suggests that cloud computing introduces a new way of making computations. Clouds are sometimes perceived as marketing hype and a way of integrating and selling technologies developed so far such as grid computing, virtualization, service provision, etc. At most they are an evolution of these technologies with added focus on simplicity and usability. Additionally, they usually propose a certain business model. The main reason of this is the fact that the emergence of clouds was led by vendors and providers who advertised their offers as a cloud computing service rather than by academia and research communities (as it was in case of Grids) [2].

Cloud computing may be viewed as a resource available as a service for virtual data centers, but cloud computing and virtual data centers are not the same. For example, consider Amazon's S3 Storage Service. This is a data storage service designed for use across the Internet (i.e., the cloud). It is designed to make web-scale computing easier for developers. According to Amazon: Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, fast, inexpensive data storage infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers [3].

The paper begins by presenting cloud computing an area of rapid evolution. Second, it presents generalized and extensible simulation framework: CloudSim. This simulator enables seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application services [4]. Third, we present our broker policy to deal some problems encountered in CloudSim version b1.0. In section 5, we evaluate the performance of policy presented and report the results obtained finally we conclude the paper.

2 CLOUD COMPUTING

Type The cloud is not simply the latest fashionable term for the Internet. Though the Internet is a necessary foundation for the cloud, the cloud is something more than the Internet. The cloud is where you go to use technology when you need it, for as long as you need it, and not a minute more. You do not install anything on your desktop, and you do not pay for the technology when you are not using it [5]. According to the IEEE Computer Society Cloud Computing is:

"*A paradigm in which information is permanently stored in servers on the Internet and cached temporarily on clients that include desktops, entertainment centers, table computers, notebooks, wall computers, handhelds, etc*."

"Cloud Computing is a type of parallel and distributed system that consists of a collection of computers interconnected, virtualized and presented as a single processing unit based on a SLA with a negotiation mechanism between the service provider and the consumer of this service" [6].

Cloud computing is typically divided into three levels of service offerings: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a service (IaaS). These levels support virtualization and management of differing levels of the solution stack [7].

Software as a Service (SaaS)

SaaS is a type of cloud computing that delivers applications through a browser to thousands of customers using a multiuser architecture. The focus for SaaS is on the end user as opposed to managed services (described below). For the customer, there are no up-front investment costs in servers or software licensing. For the service provider, with just one product to maintain, costs are relatively low compared to the costs incurred with a conventional hosting model. An example of this model is Google Apps, which provides online access via a web browser to the most common office, and business applications used today, all the while keeping the software and user data stored on Google servers.

Platform-as-a-Service (PaaS)

PaaS is yet another variation of SaaS. Sometimes referred to simply as web services in the cloud, PaaS is closely related to SaaS but delivers a platform from which to work rather than an application to work with. These service providers offer application programming interfaces (APIs) that enable developers to exploit functionality over the Internet, rather than delivering full-blown applications. This variation of cloud computing delivers development environments to programmers, analysts, and software engineers as a service. Mostly known example of Platform as a Service model (PaaS) is provided by Google. Google has a service called "Google App Engine" that allows users to code their applications rapidly without any integration needs.

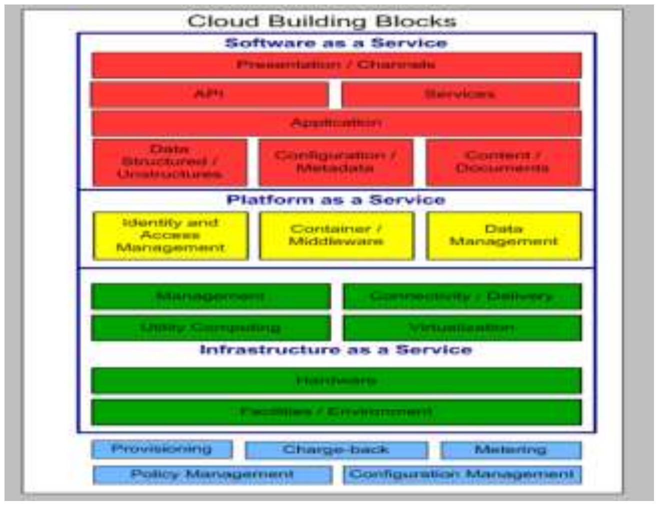


Figure 1. Oracle private cloud [6]

Infrastructure-as-a-Service (IaaS)

According to Google [8], "Google App Engine makes it easy to build an application that runs reliably, even under heavy load and with large amounts of data".

IaaS is the delivery of computer infrastructure (typically a platform virtualization environment) as a service. Rather than purchasing servers, software, data center space or network equipment, clients instead buy those resources as a fully outsourced service. The service is typically billed on a utility computing basis and amount of resources consumed (and therefore the cost) will typically reflect the level of activity. It is an evolution of virtual private server offerings [9].

Amazon is one of the big players in IaaS platform. They base their system on virtualization. Amazon Web Services (AWS) offers their cloud computing infrastructure to different sizes of companies. Companies request to have their infrastructure and hardware in a specified term and they have full access over all systems that they request.

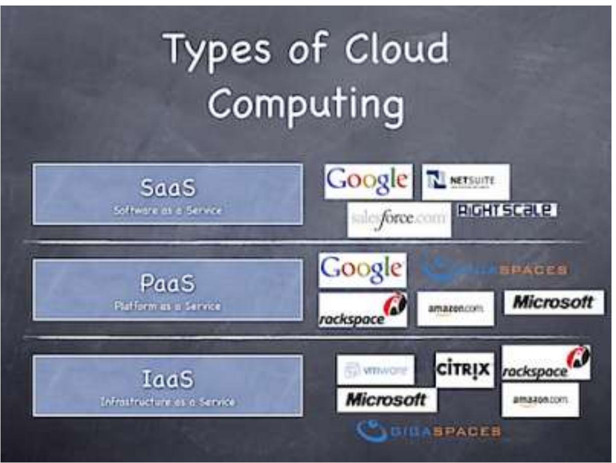


Figure 1. Cloud Services [18]

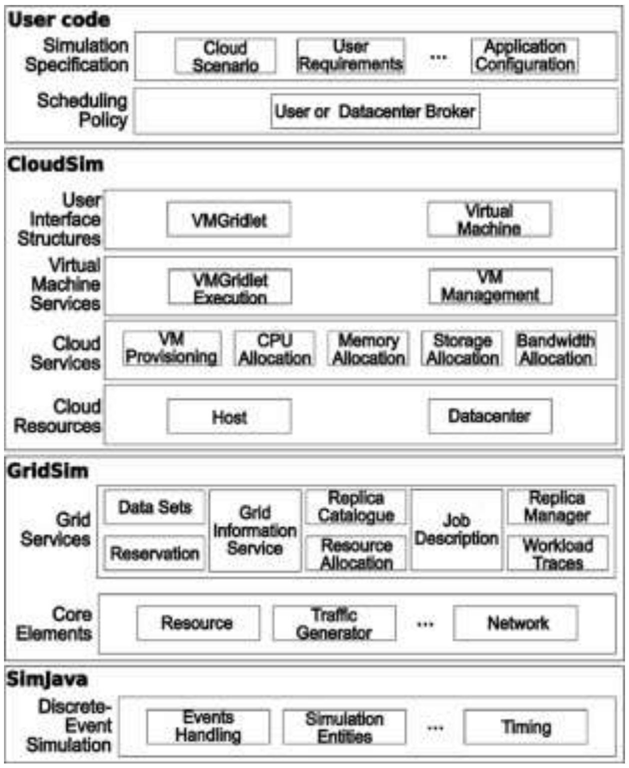
3 CLOUDSIM TOOLKIT

The recent efforts to design and develop Cloud technologies focus on defining novel methods, policies and mechanisms for efficiently managing Cloud infrastructures. To test these newly developed methods and policies, researchers need tools that allow them to evaluate the hypothesis prior to real deployment in an environment where one can reproduce tests. Especially in the case of Cloud computing, where access to the infrastructure incurs payments in real currency, simulation-based approaches offer significant benefits, as it allows Cloud developers to test performance of their provisioning and service delivery policies in repeatable and controllable environment free of cost, and to tune the performance bottlenecks before deploying on real Clouds.

CloudSim is a new, generalized, and extensible simulation framework that enables seamless modeling, simulation, and experimentation of emerging Cloud computing infrastructures and application services.

CloudSim offers the following novel features: (i) support for modeling and simulation of large scale Cloud computing infrastructure, including data centers on a single physical computing node; and (ii) a self-contained platform for modeling data centers, service brokers, scheduling, and allocations policies.

Figure 2 shows the conception of the CloudSim toolkit [10]. At the lowest layer, we find the SimJava [11] that



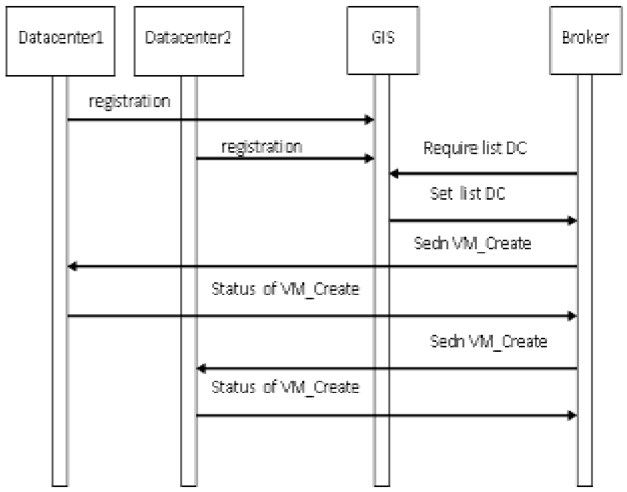
implements the core functionalities required for higher-level simulation such as queuing and processing of events, creation of system components (services, host, Datacenter, broker, virtual machines), communication between components, and management of the simulation clock.

Figure 3. CloudSim Architecture [4]

In the next layer follows the GridSim toolkit that support high level components for modeling multiple Grid components Such as networks, resources, and information services. The CloudSim is implemented as layer by extending the core functionality of the GridSim layer. CloudSim provides support for modeling and simulation of virtualized Datacenters environments such as management interfaces for VMs, memory, storage, and bandwidth. CloudSim layer manages the creation and execution of core entities (VMs, hosts, Datacenters, application) during the simulation period. This layer handle the provisioning of hosts to VMs based on user requests, managing application execution, and dynamic monitoring. The final layer in the simulation stack is the User Code that exposes configuration functionality for hosts (number of machines, their specification and so on), applications (number of tasks and their requirements), VMs, number of users and their application types, and broker scheduling policies. A Cloud application developer can write an application configurations and Cloud scenarios at this layer to perform a cloud computing scenario simulations [1, 10].
In CloudSim, Cloudlet models the Cloud-based application services (content delivery, social networking, business workflow), which are commonly deployed in the data centers. CloudSim represents the complexity of an application in terms of its computational requirements. Every application component has a pre-assigned instruction length (inherited from GridSim's Gridlet component) and amount of data transfer (both pre and post fetches) that needs to be undertaken for successfully hosting the application [6].

4 PROPOSED APPROACH

When a user submits a service request to the cloud, the request will first arrive at the cloud management system (CMS) which maintains a request queue. If the queue is not full, the request will enter the queue; otherwise it will be dropped and the requested service fails. A request that successfully enters the queue will then reach a scheduler which will divide the request into several subtasks and assign the subtasks to different processing nodes (nodes). Generally there are multiple schedulers which are homogeneous with similar structures, schemes and equipments, to serve the requests. After all subtasks are assigned to corresponding nodes, they will be exec uted on the node s. Duri ng the exec ution of subta sks, there



may be exchange of data through communication channels. After all subtasks are completed, the results are returned and integrated into a final output, which is sent to the user [12].
During the operation of the Simulator CloudSim in our research work for resources management, several problems and inconstancies were identified.

Problems identified in CloudSim Version 1 are:

- When we try to create multiple VMs in several data center, some VMs cannot be created due to a saturation of resource in data centers and cloudlets assigned to this virtual machines not created are lost.
  The creation of VMs in the data center is done sequentially. We send as much vms as possible for this datacenter before trying the next one. when the datacenter is saturated, we try to create in the next and so on, until no longer have DataCenter available, the message “Creation of VM # __ failed in DataCenter__” appear and cloudlets submitted to these VMs are not run (even after the creation of VM).
- There is no link between Datacenters. This lack of links will lead necessarily to a lack of communication between them and therefore no exchange of any service or information on the load of each for a possible load balancing policy. Our solution was to create a "ring topology" of the data center (each datacenter knows his predecessor and his successor).

Figure 4. Sequence Diagram of CloudSim before optimization

Given that each data center has a number of resources available and ready to be dynamically allocated to each request, this number is still limited, which we refer to the first problem. Skip to next Datacenter for allocation of other resources for other VMs, not necessarily the most optimal or most economical for the owner of the data center,

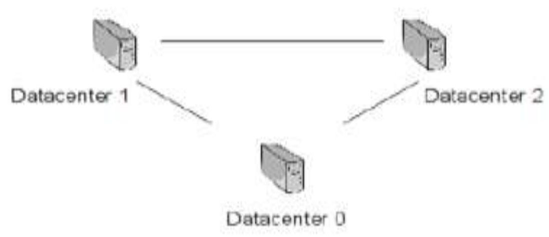


Figure 5. Ring Topology

The main goal for him is to use the Max resources to maximize its gain (or benefit). Indeed, there may have resources already available in the data center, but not enough to create a VM with the number requested.
Our solution is to exploit these resources that are more or less wasted, to gather them from different data center and use it to create VMs that this time are on two or more data center. This solution is not possible if there are no links between the different datacenters that can exchange services and information on these VMs. We have defined for this a search method; this method calculates the number of free resources in each datacenters. In a possible creation of VM fails, an event is triggered to call *search method* with a calculation of "threshold" minimum to make a decision of whether or not a particular data center.

It was considered necessary to take into account a minimum threshold to exceed for eventual decision to take this or that DataCenter for allocated a virtual machine for three reasons:

- To minimize the cost of transfer due to exchanges of different services for a single VM allocated in several Datacenters
- To reduce the number of Datacenters allocated to a VM: we will take only datacenters with a large number of resources available.
- To minimize run time.

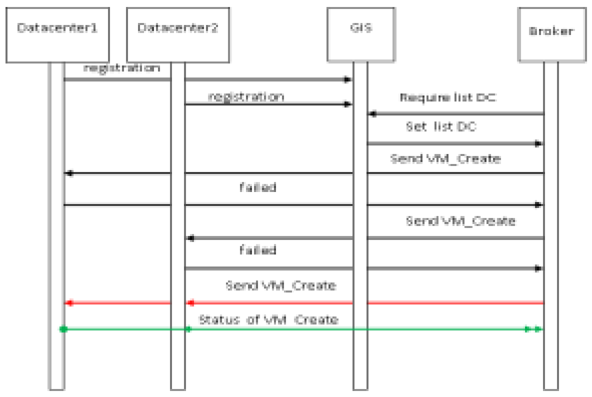


Figure 6. Sequence Diagram of CloudSim Optimized

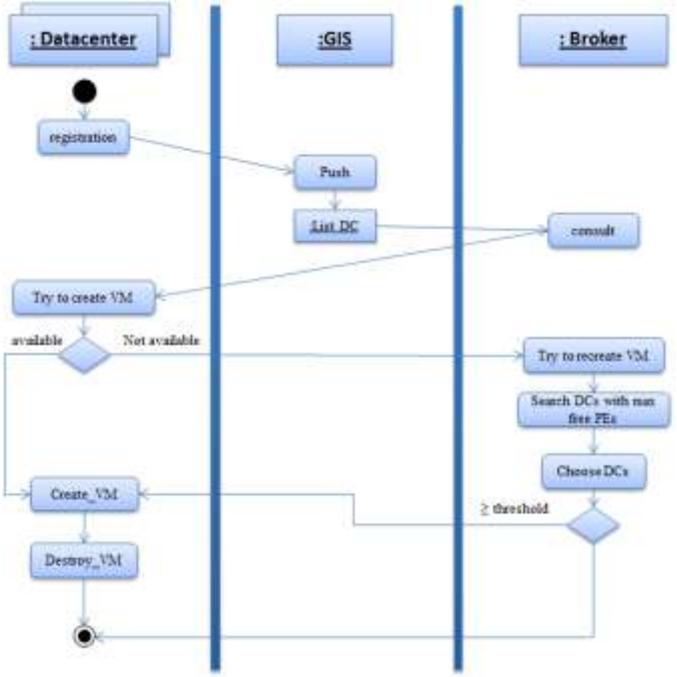


Figure 7. Activity Diagram of CloudSim Optimized

The following figures (see Fig 8.a, Fig 8.b, and Fig 8.d) show an example of simulation for an environment containing four datacenters with 4 hosts each, we try to create five VMs with the following characteristics:

VMID	*Number Of CPUs*
1	5
2	2
3	4
4	3
5	1

Table 1. The description of virtual machines

5 SIMULATION STUDY

At the beginning of simulation, Datacenters are recorded in a directory called GIS. Broker acting on behalf of a user consults this directory and obtains the list of all Datacenters. It sends to the

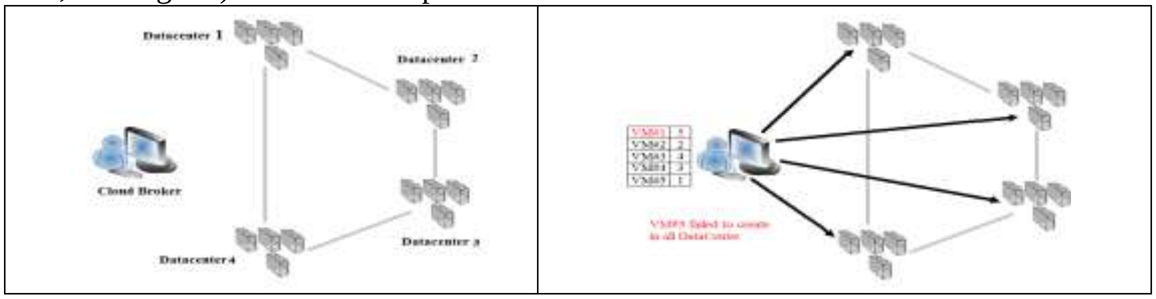


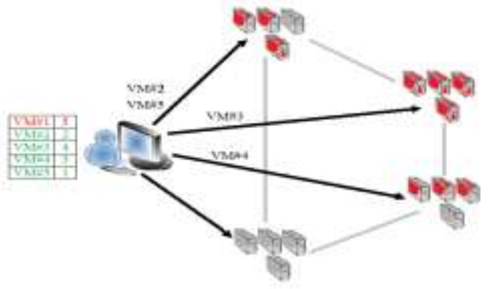
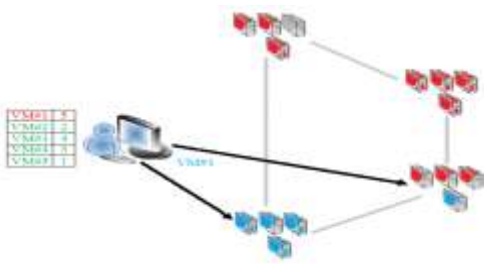
Fig 8. a	**Fig 8. b**
 **Fig 8. c**	 **Fig 8. d**

Figure 8. Example of simulation with four datacenters

first element of this list a VM_create message (to create the virtual machine and to make there carry out Cloudlet), if this creation fails, it remakes the same thing with second Datacenter and so on.
If Broker traverses the entire list and finds all Datacenters taken, CloudSim declares a "failed" and the request (the request) of the user is rejected. [4]
The goal of the first experiment is to calculate the number of failure. The simulation environment consisted of 6 data centers with 5 hosts, where each host was modeled to have one CPU core (1000MIPS), 2GB of RAM memory. We vary the number of virtual machine that want to create, VMs having following constraints: 512MB of physical memory, 3 CPU core.

As evident from Figures 9 the numbers of creation of VMs failed increase so rejection of cloudlet submitted to these VMs increase when the numbers of virtual machines increase.
The number of virtual machine that can be simultaneously created is 10 and we can create only 6 VMs so we have 12 PEs not used (40 % of resources not used).

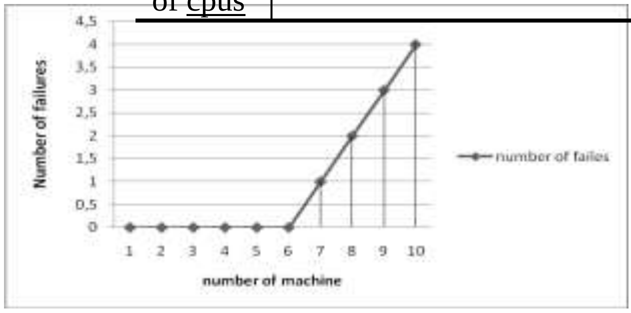


Figure 9. Number of failures in CloudSim

In the second experiment, we fixed the number of datacenter at 15 and measured the number of failures for the simple version and our extension. We calculated the number of resource waste (free resources not used) for both versions.

VMID	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
Number of cpus	4	4	4	8	8	4	4	4	4	4	8	8

VMID	*13*	*14*	*15*	*16*	*17*	*18*	*19*	*20*	*21*	*22*	*23*	*24*
Number of cpus	8	8	4	8	4	8	8	4	8	4	4	8

VMID	*25*	*26*	*27*	*28*	*29*	*30*	*31*	*32*	*33*	*34*	*35*	*36*
Number of cpus	8	8	4	8	4	4	8	8	8	8	8	8

Table 2. The description of virtual machines

For the tests, we created 15 datacenter s. each datacenter contain 15 hosts, where each host have 2GB of memory, 1 processor with 1000 MIPS of capacity and the description of virtual machines that want to create is given in the table 2.

In Figure 10 and 11, we present a summary of some of these tests. The figure 10 represents the number of failure and the Figure 11 shows the resource waste for both versions (the simple and extended version)

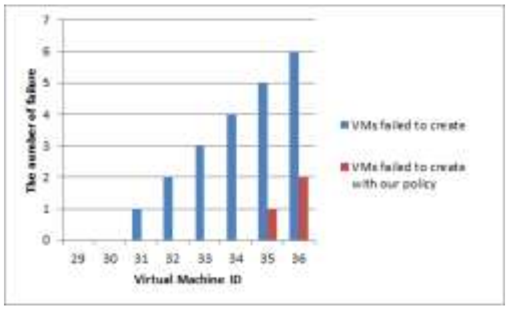


Figure 10. Number of VMs failed to create

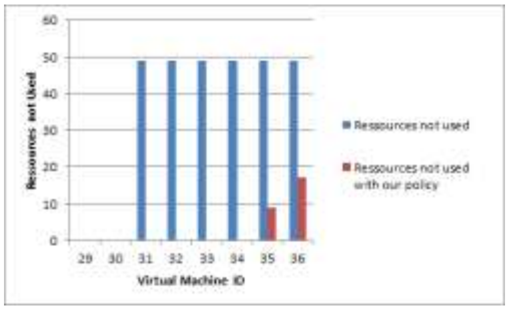


Figure 11. Number of waste-resources

The Figure 12 and 13 contains the results for four experiments with different simulation environments. Figure 12 shows the maximum number of VM's that can be created and figure 13 shows the total cost benefit for the simple version and our extension. In all these tests, we try to create the maximum number of virtual machine for both versions and we associate one Cloudlet to each VM to be executed. Each Cloudlet is modeled to be having 40000 MIs.

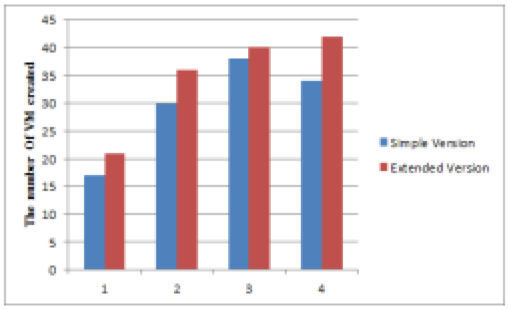


Figure 12. Number of VM's that can be created

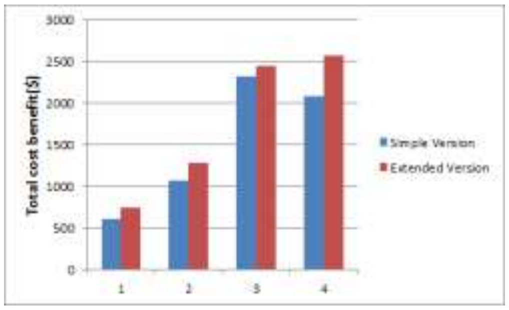


Figure 13. The total cost benefit

The results show that our policy attains much better performance and better resources utilization than the policy used in the version 1. The cases to have VMs failed is not excluded, but significantly reduced.

6 CONCLUSIONS

The recent efforts to design and develop Cloud technologies focus on defining novel methods, policies and mechanisms for efficiently managing Cloud infrastructures. The principal objective that we aimed at solving; it is the satisfaction of the users, while avoiding rejecting some requests subjected by the customers of Cloud Computing. We have developed own broker policy to create virtual machine in one or more datacenters. We also introduced the links between datacenters to exchange services and information on virtual machines; we use simulation to compare the performance of our proposed policy with the policy version B.0.5 of CloudSim. The simulation results demonstrated that: our policy leads to better performance and better resources utilization.

Several future tracks can be the subject of extension of this work in our laboratory. We can quote: i) To equip the first approach by an agent which makes it possible to select most suitable Datacenter for the assumption of responsibility of the requests; ii) To extend the second approach by a module of balancing enters the various files associated with Cloudlets of Datacenters; iii) Integrating semantic aspect in the cloud can improve performance and quality of service [17], something that already proved in several works in the semantic grid [14,16]; iv) To use a system of multi-agents for negotiation enters the agents with an aim of migrating Cloudlets of Datacenter towards another; v) Improving and optimizing the economic models used for resource management in CloudSim as the business model of English auction [13] and the double auction economic model [15].

7 REFERENCES

1. Cloud Services from http://www.capstonets.com/Cloud_Services.html
2. Rittinghouse, J-W., Ransome, J-F.: Cloud Computing: Implementation, Management, and Security, CRC Press Taylor& Francis Group, 2009.
3. Amazon Simple Storage Service (Amazon S3), http://aws.amazon.com/s3/
4. Belalem, G., Tayeb, F-Z., Zoaui, W.: Approaches to Improve the Resources Management in the Simulator CloudSim, In: Rongbo, R., Zhang, Y., Liu, B., Liu, C. (eds.), Information Computing and Applications - First International Conference, ICICA'10, Tangshan, China, October 15-18, Lecture Notes in: Computer Science (LNCS), vol. 6377, pp. 189-196, 2010.
5. George Reese. : Cloud Application Architectures, April 2009.
6. Buyya, R., Ranjan, R., Calheiros, R.N.: Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit
7. Oracle White Paper. : Architectural Strategies for Cloud Computing, August 2009
8. Google App Engine, http://code.google.com/intl/fr/appengine/docs/ whatisgoogleappengine .html
9. Cloud computing. From Wikipedia, http://en.wikipedia.org/wiki/Cloud_computing
10. Buyya, R., Murshed, M.: GridSim: A Toolkit for the Modeling and Simulation of Distributed Resource Management and Scheduling for Grid Computing, The Journal of Concurrency and Computation: Practice and Experience (CCPE), 14(13-15), pp. 1175–1220, 2002.
11. Howell, F., Mcnab, R.: SimJava:A discrete event simulation library for java, In Proceedings of the first International Conference on Web-Based Modeling and Simulation, 1998.
12. Yang, B., Tan, F., Dai, Y., Guo, S.: Performance Evaluation of Cloud Service Considering Fault Recovery. In CloudCom(2009) 571-576.

13. Badica, A., Badica, C.: Formalizing Agent-Based English Auctions Using Finite State Process Algebra. Journal of Universal Computer Science , 14(7), pp. 1118-1135, 2008.
14. Flahive, A., Apduhan, B. O., Rahayu, J. W., Taniar, D.: Large scale ontology tailoring and simulation in the Semantic Grid Environment, International Journal of Metadata, Semantics and Ontologies (IJMSO) 1(4), pp. 265-281, 2006.
15. Joita, L. , Rana, O. F., Gray W. A. and Miles, J. C.: A Double Auction Economic Model for Grid Services, In Proceeding Euro-Par 2004 Parallel Processing, 10th International Euro-Par Conference, Pisa, Italy, August 31-September 3, LNCS Vol. 3149, pp.409-416, 2004.
16. Kotsis, G., Taniar, D., Khalil Ibrahim I., Pardede, E.: Information Integration on Web based Applications and Services, Journal of Universal Computer Science (J. UCS), 15(10), pp. 2026-2027, 2009.
17. Mika, P., Tummarello, G.: Web Semantic in the Clouds, IEEE Intelligent Systems journal, 23(5), pp.82-87, 2008.
18. Types of Cloud services & Microsoft Cloud, http://www.thewindowsclub.com/types-of-cloud-services-microsoft-cloud