Cloud Computing-An Empirical Modelling Perspective

1060525

Abstract

The intention of this paper is going to explore the performance of the empirical perspective and tkeden toolkit with relevance to the creation of education, decision support and simulation tools in the field of cloud computing. A suitable model will be provided to represent the primary features of cloud computing and highlight several of its benefits as well as short comings. My selected weighting for this coursework is 30% paper, 70%.

1. Introduction

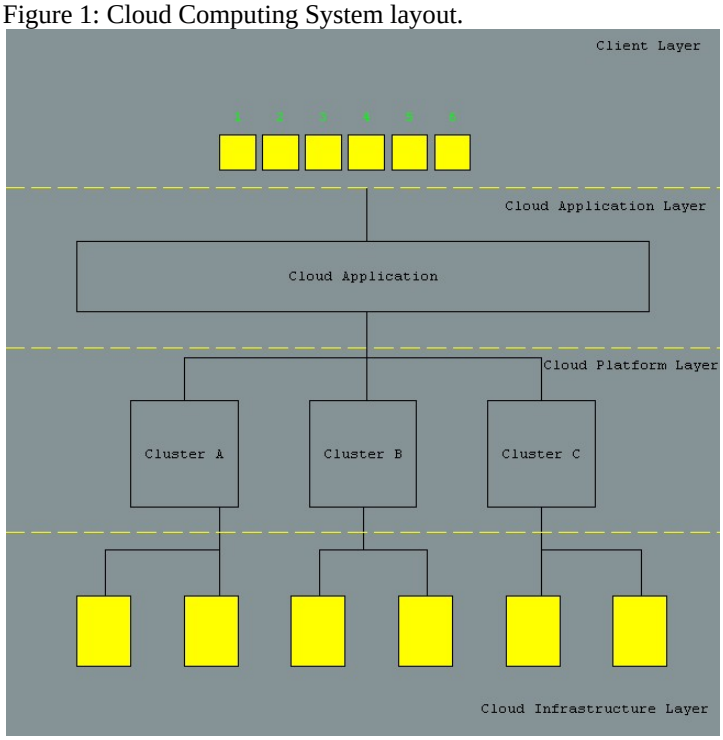
Cloud computing is the access to computers and their functionality via Internet or a local area network. This is dramatically accelerating the development of enterprises in terms of reducing large capital expenditure on hardware, software, and services. The key of cloud computing depends primarily on its architecture, monitoring and schedules. This paper is going to focus on the utility of Empirical Modelling techniques in allowing users to observe cloud computing system's performance and interact with it. I will present the architecture of cloud computing and how it uses a global monitoring to schedule user requests to allocation different situations of computer resources, such as the CPU usage, flash memory and disk stream. This model also includes functions that have login, logout, upload, delete, read data, run and close program.

2. The Modelling Study

2.1 Description

This model takes the four layer architecture (see Fig.1) to display base on Eucalyptus cloud system. A client firstly login at client layer, and then a login request is sent to Cloud Application layer by this client; the Cloud Application layer has conveyed this login request to a related Cluster, which is formed by several node controllers. The node controller virtualizes a virtual machine for this client while the client login request is arrived. This client thus login this cloud computing system and has their own machine that can upload, delete and read data and run and close programs on this virtualized machine. Moreover, there are two resource monitoring windows that are monitoring cloud system resource. One of monitoring window is for observing each login user's system resource, in terms of CPU%, Memory% and Hard-disk%, another one is for observing each cluster's system resource.

Figure 1: Cloud Computing System layout.



Furthermore, there is a user control window that is for all users to execute different applications and requests. These applications and request commands would consume different level system resource. For instance, once a user login will take 10% CPU and Memory on a virtualized machine, every time the client clicks the upload button, a 10% increment will be added on the Hard-disk%. Meanwhile, client can use "Delete" button to delete 10% data uploaded on the cloud system. Client also can "read" these data they uploaded, which needs 5% CPU and Memory. There is a "Run Program" button that is for user to run programs or applications; it will take 20% CPU

and Memory. To click the “Close” button is going to relief 10% CPU and memory taken by programs or applications. Moreover, these single user’ resource modechanges will affect correlative cluster system resource, because client 1 and 2’s virtual machines are virtualized in cluster A, client 3 and 4 are in cluster B, client 5 and 6 are in cluster C. therefore, according to User resource window and Cluster resource window, we can monitor users and system resource consume.

2.2 Observables

Observables are entities whose identity is established through experience, whose current status can be reliably captured by experiment.

User 1, 2, 3, 4, 5, 6‘s control window(see Fig 2), there are Login, Logout, Upload, Delete, Read, Program, Close buttons that are for each user executing different requests to cloud system. There are CPU%, Memory%, Hard-disk%, User and Cluster number in User and Cluster resource window (see Fig 3, 4). The rate of CPU%, Memory% and Hard-disk% will be changed and affected from 0% to 100% by different client commands and requests. Furthermore, once one of users login， a rectangle shape will display in the client layer of this model, which means a login client. In addition, another rectangle is simultaneously shown at cloud infrastructure layer, which indicates a relevant virtual machine is started on the cluster since this client login.

User	Login	Logout	Upload	Delete	Read	Program	Close
1	O	O	+	-	+	+	-
2	O	O	+	-	+	+	-
3	O	O	+	-	+	+	-
4	O	O	+	-	+	+	-
5	O	O	+	-	+	+	-
6	O	O	+	-	+	+	-

Figure 2 User Control Window.

User Resource Window

User	CPU %	Memory %	Hard-Disk %
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0

Figure 3 User Resource Window.

Cluster Resource Window

Cluster	CPU %	Memory %	Hard-Disk %
A	0	0	0
B	0	0	0
C	0	0	0

Figure 4 Cluster Resource Window.

2.3 Dependencies

A dependency is a relationship within observables, which describes how the values of these observables are affected if there is a change in others. Different parts of this model are connected together, and it is not a group of individuals but a completed system. An example of dependency in this model is that the change of system resource based on user different activities.

Each different request could bring out the change of rate resource on the User and Cluster resource windows. As mention before, these are following applications that are affecting relevant single user resource. A login request will consume 10% CPU and Memory, respectively. The “Upload” needs 10% Hard-disk capability, the “Delete” is going to decrease 10% Hard-disk data, and the “Read” increases 5% CPU and Memory, the “Program” will add 20% CPU and Memory, the “Close” is capable of reducing 10% CPU and Memory from this user resource.

Moreover, once these user system resources have been changed, the relative cluster system resource is being changed as well. For instance, after client 1 login system, there is 10% CPU and Memory increment on the User resource window. Meanwhile, a 5% CPU and Memory increment will be added on the Cluster A, as Cluster A controls client 1 and 2.

Additionally, each different request could bring out the change of resource rate on the User and Cluster resource windows. The rate of system resource on User resource window is going to change correlative cluster resource window.

Logically， all these application buttons would not have any response before the “Login” button is clicked. If one of parameters of CPU%, Memory% and Hard-disk% reaches 100%, all these applications buttons immediately stop functionalities. However, once user clicks the “Logout” button to relief all CPU% and Memory%, this client’s system resource will become the initial value except Hard-disk%.

2.4 Agency

Agents are responsible for state-changes. The example of agency in this model is each user and cluster system resource situation, which simultaneous responses to observers based on different user’s commands. These application buttons are listening users

for using. Furthermore, these rectangles of virtual machines and login clients shown on client layer and cloud infrastructure layer are agents for reflecting system changes. First of all, the rectangles of virtual machine and client are invisible in the model, after the client clicks login button, these two rectangles are shown on client layer and cloud infrastructure layer, respectively. The rectangle of virtual machine on cloud infrastructure layer will become red and warn your one of system resource is overload whilst one of CPU%, Memory% or Hard-disk% parameters is exceeded 100%. However, these two virtual machine and client rectangles will disappear and their CPU% and Memory% turns zero if this user clicks the logout button.

2.5 Model applications

This cloud computing system model represents a thoughtful perspective from administration and client level. When someone is going to learn and work with cloud computing, this model is suitable for people in any level of understanding about cloud computing, as it includes two most important issues on cloud computing: a) Virtualization; b) Schedule. User could follow these application buttons and system architecture to understand how is the cloud system working and what are the features of cloud computing system. For instance, the user can login the cloud computing system by the "Login" button, once this client login the cloud system, start a virtual machine, he can use different system applications and change this virtual machine resource by several applications.

3. Conclusion

In a short, the model has been developed for simulating the cloud computing system, but it also can educate about basic dependencies and be the trigger for more elaborate research on empiric modelling. This paper highlighted the benefits of using an empirical modelling approach to develop the model; the major features are these connected rates of system resource in this model. Each of these pentameters and applications buttons is simultaneously affecting each other. However, there are still many problems on how to display this model working. This model is dynamic creation on the virtual machines and login clients. The nature of cloud computing system is involved a series of interfaces and network transition. In this model, the algorithms are found in the changes of client and cluster resource. Therefore, tkeden was slightly inappropriate for modelling this particular feature of the cloud system. Further development of the cloud computing system includes dynamic graph description the changes of cloud computing system, such as the network interface delay. It is also a special purpose notation for students wishing to understand the dependencies in this model.

4. Acknowledgements

I would like to acknowledge the priceless and invaluable advices about stimulating this cloud computing system from Meurig Beynon, Yu Guan, and Yi Yao throughout this project.

5. References

[1] The Eucalyptus Open-source Cloud-computing System**,** Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitrii Zagorodnov, in Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid, Shanghai, China.

[2] Eucalyptus: A Technical Report on an Elastic Utility Computing Architecture Linking Your Programs to Useful Systems**,** Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitrii Zagorodnov, UCSB Computer Science Technical Report Number 2008-10 (August 2008)

[3] Meurig Beynon. Definitive notations for interaction. Proc. HCI'85, pages 23–34, 1985