Human Readable Scenario Specification for Automated Creation of Simulations on CloudSim

Manoel C. Silva Filho[1,2] * and Joel José P. C. Rodrigues[2]

[1] Instituto Federal de Educação do Tocantins (IFTO)
Palmas, Tocantins, 77021-090, Brazil
mcampos@ifto.edu.br
http://www.ifto.edu.br
[2] University of Beira Interior (UBI)
Covilhã, Castelo Branco, Portugal 6201-001
d1365@ubi.pt,joeljr@ieee.org
http://di.ubi.pt

Abstract. Cloud Computing is a widely used computing model which enables customers to deliver services to their end users, with reduced IT management and costs. However, the deployment of applications for the cloud may require the design of the needed environment. CloudSim is a very well-known simulation tool to project these cloud scenarios. However, each scenario creation requires programming efforts that are time-consuming, error prone, not necessarily reusable and commonly only feasible for experienced programmers. This paper presents a tool to define cloud scenarios using YAML files and automates the creation of simulations at CloudSim. The use of human readable YAML data format allows the entire split of the scenario specification and the simulation execution and allows non-programmers to clear and directly define the simulation scenarios, facilitating the scenarios sharing, extension and reuse.

Keywords: CloudSim, Simulation, Automation, YAML, Scenarios, Java

1 Introduction

Cloud Computing is a consolidated way to provide computing services to customers in a pay-per-use basis. The customers account with virtually unlimited resources which can be used at any time according to the demand of services hosted at the cloud provider. Cloud resources are managed autonomously to enable the multi-tenancy of the hosts, typically using virtualization [1–10].

However, the deployment of applications for the cloud may require tests and experiments prior to the stage where they are turned available for the users. These tests may define the feasibility of the project, but experiments in a real cloud scenario may not be easy and represent a cost which may be avoided. The use of simulation tools such as CloudSim [11] facilitates the cloud computing research process, avoiding costs and wasting time with other details that are

* Scholarship from CAPES - Proc. no 13585/13-4.

not related to the desired research topic. For customers who want to deploy applications to cloud, the resources required by the application can be measured, demand fluctuations can be tested and costs can be estimated in an easy and predicted way.

The creation of simulation scenarios at CloudSim tool requires programming [12, 11]. So, the researcher will waste time with concerns that are not directly related to the scenario definition and he/she does not have a split between the scenario specification and the code needed to create and execute it. Every change in the scenario may require new changes on the code, so the researcher cannot only focus the attention on the problem that he/she wants to solve, such as creating new algorithms to load balancing, new tasks and virtual machines (VM) scheduling policies, VM placement, resource provisioning, workload prediction, server consolidation, energy efficiency, cost reduction and so on. Besides, the definition of complex simulation scenarios may not be clear to others researches or to customers that will pay for a service that is being developed. This makes difficult the creation of new simulation scenarios, due to the needed programming, and does not give a complete overview of the created scenario.

The current paper presents a solution to solve these problems. It tries to completely avoid the use of programming to define cloud simulation scenarios at CloudSim. The proposed tool allows the specification of simulation scenarios to be defined using a YAML file, that is a pretty human readable data format. The proposed tool was developed using Java 7 and is available under GNU GPLv3 License at `http://github.com/manoelcampos/CloudSimAutomation/`.

The paper is organised as follows. The related works are reviewed in Section 2. Section 3 presents the available tools used in this work while the proposed approach is described in Section 4. Results are discussed in Section 5. And finally, Section 6 includes the conclusions of this work and proposes several future works.

2 Related Work

There is a restrict set of original and independent cloud computing simulation tools. Some of the most recent ones are presented in the section.

The CloudSched simulation tool is presented in [13]. It is a simulation tool aiming the evaluation of resource scheduling for cloud computing applications. The simulator can be used to experiment different scheduling policies and resource allocation schemes. The tool focuses only in scheduling VMs at the IaaS layer. It has a graphical user interface (GUI) to create the simulation setup and it allows the specification of scenarios using files, but in non-standard and undocumented text file format.

The tool is developed in Java, available for multi-platform and is freely available on the Web (http://cloudsched.sourceforge.net) but it is not open source. So, this limits the tool extension, for instance, forbidding external developers to create and evaluate new scheduling policies.

The GUI allows non-programmers to define simulation scenarios but it is overly simplistic and very limited. In the version available to download, different from the paper, there is only one window where the simulation scenario is specified. At this window, there are only the limited quantity of 3 physical machines (PM) and 8 virtual machine (VM) types. Despite the fact that it is possible to change the configuration using the text files, the number of PM's and VM's is fixed, what severally limits the tool usefulness to simulate real cloud environments. The results are only presented with graphs that are unclear and without the tool source code, they cannot be improved. The scheduling algorithms, despite of many, are pre-defined too.

A survey on cloud-based simulation is presented in [14]. In this kind of simulation the tools run into a real cloud environment instead of the developer local machine. These simulation tools provide new cloud computing service models such as modeling as a service (MaaS), execution as a service (EaaS), analysis as a service (AaaS) and simulation resources as a service (SRaaS). These new service models enable developers to use simulation tools as a service, like software as a service (SaaS) model, but called simulation as a service (SIMaaS).

This work presents an architecture to cloud-based simulation and reports about a prototype which, apparently, is not available on the Web. This kind of simulation is helpful to perform cumbersome simulations directly in the cloud. It releases the developer to configure a simulation environment in his/her local computer and can reduce simulation time using more powerful machines in the cloud. However, the use of simulation as a service (SIMaaS) will represent an additional cost if the developer needs to pay for this service. Not all institutions and research groups can afford this cost, even less develop their own SIMaaS infrastructure. Moreover, depending on the tools available in the SIMaaS provider, to specify and execute the simulation it can require the use of some application programming interfaces (API's), like in the traditional cloud services such as Amazon WS (AWS).

In [15], the design and development of a cloud-based simulation tool is presented, following the simulation service models presented at [14]. However, there is no mentioned a concrete tool or service freely available.

The Stratus cloud computing simulation framework is presented in [16], inside the SciCloud project (http://ds.cs.ut.ee/research/scicloud) [17]. It aims to provide simulations of data processing for scientific applications on the cloud, like the established Hadoop MapReduce framework, but using the bulk synchronous parallel (BSP) computing model. One of the main focus of the framework is to simplify the work for researchers not experts in developing distributed applications. So, the efforts can be concentrated to solve a specific problem, leaving the framework accountable of the parallelization to accomplish the task. The framework uses the elasticity benefits of cloud computing to on-demand provisioning resources for scientific applications. The parallelization task will be realized automatically by the framework using the BSP approach. So, the main goal is to provide a framework for running cumbersome scientific applications on the cloud, not a more general framework to simulate and test different cloud computing features like scheduling algorithms, VM placement and live migration, and so on.

3 Used Simulation Tools

The next sub-sections present the simulation tools used in this work. These tools were chosen because they are open source and widely used tools.

3.1 CloudSim Simulation Tool

CloudSim is an open source tool for modeling and simulation of cloud environments. It was developed on the top of GridSim simulation toolkit, that uses the SimJava discrete event simulation engine [12]. It simplifies the process of asses large cloud environments and the test of algorithms for cloud, such as tasks and VM scheduling, VM placement and migration, load balancing and so on.

However, the creation of cloud environments, and each object on it, is performed programmatically. The tool does not provide a simpler manner to specify the environment and execute the simulation. Each simulation scenario needs to be created by a developer using Java programming language.

3.2 CloudReports Graphical Tool

CloudReports [18] is a front-end to CloudSim [12]. It facilitates the process of creating cloud simulation scenarios using a graphical user interface and avoiding the need of programming. It runs on top of CloudSim, creating and executing the simulation scenarios on it.

It is a ready and easy to use tool which can be used instead of directly programming the cloud scenarios at CloudSim. Nevertheless, the tool does not allow the automation of the process of creating the whole scenario. The scenario is created graphically using the application interface. For larger environments, this can be boring and time-consuming. In addition, the tool does not allow the sharing of specifications among different research teams.

4 Proposal to Automate the Creation of CloudSim Simulation Environments

In this proposed work, cloud simulation scenarios are defined into a YAML file. YAML is the simpler human readable data serialization standard which was found in the literature and it makes easy to define a cloud simulation scenario directly, using any YAML editor or even a simple text editor. It is a non-bureaucratic data format that, different from the XML standard, does not need to close tags (keys). This simplifies and speeds up manual creation of YAML files.

In the YAML simulation scenarios, the YAML keys were defined based on the CloudReports' registries classes, as presented at Figure 1. These classes represent an abstract definition of CloudSim objects for the creation of simulation scenarios. By means of these classes, objects like datacenters, storage area networks (SANs), hosts, virtual machines (VMs), customers and applications (cloudlets) can be defined at a higher level. Using them, the amount of determined objects, like hosts, owning the same configuration, can be specified only one time. So, this speeds up the process of creating the cloud environment's objects. Thus, the creation of, for instance, 1, 2, 10, or even 100000 hosts with the same configuration has the same effort.

At Figure 1 there are the top level registries objects *datacenterRegistries* and *customerRegistries*, that represent, respectively, the abstract information of datacenters and customers in the cloud simulation scenario. These registries are a list of registry objects. The amount key, present in almost all objects which can be specified in the YAML file, defines the amount of the current object to be created during the scenario creation. The scenario is created by the proposed tool and, after this, it is executed on CloudSim. So, for instance, each datacenter in the *datacenterRegistries* list, represents a specific datacenter configuration. For each one in this list, will be created the concrete objects in the amount specified by the amount key.

The YAML snippet, presented at Listing 1.1, shows the creation of 1 datacenter with a specific configuration and the creation of 2 more datacenters that share another configuration. The first datacenter was defined with different costs of the other two ones and has VM migration disabled.

Listing 1.1: Cloud simulation environment 1: YAML sample file

```
datacenterRegistries:
  - !datacenter
    amount: 1
    allocationPolicyAlias: Simple
    vmMigration: false
    costPerSec: 0.2
    costPerMem: 0.04
    costPerStorage: 0.004
    costPerBw: 0.25
  - !datacenter
    amount: 2
    allocationPolicyAlias: Simple
    vmMigration: true
    costPerSec: 0.1
    ...
```

As can be seen in the Listing 1.2, other cloud objects can be specified. Inside each defined datacenter, a list of storage area networks (SANs) and physical hosts can be specified. Inside each defined customer, a list of virtual machines and cloudlets can be specified.

4.1 Architecture

The tool was developed with Java 7 and runs on top of CloudSim, using some CloudReports' classes presented in the Figure 1. It is a command line application, like CloudSim, that loads a specified YAML file containing the definition(s) of simulation scenario(s). It uses the YamlBeans[3], a very simple and complete YAML parser for Java that can serialize plain old java objects (POJOs) to YAML and deserialize YAML content to POJOs.

The proposed tool parses the YAML file containing the simulation scenario specification, storing the definitions into CloudReports registries objects. These registries are used to autonomously creating the concrete objects that will constitute the cloud environment (datacenters, hosts, VMs, etc). Besides, with all

[3] http://github.com/EsotericSoftware/yamlbeans

concrete objects created at the desired amount, specified at the YAML file, the tool runs the simulation in CloudSim, organizing and showing the results into appropriated tables.

At Figure 1, this process is drawn. The automation module will parse the YAML file, creating CloudReports' registries objects, and generating the Java code needed to create the cloud environment and run the simulation on CloudSim. This generated code is called "User Code" by CloudSim. After the code generation, the simulation is executed into CloudSim, presenting the results in the terminal.

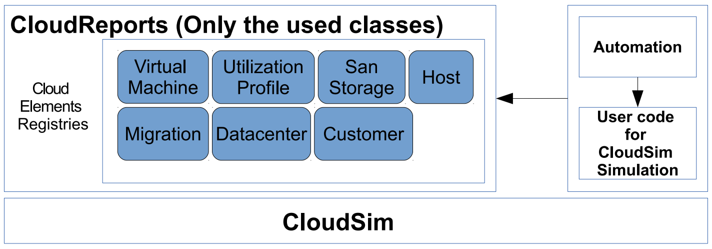

CloudReports (Only the used classes)
Cloud Elements Registries
Virtual Machine
Utilization Profile
San Storage
Host
Migration
Datacenter
Customer
Automation
User code for CloudSim Simulation
CloudSim

Fig. 1: Architecture for the creation of cloud scenarios (Adapted from [11])

4.2 Demonstration Scenario

To demonstrate the use of the proposed tool, the Listing 1.2 presents a simulation scenario defined into a YAML file. The scenario has 1 datacenter with 4 physical hosts owning the same configuration. Each host has the following configuration: 1 TB of RAM (1000000 MB), 100 Mbps of bandwidth (100000 bps), 40 TB of storage (40000 GB), and a quad-core processor (numOfPes - number of processing elements) with each core having a processing capacity of 50000 million instructions per second (MIPS). The virtual machines placed on each host will be executed using a time shared scheduler, allowing the VMs' execution to be scaled along the time.

Each host will use a simple policy on provisioning RAM (ramProvisionerAlias), bandwidth (bwProvisionerAlias) and processor's cores (peProvisionerAlias) to the virtual machines allocated on it. The simple provisioning uses the best effort algorithm to allocate the requested resources. So, if there are enough resources, they will be provisioned, otherwise, the provisioning will fail. These simple provisioners are the only available with CloudSim currently. Whether the developer wants another provisioner policies, he/she needs to implement them, extending the CloudSim classes.

On this scenario 2 customers were defined. In CloudSim they are represented by a broker. A broker is a component on the cloud that works on behalf of the service's users. So, when the users make a request to the service, the broker is responsible for negotiating with the cloud coordinator the provision of the resources needed to meet the application's QoS [12]. Each customer has 2 VMs with the following configuration: 500 MB of image size (size of the entire VM on disk), a dual-core processor (numOfPes) with each core having a processing capacity of 1000 million instructions per second, 2 GB of RAM (2000 MB) and 1 Mbps of bandwidth (1000 bps). The tasks will be executed using a space shared scheduler. This means that when an application starts, it will use the provisioned processor cores until the end of its execution. This can cause a longer wait time for the applications that depend of the busy cores.

For each customer created, 6 cloudlets to be executed were specified. The cloudlets are objects of CloudSim that represent the customer's applications in the simulation scenario. They are abstractions of the real cloud applications. So, one cloudlet only represents the computing requirements of a real application [12].

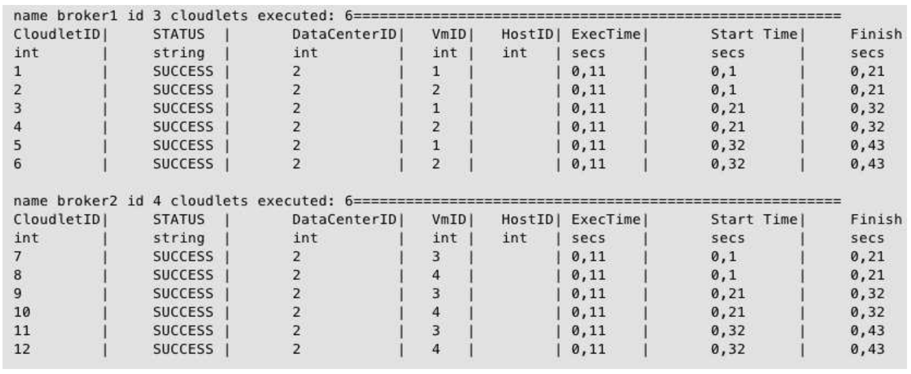
Each cloudlet was defined with the following requirements: 2 processor cores (cloudletsPesNumber) needed, 100 million of instructions (MI) to be executed, with 50 bytes of input file size (application code + input data) and 70 bytes of output file size (input file size + output data). The cloudlets were defined to use the whole required resources (RAM, bandwidth and CPU) during 100% of its execution time. This is defined by the "Full" value in the *utilizationModelXXX* keys.

Listing 1.2: Cloud simulation environment 2: YAML sample file

```yaml
datacenterRegistries:
  - !datacenter
    amount: 1
    allocationPolicyAlias: Simple
    vmMigration: enabled
    costPerSec: 0.1
    ...
    sanList:
      - !san
        capacity: 10000
        bandwidth: 10000
        networkLatency: 5
    hostList:
      - !host
        amount: 4
        ram: 1000000
        bw: 100000
        storage: 40000
        numOfPes: 4
        mipsPerPe: 50000
        schedulingPolicyAlias: TimeShared
        ramProvisionerAlias: Simple
        ...
customerRegistries:
  - !customer
    amount: 2
    vmList:
      - !vm
        amount: 10
        size: 500
        pesNumber: 4
        mips: 1000
        ram: 2000
        bw: 1000
        schedulingPolicyAlias: SpaceShared
        vmm: Xen
    utilizationProfile:
      - !profile
        numOfCloudlets: 6
        cloudletsPesNumber: 2
        length: 100
        fileSize: 50
        outputSize: 70
        utilizationModelCpuAlias: Full
        ...
```

5 Result Analysis

As can be seen in the scenario presented at Listing 1.2, there are 2 dual-core VMs for each one of the 2 customers. However, for each customer there are 6 cloudlets (applications) requiring 2 processor cores each one. Therefore, there are not enough VMs for each customer to run his/her applications at the same time. So, a space shared scheduler to the VM's tasks was defined and each cloudlet will require all the 2 cores of one customer's VM, only one cloudlet will be executed per time. By this way, the others cloudlets will wait in a queue. This can be seen at the Figure 2.



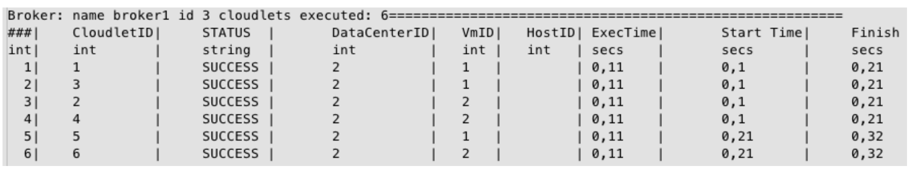
```
name broker1 id 3 cloudlets executed: 6=====================================================
CloudletID|     STATUS  |       DataCenterID|   VmID|   HostID| ExecTime|       Start Time|     Finish
int       |     string  |       int         |   int |   int   | secs    |       secs      |     secs
1         |     SUCCESS |       2           |   1   |         | 0,11    |       0,1       |     0,21
2         |     SUCCESS |       2           |   2   |         | 0,11    |       0,1       |     0,21
3         |     SUCCESS |       2           |   1   |         | 0,11    |       0,21      |     0,32
4         |     SUCCESS |       2           |   2   |         | 0,11    |       0,21      |     0,32
5         |     SUCCESS |       2           |   1   |         | 0,11    |       0,32      |     0,43
6         |     SUCCESS |       2           |   2   |         | 0,11    |       0,32      |     0,43

name broker2 id 4 cloudlets executed: 6=====================================================
CloudletID|     STATUS  |       DataCenterID|   VmID|   HostID| ExecTime|       Start Time|     Finish
int       |     string  |       int         |   int |   int   | secs    |       secs      |     secs
7         |     SUCCESS |       2           |   3   |         | 0,11    |       0,1       |     0,21
8         |     SUCCESS |       2           |   4   |         | 0,11    |       0,1       |     0,21
9         |     SUCCESS |       2           |   3   |         | 0,11    |       0,21      |     0,32
10        |     SUCCESS |       2           |   4   |         | 0,11    |       0,21      |     0,32
11        |     SUCCESS |       2           |   3   |         | 0,11    |       0,32      |     0,43
12        |     SUCCESS |       2           |   4   |         | 0,11    |       0,32      |     0,43
```

Fig. 2: Execution results of cloud simulation scenario presented at Listing 1.2

At the referred figure, it can be seen 2 brokers representing the customers defined in the simulation scenario. For each customer, 6 cloudlets were executed. The two VMs of each customer were instantiated. The VMs number 1 and 2 were assigned to the customer 1 (represented by broker 1) and the VMs number 3 and 4 were assigned to the customer 2 (represented by broker 2). At the column "ExecTime" it can be seen that all cloudlets took the same time to finish, because all cloudlets and VMs have the same configurations. However, the finish time was variable. According to the VMs and cloudlets configurations and defined restrictions, only one cloudlet can be executed at a VM per time. So, since each customer has two VMs, two customers cloudlets can be executed per time (each one into a separated VM). By this way, the first two cloudlets of each customer have the same finish time. The next two cloudlets have another finish time and the next ones too.

If the processor cores of the VMs are changed to 4, it will be possible to run two cloudlets at the same VM, so the waiting time will be reduced. This can be seen at Figure 3. The cloudlets 1, 2, 3 and 4 had the same finish time due to the fact they started at the same time. Cloudlets 1 and 3 were executed at VM 1 and cloudlets 2 and 4 were executed at VM 2. The cloudlets 5 and 6 had to wait until the end of the other ones, so their start time is exactly the finish time of the previous cloudlets. In this case, the used broker chose to allocate one customer's VM to each cloudlet, even if the two remaining cloudlets would be executed at the same VM. To change this behaviour it is needed to extend the DatacenterBroker CloudSim's class. Only the first customer results were shown because they are identical, once his/her VMs and cloudlets configurations are the same.



```
Broker: name broker1 id 3 cloudlets executed: 6=====================================================
###|    CloudletID|     STATUS  |       DataCenterID|   VmID|   HostID| ExecTime|       Start Time|     Finish
int|    int       |     string  |       int         |   int |   int   | secs    |       secs      |     secs
  1|    1         |     SUCCESS |       2           |   1   |         | 0,11    |       0,1       |     0,21
  2|    3         |     SUCCESS |       2           |   1   |         | 0,11    |       0,1       |     0,21
  3|    2         |     SUCCESS |       2           |   2   |         | 0,11    |       0,1       |     0,21
  4|    4         |     SUCCESS |       2           |   2   |         | 0,11    |       0,1       |     0,21
  5|    5         |     SUCCESS |       2           |   1   |         | 0,11    |       0,21      |     0,32
  6|    6         |     SUCCESS |       2           |   2   |         | 0,11    |       0,21      |     0,32
```

Fig. 3: Execution's results of cloud simulation scenario with quad-core VMs

The results presented were based on those ones commonly displayed by CloudSim sample applications included with the tool. They were only organized and formatted to be more readable than in the samples. As the developed tool is open source, these results can be easily adapted to the needs of the researcher, because they only summarize the information obtained from the CloudSim objects after running the simulation.

The developed tool, instead of having a limited graphical user interface, like some of the related works, it enables the researcher to use all the features available for CloudSim. This flexibility is feasible due to the fact of the runtime load of the simulation scenario, specified into a external YAML file. The tool also allows the definition of entire and complex cloud environments, without limiting the amount of elements like physical hosts, virtual machines or scheduling algorithms. Considering that it depends of CloudSim, if new scheduling algorithms are included in the tool, there is only the need to recompile the project using the new CloudSim classes to make these new scheduling approaches available to be used in YAML simulation scenarios. Even the *MapReduce* computing model can be simulated by CloudSim applications using works like [19].

Therefore, CloudSim is a very flexible tool that allows a lot of computing paradigms to be implemented and experimented on it, instead of dictating a specific one, like in some presented related works. The proposed tool running on top of CloudSim brings new flexibilities and facilities for cloud simulations.

6 Conclusion and Future Works

This paper proposed a flexible and open source solution to facilitate the process of creating cloud simulation scenarios on CloudSim tool. The tool has been shown easy to use and the specification of simulation scenarios using the human readable YAML data format speeds up this task.

The YAML defined scenarios turn the process easy to overview, understanding, extend and share. They can be created using any simple text file editor. So, the tool can reduce the learning curve in understanding cloud technologies and simulation. Further, it may motivate researches in the cloud computing area due to the easiness of creating any desired cloud simulation scenarios.

As future works it is proposed the implementation of different resource allocation, VM scheduling and placement policies; the creation of random workload; the creation of policies to define elasticity for VM resources; the creation of policies for VM replication to enable load balancing; and the integration with CloudReports graphical user interface.

Acknowledgements

This work has been partially supported by scholarship from the CAPES Brazilian agency; by the Instituto Federal de Educação, Ciência e Tecnologia do Tocantins (Federal Institute of Education, Science and Technology of Tocantins - Brazil); by Instituto de Telecomunicações, Next Generation Networks and Applications Group (NetGNA), Covilhã Delegation; and by National Funding from the FCT – Fundação para a Ciência e a Tecnologia through the Pest-OE/EEI/LA0008/2013 Project.

References

1. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, Jun. 2009. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167739X08001957
2. S. Patidar, D. Rane, and P. Jain, "A Survey Paper on Cloud Computing," *2012 Second International Conference on Advanced Computing and Communication Technologies*, pp. 394–398, Jan. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6168399
3. A. J. Younge, G. von Laszewski, L. Wang, S. Lopez-Alarcon, and W. Carithers, "Efficient resource management for Cloud computing environments," *International Conference on Green Computing*, pp. 357–364, Aug. 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5598294

4. B. Armbrust, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, and A. Rabkin, "A view of cloud computing," *Communications of the ACM*, 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1721672
5. J. Espadas, A. Molina, G. Jiménez, M. Molina, R. Ramírez, and D. Concha, "A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures," *Future Generation Computer Systems*, vol. 29, no. 1, pp. 273–286, Jan. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167739X1100210X
6. A. Goyal and S. Dadizadeh, "A survey on cloud computing," University of British Columbia, Tech. Rep. December, 2009. [Online]. Available: http://unbreakablecloud.com/wordpress/wp-content/uploads/2011/02/A-Survey-On-Cloud-Computing.pdf
7. Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, Apr. 2010. [Online]. Available: http://www.springerlink.com/index/10.1007/s13174-010-0007-6
8. Z. Zhang, C. Wu, and D. Cheung, "A survey on cloud interoperability: taxonomies, standards, and practice," *ACM SIGMETRICS Performance ...*, vol. 40, no. 4, pp. 13–22, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2479945
9. T. Dillon, C. Wu, and E. Chang, "Cloud Computing: Issues and Challenges," *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, pp. 27–33, 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5474674
10. S. S. Manvi and G. Krishna Shyam, "Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey," *Journal of Network and Computer Applications*, pp. 1–17, Oct. 2013. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804513002099
11. R. N. R. Calheiros, R. Ranjan, A. Beloglazov, and A. F. D. Rose, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, no. Issue 1, pp. 23–50, 2011. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/spe.995/full?cm=email-eng&cs=if-2012&cu=psj-13-54122&cd=psj-13-54122-compsci-specloudsim
12. R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable Cloud computing environments and the CloudSim toolkit: Challenges and opportunities," *2009 International Conference on High Performance Computing and Simulation*, pp. 1–11, Jun. 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5192685
13. W. Tian, Y. Zhao, M. Xu, Y. Zhong, and X. Sun, "A Toolkit for Modeling and Simulation of Real-Time Virtual Machine Allocation in a Cloud Data Center," *IEEE Transactions on Automation Science and Engineering*, pp. 1–9, 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6558510
14. X. Liu, X. Qiu, B. Chen, and K. Huang, "Cloud-Based Simulation: The State-of-the-Art Computer Simulation Paradigm," *2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*, pp. 71–74, Jul. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6305887
15. Q. Pan, J. Pan, and C. Wang, "Simulation in Cloud Computing Envrionment," *2013 International Conference on Service Sciences (ICSS)*, pp. 107–112, Apr.

2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6519772

16. P. Jakovits, S. N. Srirama, and I. Kromonov, "Stratus: A Distributed Computing Framework for Scientific Simulations on the Cloud," *2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems*, pp. 1053–1059, Jun. 2012. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6332290
17. S. Srirama, O. Batrashev, and E. Vainikko, "SciCloud: Scientific Computing on the Cloud," *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 579–580, 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5493427
18. T. Teixeira Sá, "CloudReports: uma ferramenta gráfica para a simulação de ambientes computacionais em nuvem baseada no framework CloudSim," *Simpósio Brasileiro de Sistemas Distribuídos e Redes de Computadores (SBRC)*, pp. 103–116, 2011. [Online]. Available: http://sbrc2011.facom.ufms.br/files/workshops/wcga/ST03_2.pdf
19. J. Jung and H. Kim, "MR-CloudSim: Designing and implementing MapReduce computing model on CloudSim," in *2012 International Conference on ICT Convergence (ICTC)*, 2012, pp. 504 – 509. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6387186