloud computing vital in Data Management. In general, the information technology services can be identified as "Software (SaaS), Infrastructure (IaaS) and Platform as Services (PaaS). Cloud can be known for one can access relevant data elsewhere with/without knowing the background of the data handling mechanism. In order to attain efficient clouding, researchers perform cloud simulation. CloudSim, SPECI, CDOSIM and DCSim are some available open-source software packages to help the cloud simulation. Lack of Graphical User Interfaces and Back-End support, Application Security are few major cons with the open-source cloud simulators and the customization towards specific application are really a challenge with the open-source simulators. This work describes an indigenously developed (NoJavaCloud) cloud simulator to resolve the struggles faced with the existing open-source software packages. A typical data set of input parameters resolved and reported in the existing literature is taken as bench-mark problem and same was processed in the indigenously developed (NoJavaCloud) simulator. The main intension of this article is to find the suitability of this indigenous package for real-time application compared with the existing simulator (Cloudsim). Results revealed that, NoJavaCloud produced exactly better or similar simulation that produced by the CloudSim.

1. INTRODUCTION

Cloud computing allows to sharing tasks, data storage, computer services or resources [1]. The "pay as you use" method in Cloud Computing used to pay what we are used and avoid to pay inefficiencies and expense of any unused resources. A particular design worldview that accentuates usage of segments as secluded services that can be found and utilized by clients is known as Service Oriented Architectures (SOAs) [2]. Foundations in view of the SOA standards are called Service Oriented Infrastructures (SOIs). Through the deftness, adaptability, flexibility, quick self-benefit provisioning and virtualization of tools, Service Oriented Architecture standards are reflected into Clouds, which give the capacity to effectively adjust resource provisioning to the dynamic requests of Internet clients. There are three kinds of administrative services in distributed computing IaaS, PaaS and SaaS [3]. Infrastructure as a Service (IaaS) alludes to the equipment like tools and hardware and programming components, for example, a servers, stockpiling and working frameworks like operating systems. Platform as a Service (PaaS) is the arrangement of hardware and instruments and services intended to software advancement and conveying those applications quickly and proficiently. Software as a Service (SaaS) is a product authorizing and dissemination demonstrate as appeared in Figure 1. Distributed computing is a shifted figuring model joining the advantages of service-oriented architecture and utility registering. In distributed computing, asset portion and its legitimate usage, to accomplish a higher throughput and quality of service (QoS), has turned into an incredible research issue [4]. This paper features another cloudlet designation procedure that uses every single accessible resource proficiently and improves the QoS by applying deadline based workload dissemination. It is trusted that this paper would profit both cloud clients and specialists in different perspectives of research. The applications are intended for end clients and conveyed over the web.

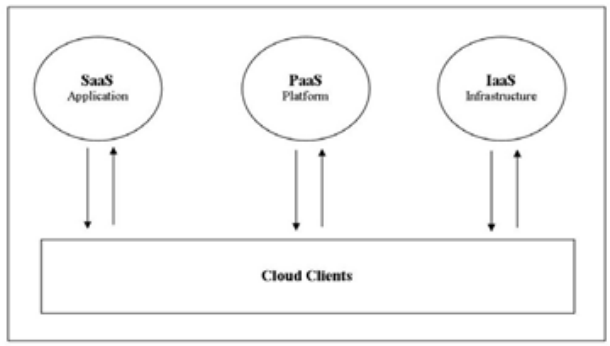

Figure 1. Cloud computing services

2. CLOUDSIM

The CloudSim is exceptionally famous simulation tool to reenact distributed or cloud computing, which empowers demonstrating and recreation of Cloud computing frameworks [5]. The framework and behavior modeling of

Cloud framework segments are supported by Cloudsim. CloudSim does not provide ready accessibility, so the clients need to assess, characterize the required output, and set the input parameters for simulation. The CloudSim exploring particular framework issues without considering the low level subtle elements identified with Cloud-based services and infrastructures as appeared in Figure 2.

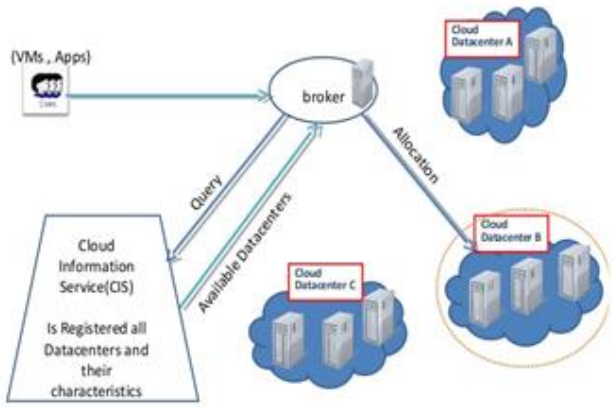

Figure 2. Cloud-based services and infrastructures

The framework and behavior modeling of Cloud framework segments are actualized by Cloudsim toolbox, like Data Centers (DC), virtual machines and resource allocation strategies. It bolsters generic application allocation strategies, which can be interact effortlessly and controlled utilization. Right now, it underpins modeling and simulation of Cloud computing environments comprising of both single and between organized clouds (inter-connected clouds) [11]. Also, it uncovered custom interfaces for actualizing approaches and provisioning systems for distribution of Virtual Machines under inter-connected cloud or distributed computing situations [6].

CloudSim offers the novel properties: (i) bolster for modeling and simulation of extensive scale distributed computing infrastructure, including data centers on a solitary physical computing hub; and (ii) an independent stage for modeling data centers, service brokers, scheduling, and allocations policies.

The highlights of CloudSim, there are (i) accessibility of virtualization motor, which helps in creation and management of multiple, independent, and co-hosted virtualized services on a data center node; and (ii) adaptability to switch between space-shared and time-shared designation of processors to virtualized services.

These convincing highlights of CloudSim would accelerate the improvement of new methods, techniques, and conventions in Cloud Computing, thus contributing towards speedier development of the paradigm [12]. Simulation steps for CloudSim on Java Platform:

CloudSim project follows the steps to implement the specified configuration to start a environment for simulation [7]. To understand the working of CloudSim simulation framework, we need to have the knowledge about these steps.

1. Configure CloudSim with java platform
2. Set the number of users
3. Include the CloudSim library
4. Create new Cloud Information Service (CIS).
5. Create Data Center.
6. Create Process Element list.
7. Create Host List.
8. Create Datacenter object with help of Vm Allocation Policy, storage List.
9. Create Data center Broker
10. Create Virtual Machine instances and submitted to the broker
11. Create cloudlets with parameters.
12. Call the simulation process (Now all the functions are invoked and simulation events are triggered)
13. Stop the simulation process (Now all the entities are shut down)
14. Print the simulation results.

2.1 Bench mark problem identification

2.1.1 Real time implementation issues

Cloud computing enabling on demand network access to a wide variety of resources like software, hardware resources and network etc. These services are provided to the customer by the cloud service provider as per his/her request. In cloud computing need different composition, configuration, and deployment required for web hosting, social networking, content delivery, real time instrumented data processing, research work, government organizations, and academy community [8]. The real challenging tasks in real Cloud computing environment are Task scheduling and resource allocation policies in different application models. Because in the real-time deployment of cloud tests, bed limits the experiments to the scale of the test bed and makes the reproduction of results is more complicated. Cloud testing in a real environment is very expensive, time costly and not repeatable and it is hard to analyze the performance issues on real cloud environments. The Quality-of-service (QoS) management needs to improve in real time side.

2.1.2 Simulation issues

Cloud simulators have different characteristics and functions that can be used to address different Cloud related issues. Several Grid Computing simulators are available like SimGrid that is good for Grid Computing but does not support the model of Cloud Infrastructure. Traditional IT simulators are also available now for the simulation and modeling. Virtualization simulation and modeling simulations are most widely used for the development of private as well as public Cloud models. CloudSim is one the leading Cloud simulator which provides modeling at a large scale. It also supports heterogeneous Grid resources and multiple scheduling applications which run across multiple organizations. The simulation process gives more flexibility compare to the real system, which give freedom to set parameters that provides better grooming to your work [14]. The leading technology of the cloud computing has lot of testing tools are there for research as well as development purpose. Most of the researchers and industry-based developers utilize the third party tools [9]. But these tools are needed to update every day for the real time need, enhancement and bugs. Some tools come with lot of bugs or technical mistakes. Some tools are not support graphical interface. Most of the JAVA based simulators are more complicated in programming as well as huge simulation steps. This kind of problems can change or affect the aim of the research area.

2.2 Bench mark problem

One of the main drawbacks of CloudSim is lack of Graphical User Interfaces (GUI). They used JAVA so the end user should be familiar in java platform than only they can able to finish his process without struggle. The creation of VM, DATA CENTER, PE list, HOST list and policies etc. in CloudSim is not a user friendly approach. Computer background people only use this tool. The coding structure of CloudSim is very complex therefore the researchers need to spend lot of time to learn JAVA as well as CloudSim. The CloudSim did not support backend so the user can store data separately and write bulk code for that. There is no application security so someone easy to simulate the vital research works without any credentials. Hence a cloud simulation tool should be in GUI and easy to create such a VM, HOST, policies etc. in a single click. Mainly the user can utilize the complete feature of the simulator with slight programming knowledge because the researchers should be focused in his research work not in programming side. A Simulator should be enclosed proper credentials with different user level authentications.

2.2.1 Bench mark problem solved with indigenous cloud tool

Hence we developed an indigenous cloud tool for task scheduling for our research work. The architectures of indigenous cloud tool, provides the rising demand for energy-efficient Information Technology strategies, repeatable, and controllable methodologies for evaluation of algorithms, applications, and policies are to be simulated before actual development of cloud products [13]. It is used to analyze complex problems in the physical world. This indigenous cloud tool reduces the cost of ownership and eliminates the IT support costs, expensive hardware purchases and enhancement. The indigenous cloud tool doesn't need to maintain the software license and doesn't need pay the software upgrade costs. The indigenous cloud tool contain attractive GUI feature and it is used to set up repeat simulations easily and also provides various user friendly capabilities that make indigenous cloud tool configurable and more flexible to use. Indigenous cloud tool divide the simulation experiment exercise from programming exercise and used to testing the large scaled Internet applications in a cloud environment.

3. PROPOSED NOJAVACLOUD ARCHITECTURE

The architecture of indigenous cloud tool (NoJavaCloud), which supports modeling and simulation of Cloud-based data center environments. The fundamental issues are allocating the hosts to Virtual Machines, monitoring the state of dynamic systems and application execution are managed by indigenous cloud tool. The indigenous cloud tool architecture consists of the Credential, User Requirements, Policies, Broker, Task, Task Manager, Resource, Resource Manager, Cloud Registry as shown in Figure 3.

Credentials explains about application and user level authentications. Research data is more vital so the indigenous cloud tool contain proper credential to access valuable research data by only the registered user with userid and password. User Requirements contain end user desirable requirements. Policies contain such a scheduling policy, resource allocation polices. Broker acts on behalf of a user and it submitting VM provisioning requests to data centers and submitting the tasks to VMs. Task explains about user's input data. The input data insert to database with help of excel sheet. It is one of the user friendly approaches to feed data to system. Task Manager provide task details such as task length size, required processing power etc. Resource are the Data Center's physical computing node such a Host related details.VM (Virtual machine) contain hardware configuration details like processor, RAM, etc. Resource Manager provides resource details such a VM capability and availability, data center detail, etc. Cloud Information Service (CIS), the cloud registry contain information of available resources such a data center, VMs, task details, user details. This information submits to Task Scheduler.

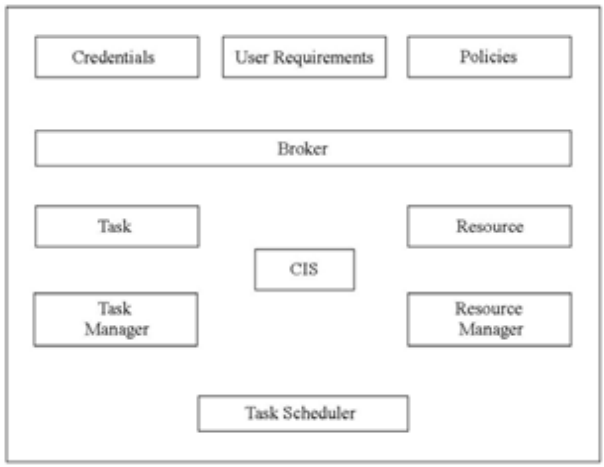

Figure 3. Block diagram for NoJavaCloud tool

Task Scheduler get information from CIS then decides how the accessible CPU assets of virtual machine are partitioned among assignments. The indigenous cloud tool offered two types of policies but researchers and developers can further enhance the policies [15].

In Time-Shared scheduling strategy the assets are shared among the cloudlets. Each cloudlet allocated with resources for execution for a specific timeframe. If the timeframe the assets are detracted from that cloudlet and are allotted to another cloudlet. In Space Shared scheduling strategy the cloudlets does not share assets. A cloudlet claims the assets until the point that it gets executed. We have thought about these two scheduling approaches. Similar approaches are connected for planning of Virtual machines which are running inside a host. In time shared the assets of the hosts are being shared by the virtual machine for an auspicious premise, however in space shared a virtual machine can keep running on a Host just if free handling elements are accessible.

Space-Shared: To assign particular CPU cores to particular VMs. The tasks are arranged in a queue and it is schedule on the given virtual machine. VM completes first task and then take the next task from the queue. If the queue is empty then checks the new task.

ptft= ptst +(tt/cs* ccp)

Where ptft is the Predictable Task Finish Time, ptst is the Predictable Task Start Time, ttis the Total Task, cs is the CPU Speed, ccp is the CPU Core (PEs).

Time-Shared: To powerfully convey the limit of a center among Virtual Machines. All accepted tasks are arranged under the queue then schedule the task concurrently on the virtual machine If the queue is empty I checks for new task.

ptft= ctst + (tt/cs * ccp)

where ptft is the Predictable Task Finish Time,ctst is the Current Task Simulation Time, tt is the Total Task, cs is the CPU Speed, ccp is the CPU Core(PEs).

4. INDIGENOUS CLOUD TOOL SIMULATION STEPS

1. Set the credential details.
2. Create data center and VM.
3. Create policies.
4. Create task with parameters
5. Insert task details.
6. Start simulation process.
7. Print the simulation results.

Pseudo code

1. Find task_count
2. Find vm_count
3. Find ptft
4. if vm_count= <ptft
5. Allocate task ito index virtual machines
6. Remove task I from Queue
7. Update the value of vm_count
8. else
9. Call RESCHEDULE minimum completion time.
10. end if
11. Return

4.1 Illustration of the pseudocode

The task_count calculate the total task allotted for the simulation tool and its details. The vm_count check the total availability of virtual machines and total. Where ptft is the Predictable Task Finish Time, if it is greater than the vm_count then the tasks will not be assigned. If ptft is less than the vm_count, then only the tasks will be assigned to virtual machine, then the procedure starts and returns the output.

4.2 Screen shot for login

Figure 4. Login screen

The user accesses this indigenous cloud tool with entering valid userid and password. If the userid and password matches, then the user can login the tool, else the user cannot login if userid and password doesn't match. A login, logging in screen is used to authenticate the user, in order to access that system after entering of information of the identifier into a system by a user. Login is the vital part of security measures. The login form is shown in Figure 4.

4.3 Screen shot for Vm creation

Cloud Resource contains Datacenter class, whose hostList are virtualized. It manages preparing of Virtual Machine queries rather than handling Cloudlet-related inquiries. Thus, despite the fact that an Alloc Policy will be instantiated (init()), it won't be utilized, as preparing of cloudlets are dealt with by the Cloudlet Scheduler and handling of Virtual Machines are taken care of by the Vm Allocation Policy. The VM creation modules contain 7 columns to create VM with configuration details. The VM Creation contains the details like Datacenter ID, Host ID, VM ID, VM Nmane, MIPS, Size, RAM, and VM Pes as shown in Figure -5.

Figure 5. VM creation

4.4 Screen shot for Vm list

This module used to view the created VM list from the database as shown in Figure 6. When using the OpenStack Nova CLI (command line client), you are able to get a list of all running instances (sometimes called VMs or servers) using the command list: nova list.

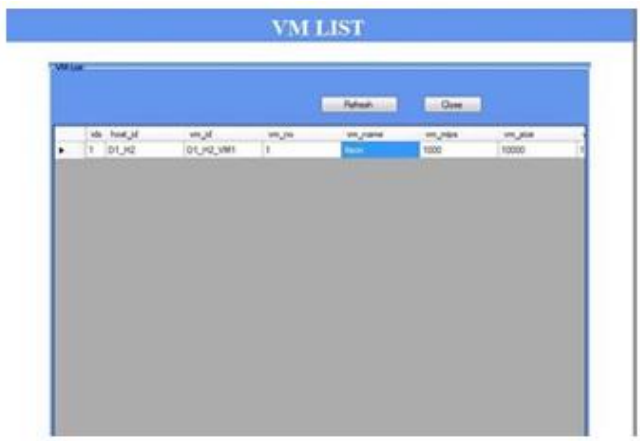

Figure 6. View VM list

This gives a list of all VMs in OpenStack Nova. There

are numerous search options available, including filtering by IP address, status, host, and more. These filters, provide a list of active VMs that is current. The throughput of the disk is assessed by the number of input/output operations per second (IOPS) and measured in terms of Megabytes per Second where MBPS = 10^6 bytes/sec. Disks works in two modes, they are cached or uncached modes. The host cache mode is set to ReadOnly or ReadWrite for stored operations of disk. The host type is set to NONE, for uncached activities of disk.

4.5 Screen shot for task creation

This Task Creation-I module used to create new tasks as shown in Figure 7.

Figure 7. Task creation-I

Figure 8. Task creation-II

When you want to process a task, you must create a new task object and place it on a queue. You can explicitly specify the service and handler that process the task, and optionally pass task-specific data along to the handler. You can also fine-tune the configuration for the task, like scheduling a time in the future when it should be executed or limiting the number of times you want the task to be retried if it fails. This Task Creation-II module used to get the task from excel sheet as shown in Figure 8.

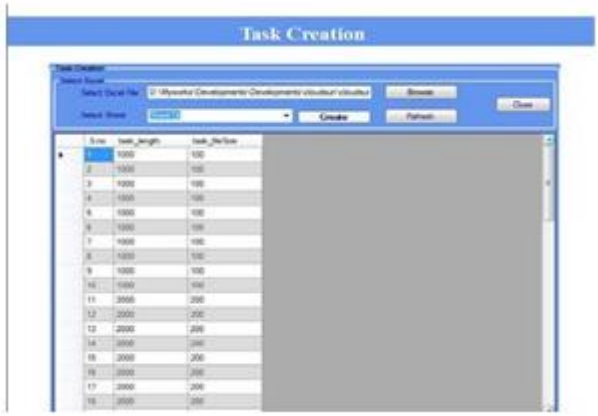

Figure 9. Task creation-III

Figures 7, 8, 9 Shows Task Creation Modules By Browse The Excel Sheet And Uploadthe Task List To Indigenous Cloud Tool.

4.6 Screen shot for VM list

This module used to simulate the task and produce output as shown in Figure 10. Output examination is the displaying stage worried about planning replications, computing statistics from them and showing them in printed or graphical arrangement. A decent outline of reenactment replications enables the analyst to acquire the most factual data from simulation keeps running for the minimum computational cost. Specifically, we look to limit the quantity of replications and their length, and still get dependable insights.

Figure 10. Output screen

5. EXPERIMENTAL RESULTS

5.1 Indigenous cloud tool result compare with CloudSim

The CloudSim experimentation is done in the following environmental conditions, the characteristics of datacenterx86, which is a family of intel 8086 CPU and intel 8088 with linux operating system and virtual machine hypervisor with xen characteristics of host having random access memory of 1024 MB and storage of 500000 MB having a number of processing entities of 1 MIPS CPU speed rating of processing entity of 1000.

The Requirements of Virtual Machine are having MIPS rating of 250 Image size on disk of 10000 Mb capacity with ram of 512 MB and the required PES are 1 and hypervisor with

xen characteristics of the cloudlets 40000 instructions and each input file size is 300kb and output file size of 300kb.

The experimental setup runs cloudlets from 1 to 20 and calculated the aggregate time taken by them to execute in CloudSim and Nojavacloud environments including the virtual machine actualizing the time shared and space shared [10] as shown in table 1.

Table 1. Execution time of CloudSim and Nojavacloud

Cloudlets	CloudSim		NoJavaCloud	
	Time shared	Space shared	Time shared	Space shared
1	160	160	140.96	147.008
2	160	160	140.96	147.008
3	266.666	213.33	234.932746	196.007604
4	320	240	281.92	220.512
5	415.915	288	366.421115	264.6144
6	479.99	320	422.87119	294.016
7	571.425	365.74	503.425425	336.041912
8	640	400	563.84	367.52
9	728.888	344.344	642.150328	316.3832672
10	800	480	704.8	441.024
11	887.264	523.636	781.679584	481.1167568
12	959.984	560	845.745904	514.528
13	1046.123	603.076	921.634363	554.1062288
14	1119.976	640	986.698856	588.032
15	1205.322	682.666	1061.888682	627.2335208
16	1280	720	1127.68	661.536
17	1364.688	762.352	1202.290128	700.4490176
18	1439.968	800	1268.611808	735.04
19	1524.195	842.105	1342.815795	773.726074
20	1600	880	1409.6	808.544

To evaluate the performance of Indigenous Cloud Tool and CloudSim with following parameters. The Indigenous Cloud Tool input parameters get from Journal “Cloud Computing Simulation using CloudSim” [8]. The table2 contains input parameters for CloudSim and NoJavaCloud.

Table 2. Input parameters

Properties	CloudSim	NoJavaCloud
Input File Length	40000	40000
Input File Size	300	300
VM mips	1000	1000
RAM	512MB	512MB
Number of PEs	1	1

5.2 Data flow

The Parameters set and feed to the Cloudsim and Indigenous Cloud Tool (NoJavaCloud). The input data stored in excel sheet of the NoJavaCloud tool, then just browse and insert then simulate the Indigenous Cloud Tool. The results of Start time and Finish time of both environments are shown in table 3.

In table 4 the shows the comparison result of CloudSim and NoJavaCloud and the supporting features.

Indigenous Cloud Tool produces same or better output of CloudSim therefore Indigenous Cloud Tool provides desirable output with GUI feature.

Table 3. Results for simulation in a given environment

Ratio	CloudSim		NoJavaCloud	
	Start Time	Finish Time	Start Time	Finish Time
1.0	0.1	10.54	0.1	5.0
1.5	0.1	6.5	0.1	5.0
2.0	0.1	5.5	0.1	4.0
2.5	0.1	4.4	0.1	3.2
3.0	0.1	3.5	0.1	2.2
3.5	0.1	3.1	0.1	2.0
4.0	0.1	2.5	0.1	2.0
4.5	0.1	2.3	0.1	2.0
5.0	0.1	2.1	0.1	2.0

Table 4. CloudSim and NoJavaCloud comparison

Features	CloudSim	NoJavaCloud
Credential	NO	YES
GUI	NO	YES
Coding Structure	Very Complex	Very Simple
Back End Support	NO	YES
Level of Simulation Steps	Long	Short

6. CONCLUSION

In this paper we have effectively thought about the two fundamental cloud simulator test systems, the "cloudlet scheduler space shared" and "cloudlet scheduler time shared" and observed the degradation of execution time of NoJavaCloud than in cases of CloudSim execution time taken by the cloudlet. At last we have effectively reproduced a heterogeneous cloud environment simulator in which we have placed all the preparing elements similar to MIPS rating in the host and effectively assigned handling resources with various MIPS rating to virtual machines. We have assigned a virtual machine which have the capacity to finish the execution of cloudlet effectively and calculated the aggregate number of cloudlets executed successfully. The experiment results show that the new proposed approach outperforms the current approach as far as the aggregate number of cloudlets executed effectively.

In this paper, indigenous cloud tool (NoJavaCloud) developed has been presented. In the cloud environment it has been specifically designed for simulating the Task Scheduling. The researcher can utilize the NoJavaCloud slight programming knowledge. GUI environment is the fine feature of NoJavaCloud reduces the learning time. This feature reduces the research duration and the researcher only focuses their research area. Application security product the vital research data of the researcher. The NoJavaCloud contain backend so the user feed more data and backup the database. Hence this indigenous cloud tool (NoJavaCloud) is better than CloudSim in so many aspects.

REFERENCES

[1] Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, Lee G, Patterson D, Rabkin A, Stoica I, Zaharia M. (2010). A view of cloud computing. Communications of the ACM 53(4): 50-58.

[2] Papazoglou MP. (2003). Service-oriented computing: Concepts, characteristics and directions. In Web Information Systems Engineering 2003. WISE 2003.

Proceedings of the Fourth International Conference on, IEEE, pp. 3-12.

[3] Dillon T, Wu C, Chang E. (2010). April. Cloud computing: Issues and challenges. In Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, pp. 27-33.

[4] Upadhyaya J, Ahuja NJ. (2017). February. Quality of service in cloud computing in higher education: A critical survey and innovative model. In I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2017 International Conference on, IEEE, pp. 137-140.

[5] Calheiros RN, Rajiv R, Anton B, César AF De R, Rajkumar B. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience 41(1): 23-50.

[6] Wickremasinghe B, Calheiros RN, Buyya R. (2010). Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, IEEE, pp. 446-452.

[7] Buyya R, Ranjan R, Calheiros RN. (2009). Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities. In High Performance Computing & Simulation, 2009. HPCS'09. International Conference on, IEEE, pp. 1-11.

[8] Goga K, Terzo O, Ruiu P, Xhafa F. (2014). Simulation, modeling, and performance evaluation tools for cloud applications. In Complex, Intelligent and Software Intensive Systems (CISIS), 2014 Eighth International Conference on, IEEE, pp. 226-232.

[9] Splieth M, Bosse S, Schulz C, Turowski K. (2015). Analyzing the effects of load distribution algorithms on energy consumption of servers in cloud data centers. In Wirtschaftsinformatik, 287-301.

[10] Ghanbari S, Mohamed O. (2012). A priority based job scheduling algorithm in cloud computing. Procedia Engineering 50(1): 778-785.

[11] Guo ZH, Geoffrey F, Mo Z. (2012). Investigation of data locality in mapreduce. In Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), IEEE, pp. 419-426.

[12] Sakellari G, Loukas G. (2013). A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. Simulation Modelling Practice and Theory 39: 92-103.

[13] Kaur R, Ghumman NS. (2015). A survey and comparison of various cloud simulators available for cloud environment. Int. J. Adv. Res. Comput. Commun. Eng. 4(5): 605-608.

[14] Kumar P, Rai AK. (2014). An overview and survey of various cloud simulation tools. International Journal of Global Research in Computer Science (UGC Approved Journal) 5(1): 24-26.

[15] Guzek M, Bouvry P, Talbi EG. (2015). A survey of evolutionary computation for resource management of processing in cloud computing. IEEE Computational Intelligence Magazine 10(2): 53-67.