ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 5, November 2014

Towards an Adaptive QoS of Cloud-based Web Services

Mohamed-K HUSSEIN
Tabuk University, Saudia Arabia
Faculty of Computers and Informatics, Suez Canal University, Egypt

Abstract*— Service oriented applications define the Quality of Services (QoSs) in terms of a Service Level Agreement (SLA) which guaranties the non-functional attributes for the users. However, web services usually have unpredictable load conditions and fluctuating demands which makes it difficult to maintain the predefined SLA. For example, extreme access of service-based application, such as e-commerce or news web services connected to a database. It is challenging to provide the expected QoS attributes, such as the response time. Cloud computing has gained a huge interest in academia and major IT organizations. The ability to create several virtual machines on each physical machine (PM), and to resize the virtual machine resource configuration up and down as needed, gives the cloud the ability to guarantees the SLA for the service-based applications. This paper presents a framework for adaptive QoSs of web services in a cloud environment. In case of predictions of the SLA violations, a dynamic resource scaling is managed based on two level. The first level, resources such as CPUs, memory of the virtual machine, is scaled up. The second level, a new virtual machine is instantiated on another physical machine to process the increasing requests using replication. The framework monitors the requests arrival rate and the response times of the web service, and uses a time series to predict the future arrival rates and response times. The process of scaling the resources and virtual machines up and down is based a queuing network model. An experimental analysis is conducted to explore the feasibility of the proposed framework on a private cloud, using Eucalyptus, and a synthetic workload.

***Index Terms*—Cloud Computing, Web services, Quality of Service,**

I. INTRODUCTION

Service Oriented Computing architecture and web services provide flexible infrastructure for constructing distributed web applications in industry and academia, such as e-commerce and scientific workflows applications [1]. Web services are standard loosely coupled software component which provide business functionality published over the internet [2, 3]. Web services are published, described, discovered and executed using standard protocols, including WSDL, SOAP and UDDI [4]. Cloud computing has gained its popularity by the ultimate advancement of the virtualization technology, which allows multiple virtual machines (VMs) to share the same physical machine (PM) [5-7]. Currently, cloud computing is the primary distributed computing environment. Therefore, major IT organizations are publishing their web services in cloud infrastructures [7]. Infrastructure as a Service (IaaS) cloud provides computational and software resources for deploying operating systems and service-based application [8]. The IaaS architecture allows the user to provision the computational resources, called elastic provisioning, to meet the changing workload demand [9, 10]. Elastic resource provisioning is an important feature of the cloud which can be used to ensure the Service-Level agreement (SLA). SLA is the predefined expected quality of service (QoS) attributes, such as the response time and availability, between the service provider and the service consumer [4]. This key feature of the cloud makes it possible for service providers to handle the unpredictable and extreme loads conditions, such as large-scale access to news or e-commerce web services.

Elastic resource provisioning can be done on two levels.

1. The first level is at the resource level where computational resources, such as CPUs and memory, are scaled up and down to the virtual machine which host the web service. These extra resources are scaled up and down adaptively as the workload of the web service increases or decreases.
2. The second level is at the architecture level where the numbers of the virtual machines are scaled up and down. In case of extreme load, the web service is replicated over a new virtual machine in order to guarantee the response time within the defined SLA level. Further, the virtual machine which has no load can be dynamically removed to free the unneeded resources.

Service based applications usually suffer unpredictable and fluctuating load conditions. For example, e-commerce may have increasing access during a release of certain offer, or news web services which may release important news [9]. To this end, dynamic resource provisioning of the cloud is an important feature which adaptively adjust the response time of the web service to guarantee the expected SLAs according to the changing load conditions of the web service. This paper presents a framework for adaptive provisioning of the cloud resources in order to guarantee the predefined QoS attributes, such as response time and availability, of a web service defined in an SLA. The architecture monitors the access rates and the response times. In case of predictions of violation of the QoSs, using time series, the adaptive strategy scale up the resources to the virtual machine hosting the web service. In extreme load conditions, the framework instantiates dynamically extra virtual machine and replicates the web service on new virtual machines. Decisions of the scaling up

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 5, November 2014

and down are based on a queuing network model. The remainder of this paper is organized as follows. Section 2 presents the related work on cloud computing and experiences in resource provisioning for web service-based applications in the cloud. Section 3 presents the proposed adaptive framework for dynamic resource provisioning. Section 4 describes the experimental setup, and the experimental results as well as the performance analysis. Finally, conclusions and future work are given in Section 5.

II. RELATED WORK

Adaptive management of QoSs of web services has been long studied in traditional distributed environments, such as clusters [11]. Replication technology is the major technique for supporting the QoSs requirements of a web service [12, 13]. This is called application scaling where multiple replicas of a web service are provided on different physical machines to handle the increasing load conditions on a web service. For example, in [4] a queuing model and time series are used to provide short-term predictions of the load conditions of a particular web service on a cluster-based environment. Whenever SLA violation is predicted, extra replica of the web service is instantiated on a different compute node of the cluster. With the advent of the cloud computing, a number of researches have been conducted to provide adaptive QoSs management of web service in a cloud-based environments. For example, in [14], a web service scaling is conducted where another virtual machine is instantiated along with a web service replica after the SLA violation is detected. However, no predictions are provided in this study. Further, instantiating a virtual machine at runtime incur overhead cost which can be avoided if only resource-level scaling along with long-term predictions are used. In [15, 16], resource–level scaling is conducted along using different prediction mechanisms, such as regression and Markov model. In [9], resource-level and application-level scaling are provided to handle the SLA violations. However, this research do not rely on predictions of the SLA violations. Further, the experiments were conducted on web-based application using servlet. In [17], a framework for adaptive management of service based application on multiple clouds. The framework handles the adaptation by monitoring the cloud resources and redeploying the components of the application over multiple resources, on multiple clouds, which satisfy the changing requirements.

III. THE PROPOSED FRAMEWORK DESIGN

This section presents the core contribution of the study. A detailed description of a framework for the adaptive management of QoSs of a web service in a private cloud environment. It starts with a detailed description of the predictions mechanisms, and the queuing network model used in the proposed framework. Finally, the proposed scaling algorithm is presented.

A. Double Exponential Smoothing

Double exponential smoothing (DES) is a computationally cheap smoothing and forecasting time series technique. For any sequence of observations, called time series, the double exponential smoothing is capable of providing short term prediction based on the trend which exist over the time series [18]. Suppose x_t is a time series value at time t. The short term forecast for the m steps ahead is as follows:

$$x_{t+m} = L_t + b_t \times m \quad (1)$$

where L_t and b_t are exponentially smoothed estimates of the level and linear trend of the series at time t:

$$L_t = \alpha\, x_t + (1 - \alpha)(L_{t-1} + b_{t-1}) \quad (2)$$

where α is a smoothing parameter, $0 < \alpha < 1$,

$$b_t = \beta\,(L_t - L_{t-1}) + (1 - \beta) b_{t-1} \quad (3)$$

where β is a trend coefficient, $0 < \beta < 1$.

A large value of α adds more weight to recent values rather than previous values in order to predict the next value. A large value of β adds more weight to the changes in the level than the previous trend. Initial estimates for L_t and b_t are as follows: $L_1 = x_1$ and $b_1 = 0$. Service rate and response time are two critical parameters for the proposed adaptive QoS of cloud-based Web services architecture. By monitoring these two parameters, they are considered two time series. Hence, double exponential smoothing can be applied to provide short term predictions for the service rate and the response time. Based on these predictions, a performance model using queuing network to accurately characterize the performance characteristics of the system under the predicted workload conditions.

B. Queuing Network Model

Queuing networks (QN) is well used in the literature to provide an analytical performance model of a system. It is capable of accurately characterizing and predicting the response time of a system under various load conditions [19]. Hence, control decisions can be made. A similar queuing network model has shown its effectiveness in providing an analytical performance model of a web service [4]. The QN model is based on two parameters, namely requests load λ_i and response time R_i for the web service on the machine i which handle the requests. The average response time R is calculated in QN model upon the utilization of each physical host (U_i) and the response time, (R_i), on the machine i using the following Equation:

$$R = \sum_{i=1}^{k} \frac{R_i}{1 - U_i} \quad (4)$$

K is the total number of machines which have replicas for the given web service. The total utilization of a machine i, U_i, is a product of arrival rates and service demands is calculated using the following Equation:

$$U_i = \lambda_i * D_i \quad (5)$$

C. The Proposed Framework

The proposed framework consists of two main services, the monitoring service and the controller service, as shown in Fig 1. The monitoring service monitors the service demand and the arrival rate, every 60 seconds, for each web service replica on each virtual machine instance i.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 5, November 2014

The collected data are reported to the main monitoring service on the node controller of the cloud. The main monitoring service apply the double exponential smoothing, using Equation 1, to predict the next 180 seconds of service demand and arrival rate for each virtual machine. The controller service guarantees the required response time in the defined SLA. Once the controller receives a service request, it applies the algorithm shown in Fig 2 in order to: (1) load balance the requests between the available virtual machines instances, (2) forward the request to a virtual machine where it guarantee the define SLA, (3) scale the resource configuration of the virtual machine, and the number of the virtual machines, up and down according to the changing load conditions. As described later in the experimental analysis, the private cloud will be installed using Eucalyptus. On each physical machine, only one virtual machine will be instantiated. Each virtual machine instantiated will host a web service replica and its monitoring and predictor service. The controller service and the monitoring service will be set on a virtual machine on the node controller of the cloud.

The main algorithm of the controller and dispatcher assumes that the upper limit R^u and lower limit R^l of the web service response time is defined in the SLA. If the predicted response time, calculated using equation 4, is greater than the upper limit, the algorithm scale the resources up. If the predicted response time is below the lower limit, the algorithm scales the resource down. Finally, the algorithm forward the request to the virtual machine instance with the least utilization, U_i, calculated using equation 5,in order to ensure the load balancing between the resources The basic idea behind the resource scaling up algorithm, shown in Fig 2, is to increase the resource, such as available CPUs and memory. The resources are increased for all instantiated virtual machines.

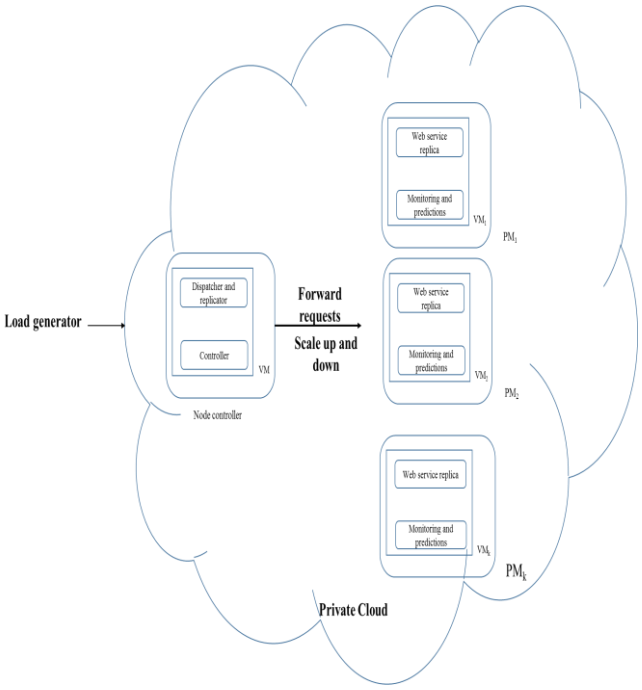

Fig 3. The architecture of QoS adaptation of web service on a private cloud.

—The controller receives a request.
—Predict the overall response time, R, for the web service using Equation 4.
—If $R > R^u$ then
- Apply the resource scaling up algorithm

—If $R<R^l$
- Apply the resource scaling down algorithm

—Forward the request to the virtual machine with least U_i using Equation 5.

Fig 1. The main algorithm of the controller and dispatcher.

— resource_scaled = false
— for each instantiated virtual machine vm_i has utilization $U_i > 0.9$
- if the physical machine for vm_i has available resources then
  - add 2 CPUs and 8 Gb memory to vm_i
  - resource_scaled = true

— if the average utilization, U, for all virtual machines > 0.9 and resource_scaled = false
- instantiate a new virtual machine with a new replica on another physical machine
- number of instantiated virtual machine = k +1

Fig 2. Resource scaling up algorithm.

This level of scaling is done in order to adaptively enhance the performance of the bottleneck virtual machines without instantiating a new virtual machine. The bottleneck virtual machine can be determined if it has a utilization, U_i, greater than a specified threshold 0.9. If average utilization of all virtual machine is greater than 0.9 and no more resources to scale the virtual machines, in this case a new virtual machine on a physical server is instantiated. Fig 2 shows the detailed algorithm for the resource scaling algorithm.

The main goal of the resource scaling down algorithm is to free the cloud resources for other services competing for resources instead of reserving resources that are no longer required. The algorithm reduces resources from virtual machines with lowest utilization. The second part, the algorithm remove the virtual machines which do not affect the overall performance of the web service. The algorithm, as shown in Fig 4, starts by calculating the average utilization of the instantiated virtual machines. The condition set for starting the process of scaling down is that U_i must be less than 0.4. If that condition holds, it means that the virtual machines are holding resources more than necessary. In this case, the algorithm selects the virtual machine with the lowest utilization, and frees half of the held resources. Further, this virtual machine is marked for removal in order to not to assign

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 5, November 2014

any requests. Finally, the algorithm removes the virtual machines which has zero request rate.

—if the average utilsation, U, of all virtual machines < 0.4 using Equation 5.
- free half CPUs and memory assigned to the vm_i with the lowest utilsation U_i
- set vm_removal = true

—for all vms
- if vm_remval = true AND $\lambda_i = 0$
  - remove the virtual machine

Fig 4. The resource scaling down algorithm.

IV. EXPERIMENTAL EVALUATION

In order to evaluate the applicability and feasibility of the proposed approach, a private cloud was set using Eucalyptus which is open source software for Infrastructure as a service (IaaS) cloud. The main advantage of Eucalyptus is that it is compatibility with commercial cloud products such as Amazon EC2 and S3 [20]. This compatibility enables to run a web services on a private cloud using Eucalyptus and a public cloud using Amazon without modification in execution framework or the application. Eucalyptus architecture consists of a number of components, namely Cloud Controller, Node Controller, Cluster Controller and Storage Controller. The Cloud Controller (CLC) is responsible for managing the available virtual resources, such as Servers, network and storage. The Node Controller (NC) runs on each physical machine (PM), and controls the available virtual machines. The Cluster Controller (CC) collects information on the installed virtual machines and schedules the VMs for execution on the NC. For data storage service, the Storage Controller (SC) which manages storage block volumes by communicating between NC and CC [21]. On each physical machine Xen is used as a VMM hypervisor to create the virtual machine of a desired configuration. Xen is a popular open source software for VMM which is used to obtain high performance of multi-cores physical machines [22]. The experiments are conducted using four physical machines; each machine has i7 core Intel 2.2 GHz processor and 32GB memory. The virtualization layer is based on Xen hypervisor version 4.3. The VMs are deployed using Eucalyptus version 4. Each node runs Ubuntu version 12 operating system. 1 Gigabit Ethernet network fabric is used for networking. On three compute node, a virtual machine is deployed and initially assigned 2 CPUs and 8 GB memory. The fourth physical machine is set as the cloud controller. The proposed framework is installed on the cloud node controller, as shown in Table 1.

Table 1. Deployment and initial resource configuration.

PM No.	VM No.	Initial resource mapping	Mapping
1	1	2 CPUs and 8 GB memory	replica
2	2	2 CPUs and 8 GB memory	replica
3	3	2 CPUs and 8 GB memory	replica
4	4	2 CPUs and 8 GB memory	Dispatcher and controller

A benchmark web service connected to a database server to execute a query is implemented. The web service is a search service which returns the cost of air flight tickets from a specific place to a specific destination. It executes an SQL query to obtain the lowest 10 prices of 1500 records. The lower and upper bound for the response time is set for 2 and 3 seconds.

Apache Benchmark [23] was used to generate the required load requests to evaluate the proposed framework. The Apache Benchmark has the ability to configure the number of requests and the time spent on the test. Fig 5 shows the amount of generated requests for 50 minutes. The load keeps increasing every 5 minutes, then keeps decreasing after 35 minutes.

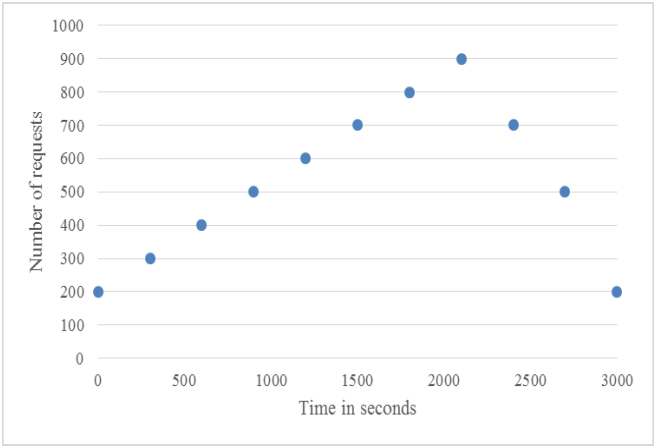

Fig 6. The generated load for 30 minutes.

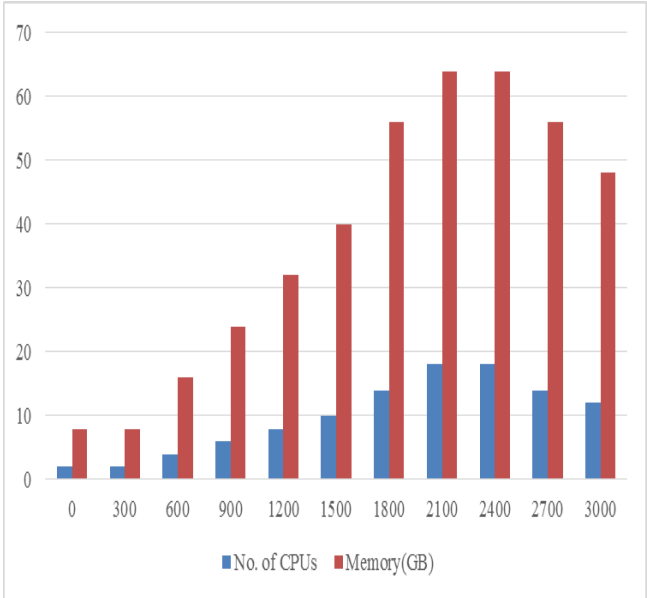

Fig 5. The amount of resources scaled up and down for 50 minutes.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 5, November 2014

Fig 6 shows the total number of CPUs and the amount of memory configuration as the load varies. Fig 7 shows the corresponding number of virtual machines instantiated during the various load conditions. The framework keeps scaling the resources up for the virtual machine as the load increases. After 1200 seconds, the framework successfully instantiated a new virtual machine, and keeps scaling the resources of the second virtual machine as the load increases. After 1800 seconds, the framework instantiated the third virtual machine, and keeps scaling its resource as well to face the increasing load requests. As the request load decrease after 2400 seconds, the framework decreases the resources until a virtual machine is removed from the configuration after 2700 seconds. Fig 8 shows the corresponding response time of the web service. The proposed framework successfully managed to keep the response time with in the defined SLA upper and lower bound.

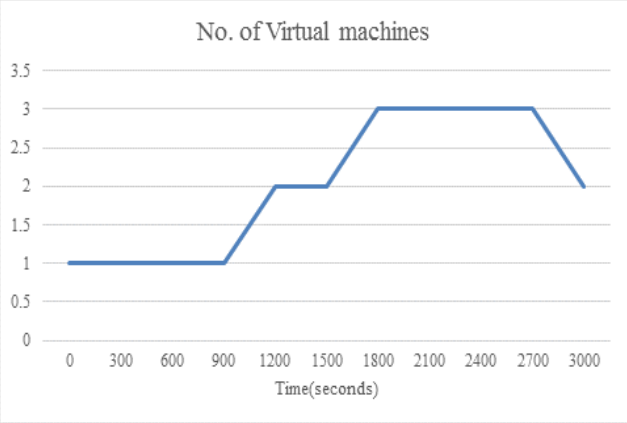

Fig 8. The number of virtual machines scaled up and down.

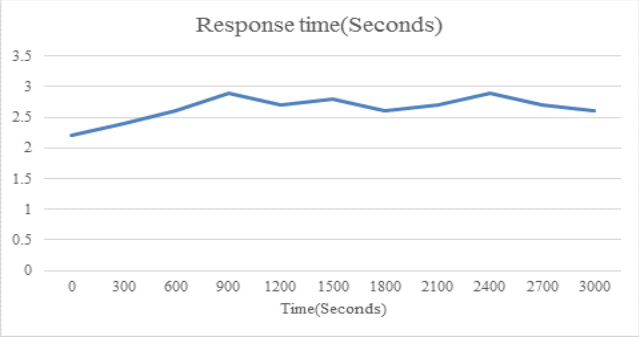

Fig 7. The corresponding response time.

V. CONCLUSION

This paper has proposed an adaptive framework for executing a web service-based application in a cloud environment. The framework scale the virtual machine and its resource configuration up and down according to the varying requests load. Further, the framework makes the scaling decisions based on predications of the load requests and the response time using double exponential smoothing time series. Also, a queuing network model is used to characterize the performance impact of the system under the predicted load requests. Hence, the scaling decisions are made according to the provided predictions. The experimental evaluation has shown the effectiveness of the proposed framework using a search web service benchmark under different scenarios of synthetic load conditions. This research is an initial design for a cloud-based adaptive performance framework of service oriented applications. There are many ways must be considered in future work of this research. First of all, the proposed framework only considered a single web service application for the sake of simplicity of the study. The future work will enhance the framework in order to handle a complex service oriented application that consists of a number of web services compete for resources with each other. Further, this research only considered a single application. The future research may consider investigating the possibility of multiple applications run at the same time for a single user with a single SLA defined. In this case, a heuristic is needed to coordinate and scale resources to meet the QoSs of all applications at the same time.

REFERENCES

[1] Y. Wei and M. B. Blake, "Adaptive Web Services Monitoring in Cloud Environments," Int. J. Web Portals, vol. 5, pp. 15-27, 2013.

[2] M. P. Papazoglou, Ed., Web Services: Principles & Technology. Pearson Education, 2008, p.^pp. Pages.

[3] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, "Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI," IEEE Internet Computing, vol. 6, pp. 86-93, 2002.

[4] M.-K. HUSSEIN and M.-H. MOUSA, "A Framework for Adaptive QoS of Web Services using Replication," International Journal of Computer Science & Communication Networks, vol. 2, pp. 288 - 294, 2012.

[5] M.-K. HUSSEIN and M.-H. MOUSA, "High-performance Execution of Scientific Multi-Physics Coupled Applications in a Private Cloud," International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, pp. 11-16, 2014.

[6] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, et al., "A view of cloud computing," Commun. ACM, vol. 53, pp. 50-58, 2010.

[7] 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 Gener. Comput. Syst., vol. 25, pp. 599-616, 2009.

[8] A. Raveendran, T. Bicer, and G. Agrawal, "A Framework for Elastic Execution of Existing MPI Programs," in Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on, 2011, pp. 940-947.

[9] R. Han, L. Guo, M. M. Ghanem, and Y. Guo, "Lightweight Resource Scaling for Cloud Applications," presented at the Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012), 2012.

[10] G. Galante and L. C. E. de Bona, "A Survey on Cloud Computing Elasticity," in Utility and Cloud Computing

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 5, November 2014

(UCC), 2012 IEEE Fifth International Conference on, 2012, pp. 263-270.

[11] M. Keidl, S. Seltzsam, and A. Kemper, "Reliable Web Service Execution and Deployment in Dynamic Environments. Technologies for E-Services." vol. 2819, B. Benatallah and M.-C. Shan, Eds., ed: Springer Berlin / Heidelberg, 2003, pp. 104-118.

[12] S. S. Yau, G. Goyal, and Y. Yao, "Replication for Adaptive Responsiveness in Service-Oriented Systems," presented at the Proceedings of the Fifth International Conference on Quality Software, 2005.

[13] X. Ye and Y. Shen, "A Middleware for Replicated Web Services," presented at the Proceedings of the IEEE International Conference on Web Services, 2005.

[14] W. Iqbal, M. Dailey, and D. Carrera, "SLA-Driven Adaptive Resource Management for Web Applications on a Heterogeneous Compute Cloud," in Cloud Computing. vol. 5931, M. Jaatun, G. Zhao, and C. Rong, Eds., ed: Springer Berlin Heidelberg, 2009, pp. 243-253.

[15] G. Zhenhuan, G. Xiaohui, and J. Wilkes, "PRESS: PRedictive Elastic ReSource Scaling for cloud systems," in Network and Service Management (CNSM), 2010 International Conference on, 2010, pp. 9-16.

[16] Z. Shen, S. Subbiah, X. Gu, and J. Wilkes, "CloudScale: elastic resource scaling for multi-tenant cloud systems," presented at the Proceedings of the 2nd ACM Symposium on Cloud Computing, Cascais, Portugal, 2011.

[17] A. Brogi, J. Carrasco, J. Cubo, F. D'Andria, A. Ibrahim, E. Pimentel, et al., "SeaClouds: Adaptive Management of Service-Based Applications Across Multiple Clouds," presented at the Proceedings of the 4th International Conference on Cloud Computing and Services Science (CLOSER 2014), Spain, 2014.

[18] S. G. Makridakis, S. C. Wheelwright, and R. J. Hyndman, Eds., Forecasting: Methods and Applications, 3rd Edition. 1998, p.^pp. Pages.

[19] D. Peng, Y. Yuan, K. Yue, X. Wang, and A. Zhou, "Capacity Planning for Composite Web Services Using Queueing Network-Based Models. Advances in Web-Age Information Management." vol. 3129, Q. Li, G. Wang, and L. Feng, Eds., ed: Springer Berlin / Heidelberg, 2004, pp. 439-448.

[20] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, et al., "The Eucalyptus Open-Source Cloud-Computing System," presented at the Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2009.

[21] E. Kijsipongse and S. Vannarat, "Autonomic resource provisioning in rocks clusters using Eucalyptus cloud computing," presented at the Proceedings of the International Conference on Management of Emergent Digital EcoSystems, Bangkok, Thailand, 2010.

[22] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, et al., "Xen and the art of virtualization," SIGOPS Oper. Syst. Rev., vol. 37, pp. 164-177, 2003.

[23] Apache benchmark. Available: http://httpd.apache.org/docs/programs/ab.html

AUTHOR'S PROFILE

Mohamed K Hussein is an assistant professor in the Computer Science department, Tabuk University and Suez Canal University. Mohamed has experience in teaching and research in computer science topics. Especially in the field of parallel and distributed applications, Service Oriented Computing and Cloud Computing.