Proposals of Architecture for Adapting Cloud Computing Services to User's Context

Kanga Koffi[1(✉)], Babri Michel[2], Goore Bi Tra[2], and Brou Konan Marcelin[2]

[1] Ecole Doctorale Polytechnique de l'Institut Nationale Polytechnique Félix Houphouët Boigny (EDP/INPHB), UMRI 78: Electronique et Electricité Appliquée Laboratoire de Recherche en Informatique et Télécommunication, Yamoussoukro, Ivory Coast
koffi.kanga@larit.net

[2] Institut National Polytechnique Félix Houphouët Boigny (INPHB), UMRI 78: Electronique et Electricité Appliquée Laboratoire de Recherche en Informatique et Télécommunication, Yamoussoukro, Ivory Coast
{michel.babri,goore,kmbrou}@inphb.edu.ci

Abstract. In cloud computing service providers offer services to be used by customers. Given the increasing number of clients and also the variety of their needs, some users adjust services to their context. Also context may differ from those services. For this, we propose in this paper a set of tools (architecture and algorithm) for achieving this adaptation taking into account the specificities related to the context of the user. This set, knowing the services offered by suppliers, enables the user to make an adjustment using methods to construct the user context and the research service related to this context. The result obtained after the different tests carried out shows that our tool could be a better simulation environment of research and selection service adaption like in other environments.

Keywords: Cloud computing · CloudAdapt · Cloud architecture · Adaptation of services · Context awareness · Cloud computing simulator

1 Introduction

Talking about service adaptation in cloud computing deserves explanations. In fact cloud computing is a service delivery model which involves two types of actors which are suppliers and customers. In this model, clients who are the users pay proportionally to their consumption just like water, electricity gas, fuel and other services. In this condition, cloud computing becomes the fifth supply model like the last mentioned.

Today, with the advent of internet of thing and also given the high mobility rate of cloud computing services users in the process of service consumption, users would like to adjust those services to their context. Context could refer to desired functional characteristic that is to say the working material environment (CPU value, screen size, Ram value etc. …) and the geographic location of users. This work aims at finding a tool to carry out experiments work to adapt services to the context of the user. In other words

T.F. Bissyande and O. Sie (Eds.): AFRICOMM 2016, LNICST 208, pp. 98–110, 2018.
https://doi.org/10.1007/978-3-319-66742-3_10

it aims to develop a set of tools for performing an adaptation of service. In other words it is meant to set up a simulator for performing an adaptation of services. In the literature, several simulation tools exist. However, those tools are used to carry out precise experimentations due to the diversity of their architecture (network, ecological environment, etc. ….). Thus, in Sect. 2 of this article, we shall present the simulation tool in cloud environment that are known to us. We shall end this section with a summary on those tools by highlighting the common points and also possible differences. In Sect. 3, the research problem will be presented in order to justify the completion of this current work. As for Sect. 4 it will give way to our simulation platform by presenting its architecture, its functioning and differents components necessary for its implementation and the underlying data model to achieve the adaptive function assigned to it. We shall conclude in Sect. 6 after discussion in Sect. 5 while generating possible future works.

2 State of the Art

In research, any theory must go through an experimentation or simulation phase before its validation. Thus, research works in cloud computing found their experimentation through a simulator. In the literature several simulators (tools) exist. Here we will present some of them we consider representative for the area in which these tool are used and their architecture without forgetting the various features.

2.1 Cloud Simulator Architecture

2.1.1 Cloudsim Architecture

The cloudsim architecture is shown on Fig. 1. This architecture has a layered structure. At the lower level is its simulation engine SimJava, which allows the implementation of required functions for a high level simulation. On the upper level (network) are the network management function between the different Data centers. As for the resources they are in this architecture from the previous allocation services (VM, CPU, MEMORY and bandwidth) in this architecture of cloudsim there is also the presence of "broker" and "cloudlet". In fact brokers manage the creation and the destruction of

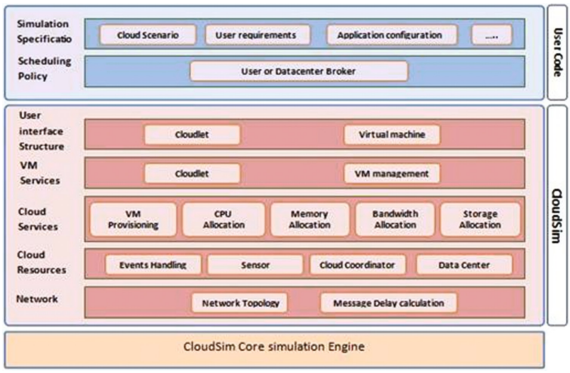

Fig. 1. Architecture of CloudSim [7]

VM. In its uppermost layer there is the user code outlining the characteristics of physical machine (number of machine and their technical specificity).

2.1.2 GreenCloud Architecture [11, 15, 20]

GreenCloud here is an extension of the simulator network NS-2. Basically speaking, it implements energy management in data centers environment. The objective of this simulator is to reduce energy consumption at the data center level while facilitating the consumption of cloud services by users. Also this simulator appears as the basis for the design, manufacture, use and disposal of IT resources with minimal environmental damage. Its includes a datacenter (containing physical machines), cloud computing users (who consume the services), or the switch nodes (for communication between different cloud spread infrastructure)

2.1.3 Architecture of CloudAnalyst [11]

CloudAnalyst is an extension of CloudSim [5]. Therefore, its architecture (Fig. 3) would be inspired and included that of Cloudsim. The difference between the two architectures lies in the integration of a graphical user interface (GUI) in that of Cloudanalyst. Mathematically speaking, the architecture of CloudAnalyst is the sum of that of cloudsim and graphical interface management functions of GUI (Fig. 4). (Fig. 2)

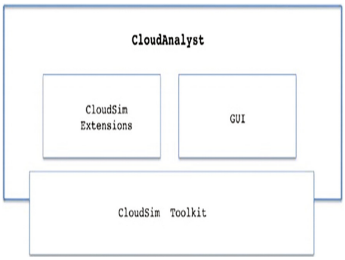

Fig. 2. Architecture of CloudAnalyst

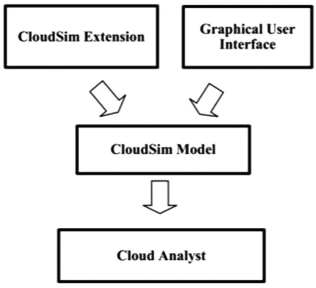

Fig. 3. Perspective view of CloudAnalyst [13]

2.1.4 Architecture NetWorkcloudSim [11]

According to these authors, this simulator has been designed to overcome the shortcomings of cloudsim, GreenSim and MDSim regarding their ability to allow deployment of network aplications in a cloud. So NetworkCloudSim is a simulator that supports communication between an application and the various elements of the network from a cloud computing. It offers two levels of planning communication tasks between the devices of cloud computing service consumers and available virtual machines. To do this, NetworkCloudSim would be appropriate to simulate a networking protocol for applications in the cloud.

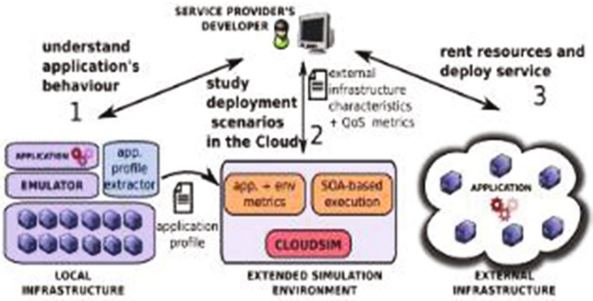

Fig. 4. Architecture EMUSIM

2.1.5 Architecture EMUSIM

This tool was developed to predict the behavior of the applications hosted on cloud computing platforms. Its implementation has required the use of two systems: AEF (Automated Emulation Framework) for emulation applications and CloudSim for the simulation of cloud computing. In its architecture (Fig. 4), we notice the presence of cloud service providers, users, local and external infrastructure to assess the behavior of applications.

2.1.6 Architecture DCSim (DataCenter Simulator)

This tool was developed to overcome the need for evaluation of datacenter management systems. It offers its members a virtual IaaS infrastructure. In its architecture there is the presence of datacenter containing each of the physical machines. Each physical machine hosting multiple virtual machines. On these machines are installed applications.

2.1.7 Architecture GroudSim (Grid and Cloud Simulator)

This simulator is an event-based tool (any changes that might occur in the grids or cloud computing) designed for grid computing applications. For event management it uses threads. According to its authors, given its reputation acquired in various areas of IaaS, it could be used in cloud computing management through IaaS and consideration

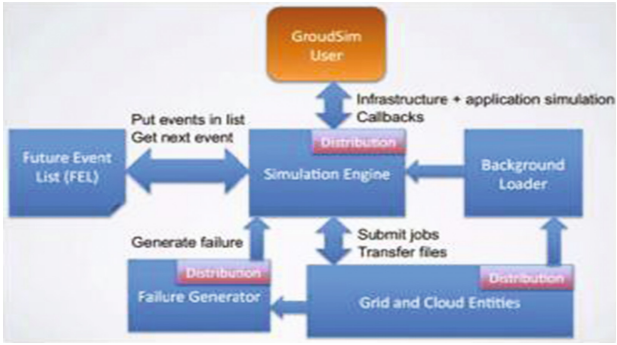

Fig. 5. Architecture of GroudSim

platforms PaaS and SaaS. Subsequently, the authors were seeking to integrate an Askalon to allow users to perform simulation and experiment in real environment. After an analysis of its architecture in Fig. 5, GroudSim included cloud events (withdrawal physical machine to VM), users (User GroudSim), objects and entity of grid or cloud computing.

2.1.8 Architecture of Open Cirrus

According to (the authors), this tool would be a testbed sponsored by HP, Intel and YAHOO in collaboration with some organizations. It is a free tool designed for cloud computing research focuses on the design, planning and management of resources at the Datacenter. The designers of this testbed have set the following objectives:

- boost research at the level of cloud computing system
- encourage research on new cloud computing applications
- to produce data collections in cloud computing
- make available to users of **source code** and **APIs** necessary to enable services (Application s) to interact with him.

2.1.9 Limitation of these Architectures

Given these architectures, one notices that their different authors have made huge contributions. However, some aspects of the services, users and their contexts and their profiles have not been taken into account.

3 Research Problem

In this literature review, it appears that excellent research works have been done. Those works have led to the development of cloud computing simulators of which we have presented the most representative in terms of their features and interesting architectures. However, these simulators still suffering limitation making it impossible to perform, some work including the adaptation or adjustment of services taking into account the specificities related to the user and his working environment, its physical characteristics and also the conditions under which a service could be consumed. The question that arises in this conditions is the following: is there a tool for achieving adaptation or adjustment of services according to taste and user's context? If not, which architecture could have such a simulator and also what would be the data model for the operation of this architecture. Also to make this simulator smart, what could be the different tools (algorithms) necessary for its perfect operation?

4 Contribution

The state of the art we presented reveals the significant level of quality of research conducted in order to establish a more complete cloud computing simulation framework. However some aspects, profile and context for both the user and the services have not been taken into account to our knowledge. To do this we are going present a

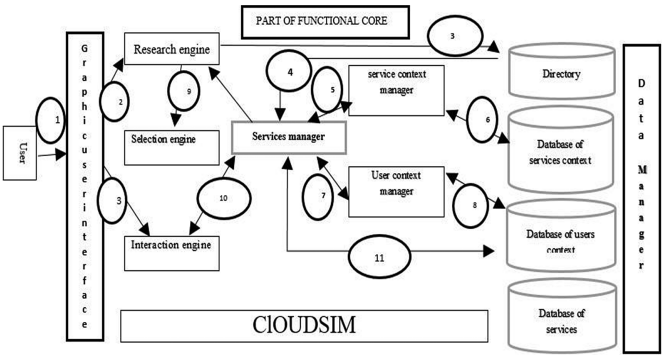

Fig. 6. Architecture of our proposal

tool with methods and algorithms to give intelligence to the architecture shown in Fig. 6, and data models to the research, the selection of services and interaction with the user.

4.1 Architecture of Our Proposal

Our architecture consists of two main parts:

- GUI: it allows dialogue or interaction between the user and the tool through the part of the user queries emissions and the simulator answers
- a functional kernel consists of a data manager and a functionality manager or application manager. In this second part, *the data Manager* consists of the sources of management of different contextual elements that are: service settings and user settings without forgetting services directories and data sent from the services. As for *functionality manager,* it abounds in research functions and service selection as well as interactions included during these phases.

4.2 Our Proposal Operation

The functioning of our architecture follows the steps explained below.

Step 1: First, the user connects to the cloud platform based on information related to his profile (email, name, password, etc.).

Step 2: After this step, the user can conduct the research services to consume **(2)**; or interact with services **(3)**.

Step 3: If the user is looking for services, service manager of the platform receives from the service directory the list of services relevant to the user context **(4)**.

Step 4: In **(5)** the user comes into contact with the service context manager to request different services contexts actually stored in the context database service **(6)**. Also the service manager contacts the context user manager **(7)** to request the context of the existing user in the context of database thereof **(8)**. If the desired service is found, it can be consumed if befits the context thereof.

Step 5: **(10 and 11)** as for the interaction with the service used, the service manager exposes the functionality of the latter so that they can be used by the user. In this conditions, the data from the consumption of these services are sent to the manager from the database services.

4.2.1 Role of the Directory Layer

This layer is the first layer of the platform. It acts as description record figure by providing opportunities and service publication easiness for the cloud provider. In other words, it represents a database of various services descriptions. This description contains various information about the features of the differents services likely to be consumed. These features are used to check if the services that the user wants to consume are compatible or adapted to its context. Therefore, the directory sends the list of services adapted to the context of the user via the service manager.

4.2.2 Role of the Service Manager

The service manager is the second layer of architecture. He is responsible for managing the process of adaptation and restitution of adaptation results (relevant services) to the user. Indeed, this layer is responsible for managing adaptation during the search for service to find a list of relevant services in the context of the user and also during interacting with the Cloud computing services to dynamically adapt the services after possible changes occurring in the context of the user (change of location, change the level of energy for moving objects, change of system operation, size of the ram, etc.)

4.2.3 Role of the Context Layer

In a context of restitution to the user of cloud services a service adapted to its context and the change of the same context, we take charge in our platform the service context so as to improve the quality of adaptation. To do this, we endow this layer with two (2) elements: the context of service and the user context. The context of service is responsible for managing the different contexts in which a cloud service can be consumed. Thereby, he receives from cloud providers the differents contexts in which each service can be used; it stores these settings in a database (DB service context) and then restores them through the service manager when services related are sought by users. As for the user's context manager, it captures the context of a user and the stock in a database (DB user context). In this context the database user is invoked using procedures, functions and components we have set up for this purpose.

4.3 Method of Construction of the User Context Profile

In the cloud, the consumption of a service is achieved through the user's connection to the provider's service platform. So in our architecture platform, this connection is made from a graphical interface in which the user declares preferences of services corresponding to his taste consumption. In our architecture; we classify preferences in the following categories:

– preferences related to service content **(Pc)**
– preferences related to the display of data, network characteristics, the cost of services, language, etc. **(Pa)**

Thus defined Pa and Pc, the entire global preferences is obtained by the union of the two previous categories. Let **P = U PaPa** with **Pc = {P1, P2… Pn}.** Preferences being defined, we define the context based information related to the user's environment, his working device, his network at his session and also his balance. This information is represented as follows:

C is set of user context and Cj is element of the user's context (with $j \in [1, k]$, where k is the maximum number of context item).

The needs of service consumer being in constant change and also given the large number of services offered to users by suppliers, possible conflicts between the preferences (Pa) and contextual information could occur. In that way we offer in our platform CloudAdapt an algorithm called **UserContexte(0 algorithm) is used** to overcome such problems. This algorithm has the role to check whether each **Pi** preference of the user corresponds or not to contextual information. Therefore, it helps determine preferences compatible with the contextual information of the user when - tries to connect with the cloud platform. In this connection, our platform recovers **(Capture (SC, EC))** session's information **(SC), environment context (EC),** its working device **(DC)** and network **(CN)** and stores **(Save ((SC, EC), UC)** in a database (user context). (Fig. 7)

```
Algorithm0 : UserContext
Begin
// user identification
Capture (SC, EC)
Save ((SC, EC), UC)
        If firstconnection then
                Enter characteristiquestatistique (CStat, UC) // name, email, etc.
                Save (CStat, UC)
                Capture (DC, NC)
                Save ((DC, NC), UC)
                Enterpreference (P)
                Firstreference(Pj)
        Else
                Capture (DC, NC)
                Save (DC, NC)
                Enterpreference (Pj)
                SecondPrference (Pj)
        End if
end
```

Fig. 7. Pseudo code of usercontext

Operation of the "UserContext" algorithm:

This algorithm in its operation goes through the following steps:

Step1: **user identification**

In this step, the user entered his login, password and other information likely to help in his identification through the procedure **Capture (SC, EC)** and then stores the backup or through the **Save** procedure **((SC, EC) UC).**

Step2: **checking connections**
At this stage our algorithm checks whether the user is logging in on our platform for the first time.

If this is the *case, step 1* is executed and then the user is invited to make known their preferences through **EntrerPreferences** procedure **(P).** In this case the procedure **FirstPreference (Pj) (Fig.** 8**)** in turn analyzes the preferences to determine which is compatible with the characteristics of the user environment to a successful adaptation of this preference. This algorithm takes as input the set of user preferences and produces at its output two (2) preference lists with their content in the overall set of user preferences. Indeed, the set of lists are: the list of preferences to satisfy **(LPSat)** and those that cannot be met **(LPNSat).**

```
Algorithm1. FirstPreference
Input : p = set of user preference
Output : LPSat = list of user preference for satisfaction
LPNSat  = liste containing preferences which cannot be satisfied
Begin
J is integer = 0
n is integer // number of preference
        While j ≤ n
                If can_satisfyPj,(Pj) then
                Add (Pj, LPSat)
                Else
                Add (Pj, LPNSat)
                Endif
        J ++
        End while
Ruturn (LPSat)
Return (LPNSat)
End
```

Fig. 8. Pseudo code of FirstPreference

If the user is not at his first connection, our algorithm uses the **SecondPreference** procedure **(Pj) (Fig.** 9**).**

Here the algorithm of **Fig.** 8 takes as input the set **P** of all preferences and **LNPSat** list containing the preferences that cannot be met. This list is included for user preferences and contexts ranging from significant way given the high rate of user mobility and different changes of implementation settings and service consumption, the list could undergo LNPSat real change. This algorithm produces for its part two (2) lists **(LNPSat** and **LPSat)** which contain preferences that cannot be met again and compatible preferences easy to meet. To produce these two lists, this algorithm **"Secondpreference"** research in **P** the preferences compatible with the characteristics of the user's environment and the latter does not exist in **LPNSat.** Depending on the case, this preference may or not be added to LPNSat. If **Pj** belongs to **LNPSat** with other values then it will be removed from and added to **LNPSatLPSat.**

If **LPSat**, have multiple values, our algorithm gives priority to the new value set by the user during the current session compared to values from **LPNSat.** However if after

```
Algorithm2. SecondPreference
Input : p = set of user preference
LPSat = list of user preference for satisfaction
LPNSat  = liste containing preferences which cannot be satisfied
Output : LPSat = New list of user preference for satisfaction
LPNSat  = New liste containing preferences which cannot be satisfied
Begin
J is integer = 0
n is integer // number of preference
VP(j) is array of string
        While j ≤ n
                If can_satisfy,(Pj) then
                If LPNSat∋ V(Pj) Then
                delete (Pj, LPNSat)
                Else
                        If can_satisfy(VPj) then
                        delete (Pj, LPNSat)
                        endif
                    add (Pj, LPSat)
                End if
                //
                If LNPSat contain Pj whith other values then
                        If LPNSat ∋ V(Pj) Then
                        delete (Pj, LPNSat)
                        add (Pj, LPSat)
                        Else
                        add (Pj, LPNSat)
                         End if
                End if
        J ++
        End while
Ruturn (LPSat)
Return (LPNSat)
End
```

Fig. 9. Algorithm SecondPreference

connecting to the platform the user does not specify preferences, the algorithm automatically takes into account the preferences stored in previous contexts.

4.4 Service Research Method

To return a list of relevant cloud computing services in the context of the user, we endowed our architecture with research functions to perform this task. In this way, this research function uses the components of the layer are the followings **"service manager", "applicant"** and **"adaptive model"** component.

The component "applicant" is responsible for finding and sending a list of items representing the contexts of services found in the second component is the **"adaptive model".**

4.4.1 Service Research Steps

In the process it performs the following steps:

Step 1: It receives from the service directory a list of relevant services in relation to the request sent by the user.

Step 2: It contacts in his turn the service context manager to request the services found in the list of services provided by the directory.

Step 3: It receives from the service context manager, the different contexts in a given format.

Step 4: He sends different documents (files) to the second component that is "adaptive model".
At this stage, rest of the search of appropriate services is devoted to the second component that follow.

Step 5: Reception of the document containing services contexts of **step 4.**

Step 6: The component "adaptive model" ask the context layer and more specifically to the manager component of the context of the user, the current user context. He then receives from this manager the current context of the user in a precise format.

Step 7: Set the correspondence between the context of the current user and for each service matching the user query. It is performed through a comparison between the content elements of both documents (containing both services and user contexts).

Based on this comparison, the algorithm returns to the user the appropriate services to its context, therefore likely to be consumed.

5 Discussion

In order to contribute to the ever growing success of cloud computing, research must develop tools for performing laboratory work experiments. Also given the increasing cost of simulation environments, it would be appropriate to devise and propose alternatives. In this paper, the recommended an alternative tool to make an adjustment services through a new architecture, data models and algorithms to give intelligence to this architecture. So what does a cloud provider get by adopting this tool? Also what do customers earn by consuming cloud services hosted on infrastructure using such architecture?

In fact, such an architecture could allow a cloud computing provider to detect and store the different contexts of use of a service and also to capture the contexts of users during the consumption of services. As for the customer, he can benefit from the consumption of services compatible with its context (may vary depending on the mobility of the client). As for the researcher, our architecture will enable him to analyze the effects (quality of service, energy consumption level) that could be produced by using our tool and its architecture.

6 Conclusion

The objective in this paper was to propose an architecture and algorithm allowing the provision of services to adapt operations as profiles and contexts related not only to users but also to the different cloud services. To do this first we had reviewed and presented in the different architectures and objects of database model for these

architectures. Then we presented our architecture, structure, operation and algorithms for the construction of the context profile. This algorithm is part of a set of tools for the coordination and operation of our architecture. The analysis of this architecture shows that the appropriate services from the latter satisfy the needs of the user if their contexts of use are consistent with those of their consumers. Also taking into account aspects related to data storage could be used to adjust the size of services to be consumed and working tools of users.

References

1. Farh, M.: A low approach é e agents for the allocation of resources in the Cloud Computing, Doctoral dissertation, University é Mohamed Khider-Biskra (2015)
2. H umane, P., Varshapriya, J.N.: Simulation of cloud infrastructure using cloudsim simulator: a practical approach for researchers. In: 2015 International Conference on Smart Technologies and Management for Computing, Communications, Controls, Energy and Materials (ICSTM), pp. 207–211. IEEE, May 2015
3. Krzywda, J., Tärneberg, W., ÖStBerG, P.O., Kihl, M., Elmroth, E.: TelcoClouds: modelling and simulation. In: Closer (2015)
4. Kecskemeti, G.: Dissect-CF: a simulator to foster energy-aware scheduling in infrastructure clouds. Simul. Model. Pract. Theory **58**, 188–218 (2015)
5. Kaur, R., Ghumman, N.S.: A survey and comparison of various cloud simulators available for cloud environment. Int. J. Adv. Res. Comput. Commun. Eng. **4**(5), 605–608 (2015)
6. Malik, A., Bilal, K., Malik, S., Anwar, Z., Aziz, K., Kliazovich, D., Buyya, R.: CloudNetSim++: a GUI based framework for modeling and simulation of data centers in OMNeT ++, 29 October 2015
7. Tian, W., Xu, M., Chen, A., Li, G., Wang, X., Chen, Y.: Open-source simulators for Cloud computing: comparative study and challenging issues. Simul. Model. Pract. Theory **58**, 239–254 (2015)
8. Serrano, N., Gallardo, G., Hernantes, J.: Infrastructure as a service and cloud technologies. IEEE Softw. **2**, 30–36 (2015)
9. García-Galán, J., Trinidad, P., Rana, O.F., Ruiz-Cortés, A.: Automated configuration supporting infrastructure for migrating to the cloud. Fut. Gener. Comput. Syst. **55**, 200–212 (2015)
10. Sá, T.T., Calheiros, R.N., Gomes, D.G.: CloudReports: an extensible simulation tool for energy-aware cloud computing environments. In: Mahmood, Z. (ed.) cloud computing, pp. 127–142. Springer, Cham (2014)
11. Ahmed, A., Sabyasachi, A.S.: Cloud computing simulators: a detailed survey and future direction. In: 2014 IEEE International Advance Computing Conference (IACC), pp. 866–872. IEEE, February 2014
12. Liu, J., Zhou, Y., Zhang, D., Fang, Y., Han, W., Zhang, Y.: Muclouds: parallel simulator for large-scale cloud computing systems. In: Ubiquitous, December 2014
13. Malhotra, R., Jain, P.: Study and comparison of cloudsim simulators in the cloud computing. SIJ Trans. Comput. Sci. Eng. Appl. **3**(9), 347–350 (2013)
14. Ray, S., De Sarkar, A.: Execution analysis of load balancing algorithms in cloud computing environment. Int. J. Cloud Comput. Serv. Archit. (IJCCSA) **2**(5), 1–13 (2012)
15. Zhao, W., Peng, Y., Xie, F., Dai, Z.: Modeling and simulation of cloud computing: a review. In: 2012 IEEE Asia Pacific Cloud Computing Congress (APCloudCC), pp. 20–24. IEEE, November 2012

16. Rak, M., Cuomo, A., Villano, U.: Mjades: concurrent simulation in the cloud. In: 2012 Sixth International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 853–860. IEEE, July 2012
17. Fittkau, F., Frey, S., Hasselbring, W.: CDOSim: simulating cloud deployment options for migration software support. In: 2012 6th IEEE International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), pp. 37–46. IEEE, September 2012
18. Wickremasinghe, B., Calheiros, R.N., Buyya, R.: Cloudanalyst: a cloudsim-based visual modeller for Analysing cloud computing environments and applications. In: 2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA), pp. 446–452. IEEE, April 2010
19. Papakos, P., Capra, L., Rosenblum, D.S.: Volare: context- aware adaptive cloud service discovery for mobile systems. In: Proceedings of the 9th International Workshop on Adaptive and Reflective Middleware, pp. 32–38. ACM, November 2010
20. Liu, L., Wang, H., Liu, X., Jin, X., He, W.B., Wang, Q.B., Chen, Y.: GreenCloud: a new architecture for green data center. In: Proceedings of the 6th International Conference on Autonomic Computing Industry Session and Communications Industry Session, pp. 29–38. ACM, June 2009
21. Wickremasinghe, B.: CloudAnalyst: A CloudSim-based tool for modeling and analysis of large scale cloud computing environments. MEDC Proj. Rep. **22**(6), 433–659 (2009)