SERVICE ORIENTED CLOUD ARCHITECTURE FOR IMPROVED PERFORMANCE OF SMART GRID APPLICATIONS

Rajeev T.[1], Ashok S.[2]

[1] *Research Scholar,* [2]*Professor, Department of Electrical Engineering, National Institute of Technology, Calicut, Kerala, India,* ***mail2rajeevt@gmail.com***, ***ashoks@nitc.ac.in***

Abstract

An effective and flexible computational platform is needed for the data coordination and processing associated with real time operational and application services in smart grid. A server environment where multiple applications are hosted by a common pool of virtualized server resources demands an open source structure for ensuring operational flexibility. In this paper, open source architecture is proposed for real time services which involve data coordination and processing. The architecture enables secure and reliable exchange of information and transactions with users over the internet to support various services. Prioritizing the applications based on complexity enhances efficiency of resource allocation in such situations. A priority based scheduling algorithm is proposed in the work for application level performance management in the structure. Analytical model based on queuing theory is developed for evaluating the performance of the test bed. The implementation is done using open stack cloud and the test results show a significant gain of 8% with the algorithm.

Index Terms: *Service Oriented Architecture, Smart grid, Mean response time, Open stack, Queuing model*

----------***----------

1. INTRODUCTION

Smart grid is a complex network involving large number of energy sources, controlling devices and load centers. Now the focus is on the development of a dynamic smart grid management platform for offering various services. An interconnected smart grid with large number of dispersed renewable energy sources, its associated measuring and control functionalities require large data storage. The existing centralized approach is not effective in such situation with huge data storage and computational needs. The new infrastructure to replace the existing one should address the future data storage and computational needs. An efficient smooth information exchange for monitoring and control of widely distributed power sources is also needed. The immense potential of cloud computing technology can be utilized to address these issues. The sharing of resources in various substations reduces the cost of operation, improves the performance of utility and offers environmental friendly smart grids. The cloud environment provides a flexible way of building, facilitating computing and storage infrastructures for varying on line and offline services. A flexible and upgradable cloud computing architecture for application deployment offers efficient application sharing over the internet.

Modern power system is structured with distributed energy resources which are required to deal with large amount of data and information systems [1]-[2]. The storage and processor resources become increasingly higher with the integration of renewable sources to the existing grid [3]. Cloud computing in large power grid and cloud data service center are considered as one of the central options which can integrate current infrastructure resources of the enterprise like hardware, high-performance distributed computing and data platform. The recent work [4] presented a cloud computing model for managing the real time streams of smart grid data for the near real time information retrieval needs of the different energy market actors. The approaches in [4]-[5] considered the model of ubiquitous data storage and data access of the smart grid data cloud, focusing on the characteristics of the underlying cloud computing techniques. Architecture for data storage, resource allocation and power management and control is presented in [6]. The paper discusses existing issues and necessity of a cloud computing architecture for power management of micro grids.

Efficient utilization of resources is important in cloud computing and for that scheduling plays a vital role to get maximum benefit from resources [7]-[10]. Wei-Tek Tsai et al. (2010) illustrated the Service-Oriented cloud computing architecture. Cloud computing is getting popular and IT giants such as Google, Amazon, Microsoft, IBM have started their cloud computing infrastructure. The paper gives an overview survey of current cloud computing architectures and discusses the existing issues of current cloud computing implementation. They presented a Service-Oriented Cloud Computing Architecture (SOCCA), so that clouds can interoperate with each other. Furthermore, the SOCCA also proposes high level designs to support multi-tenancy feature of cloud computing.

In line with the above, the paper presents general service oriented architecture for real time services in a distributed structure. The architecture for real time services generally needs to incorporate various updates in the service level as well as at the architectural level. Hence an open source structure is preferred for the implementation set up. A priority scheduling algorithm is presented in this paper for performance management. Analytical model based on queuing theory to capture the performance of the above structure is also developed for evaluation.

2. SERVICE ORIENTED ARCHITECTURE FOR REAL TIME SERVICE

Cloud computing is emerging with rapidly growing service oriented architectures for real time services. Service oriented architectures for various online services require a dynamic adaptable infrastructure for sharing data, compute and transaction services across applications. The system architecture proposed in the paper enables secure and reliable movement of information and transactions with the internal resources, also with users over the internet to support various services. The middleware provides the necessary functionality required to build and deploy fully operational application services. It includes resource management, data management and portability. The logical service model is shown in Figure1.The infrastructure as a service provides virtualized data storage and computing power. The smart grid applications in the cloud involves on line and offline computations, monitoring and analysis of large data, including online measurement data. The proposed architecture takes full advantage of functionality provided by open stack and ensures real time operations in a widely distributed environment. Operating system requires hypervisor for creation and termination of instances.

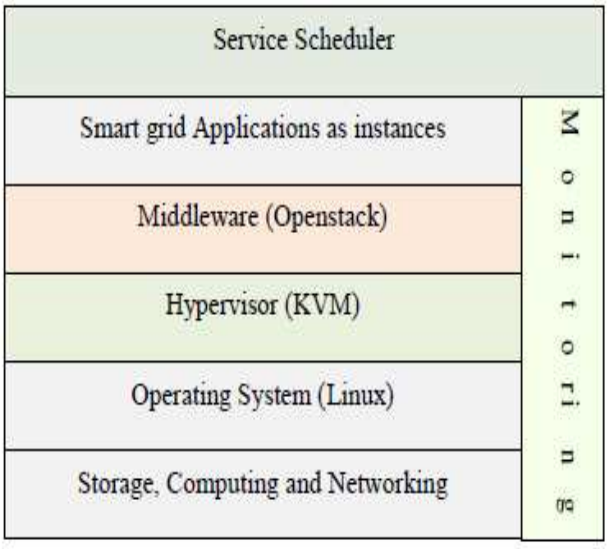

Fig -1: Logical View of Cloud Test bed for Smart grid

The process for selecting a hypervisor means priority and making decisions based on resource constraints, numerous supported features and required technical specifications. Kernel Virtual Machine is selected as hypervisor in the architecture and there is flexibility for selecting multiple hypervisors for different zones. Different algorithms and software associated with execution of the on line pricing and data analytical operations are deployed as instances in the smart grid application layer. The service scheduler is realized through open stack nova, allocates the request according to the service priority.

Each virtual machine shares resources on a physical server, including CPU capacity, disk access bandwidth and network I/O bandwidth. Hence the general terminology resource is used throughout this paper to represent all the above shared parameter together.

2.1 Implementation

The details of specifications of the test bed are shown in Table 1. Laboratory test bed has been set up for realizing the open source cloud computing environment for various data intensive and computational intensive applications in real time mode. The performance of the architecture for different test cases was recorded using web stress tool. The architecture for meeting the above goal is depicted in Figure 2. The components of open stack [12] configured for services include nova-API nova-compute, nova-scheduler, nova-volume and nova-network. Nova-API initiates most of the activities such as running an instance and provides an endpoint for all API queries. Nova-scheduler process take a virtual machine instance request from the queue and determines where it should run. The creation and termination of virtual machine instance is controlled by nova-compute process. It accepts actions from the queue and then performs a series of system commands like launching a Kernel Virtual Machine (KVM) instance to carry out while updating state in the database.

Table- 1: Recommended Hardware/Software

Item	Recommended Hardware/Software
Cloud Controller	HpProliant,64bit x86,2048KCache,12 GB RAM,2x1TBHDD,1GB NIC
Client Node	Intel Pentium 4, 3.20 GHz; 2048K Cache,8GB RAM, 1TB HDD,2X1GB NIC
Operating System	Ubuntu12.04
Middleware	Open stack
Monitoring	Web Stress Tool

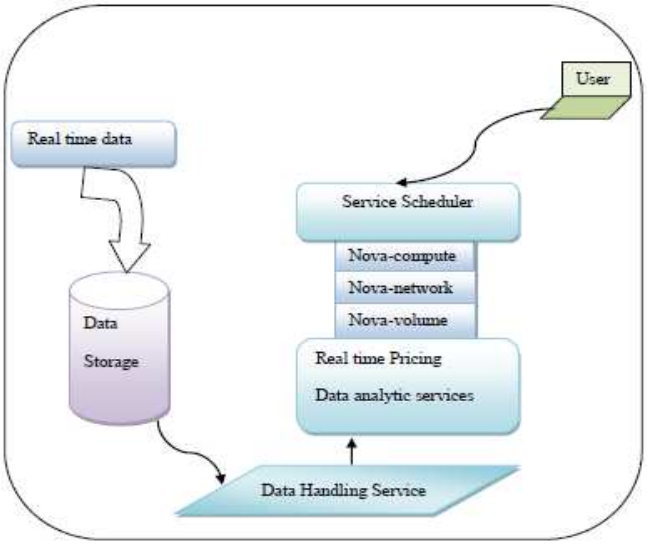

Fig -2: Architectural frame work for Application management

The nova-volume manages the creation, attaching and detaching of persistent volumes to compute instances. Nova-network accepts networking tasks from the queue and then performs tasks to manipulate the network. Data base for storing generation and consumption profile, the data handling service for aggregating data from various locations, were created as instances in the framework.

The architecture proposed here is for the operation of various application services related to smart grid. The test case considers application services relating to power system with distributed generation to simulate the interaction between the generating sources, loads and consumers. The data related to various services were stored in the database created as multiple instances. The aspect of real time data manipulation was executed through software programs. The data handling service created as another instance, manages the data traffic. The case considered for testing involves N consumers and M number of generating sources. Each production/consumption update is forwarded to the multiple instances using data handling service. Though it is an approximation of data sending and reception by smart meters in the utility and consumers, it contains all the required components and price dynamics that are likely to be present in the future smart grid.

Real time pricing and data analytic services were deployed in different virtual machines. Pricing calculation service involve several algorithmic steps to reach its decision where as simple data retrieval and processings are involved in the other category. To avoid traffic conjunction, the applications which require huge computing will be given lower priority. In the architecture, all the request which is coming from client nodes are goes through the service scheduler. It is preloaded with the priority status of various services. Here the scheduler simply allocates to the lightly loaded virtual CPU during normal situations and executes priority inversion, when admissible delay crosses the point. Python script has been used to implement the algorithm.

The timely data retrieval and computational requirements associated with the execution of on line enquiry from user perspective was tested for different values of request rate. It has been found that the response time requirement for real user interactive services is very demanding. Moreover, in virtualized environments resource allocation actuation can be done inside the server as well, using interfaces available from virtual machine monitors or hypervisors. Response time and CPU consumption are used are reference parameters in such operations. Hence in each experiment, time series of response times and CPU consumption were collected and analyzed. The performance results for 50,000 consumers requesting for services at a request rate of 30/s is shown in Figure 3and Figure 4, where actual CPU resources consumed by virtual machines are represented as CPU consumption. The rate of request considered here is comparable with the average request level for above category of services in smart grid environment with such number of consumers. The mean response time varied between 35ms-45ms. The average value of CPU consumption was about 50%-60%. These values are reasonable considering the smaller test bed and the complexity of applications used in the test. To have a better understanding of the characteristics of the system for massive hit rate and higher utilization of resources, another set of tests were conducted by simulating cases with 100% incease in the resource demand with same resource level. Figure 5 shows the performance change in the test bed for higher utilization of resources. Here, the percentage change Mean Response Time (MRT) is defined as the percentage decrease in response time for a given resource level and 100% increase in demand. High utilization of the available resources while keeping the good response is one of the advantages of the proposed algorithm. The consistency in the performance was tested by conducting experiments in the test bed with different cases. Table 2 depicts percentage of failed hits for different degree of concurrent request. Tests were conducted by applying linearly increasing request level for various numbers of users considered. It has been found that the algorithmic approach maintains good success rate for varying level of request rate.

Table- 2: Statistics of failed hits

Number of users	%Error(Failed Hits)	
	With Priority Algorithm	Without Algorithm
1000	0.5	0.6
5000	1.21	2.1
10000	5.1	7.2
15000	7.2	11.3
20000	7.9	12.1
25000	8.4	19.5
30000	11.5	21.8

35000	12.5	23.7
40000	14.4	25.8
45000	15.7	26.9
50000	17.5	31.8

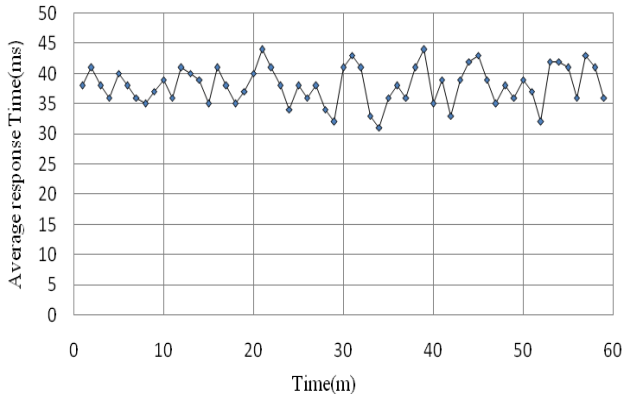

Fig -3: Mean response Time

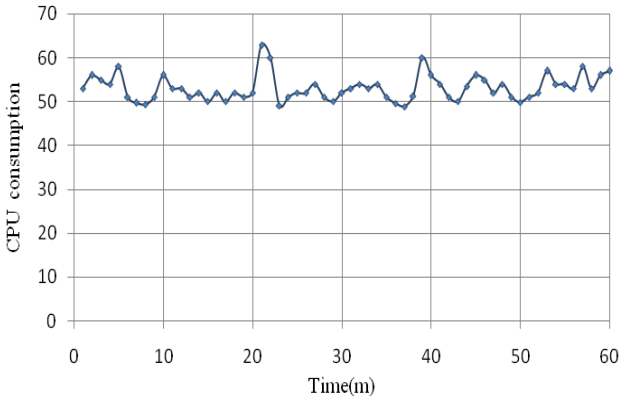

Fig -4: Time varying demand of CPU

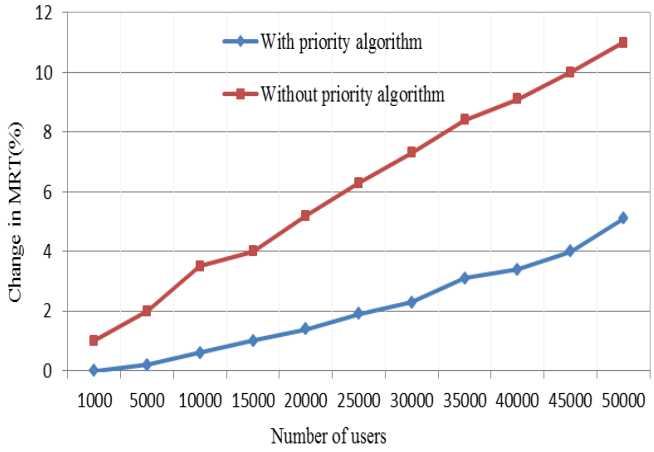

Fig -5: Performance change with same resource level

3. DISCUSSION

The objective of these experiments was to test the effectiveness of the architecture to handle real time services, in smart grid environment. The study focuses on the performance metrics. It is apparent from various results that the proposed architecture in open stack cloud environment maintains satisfactory level of responses for varying request rate. Also found that performance is affected by the resource capacity available to the system and the time varying demand of the applications. The cloud environment succeeds to limit the failure rate and offered a reasonable value of MRT for the cases considered. Scheduling the applications according to priority aids the dynamic resource allocation in virtualized server environment. Result reveals that, at higher values of resource demand, resource utilization is dynamically adjusted and resources are allocated on time. The open source software considered in the experiment permits up gradation of existing architecture for incorporating more features. The architecture can be extended to incorporate scheduling algorithms based on other performance metrics. The analytical model developed for evaluating the performance of the architecture is described in the next section. The model considers the characteristics of resource allocation in a dynamically changing workload situation

4. ANALYTICAL MODEL FOR PERFORMANCE EVALUATION.

The analytical modeling proposed in the work capture the performance of the multiple applications hosted in a virtualized environment. The work considers queuing theory based modeling to characterize the performance of cloud environment, where applications are distributed across multiple virtual machines with time varying resource demand. The modeling also uses the terminology resource to represent all the resources shared by a virtual machine on a physical server, e.g., resource consumption is considered in place of CPU consumption. The paper [13] presented an analytical model for multi-tier Internet application based on a network of queues to represent how the tiers in multitier application cooperate to process requests and demonstrated the utility of the model in managing resources for internet application under varying workloads and shifting bottlenecks. However, a virtual machine differs from a physical server in that its effective capacity varies with dynamic resource allocation. In dynamically changing workload situations, the optimal performance can be achieved by allotting the application request based on some priority. The approach used in this work is to categorize the different applications deployed in the deployment platform as high priority and low priority ones. The operational mode in the algorithm involves customer arrival ruled by the distribution of inter arrival time according to a poison process. The aggregate user request rate, R, is defined by equation (1), where N is the number of different

applications deployed and Rn is the mean user request rate of application n.

$$R = \sum_{n=1}^{N} R_n \quad (1)$$

The queue discipline is according to a priority with negligible delay and the resources in the virtualized environment can be modeled as M/G/1/PRIOR queue [14]. According to queuing theory, total resource resident time by all the request served in tier k is represented by qk/(1-qk), where qk is the resource utilization in tier k. For an aggregate user request rate of R, the mean resident time for applications with higher priority, Tsh can be described as

$$T_{sh} = \frac{1}{R}\left(\frac{q_k}{1-q_k}\right) \quad (2)$$

Resource consumption for a typical application is proportional to number of request. Let Dn represent the mean demand of applications with higher priorities. Then resource consumption by highest priority application, Mc, can be defined as a linear function of user request rate.

$$M_c = \sum_{n=1}^{N} D_{nk} * R_n \quad (3)$$

The resource utilization by all the application in the virtualized environment is defined as the ratio between the virtual machine's resource consumption and its effective resource capacity. In the virtualized environment resource capacity is dynamically modified. The utilization changes according to the changes in allocation. Hence resource utilization is modified as the ratio between resource consumption of the virtual machine and resource allocation to the virtual machine. Let Ma be the resource allocation, refers to the resource capacity that is allocated to a virtual machine. Incorporating the modification in (2),Tsh can be represented as

$$T_{sh} = \frac{1}{R}\sum_{k=1}^{K} \frac{\sum_{n=1}^{N} D_{nk} * R_n}{M_a - \sum_{n=1}^{N} D_{nk} * R_n} \quad (4)$$

The modeling assumes applications with low priority are available with maximum resources. Since priority inversion is considered in the algorithm the modeling considers total number of applications for the calculation resident time in this case also. Mean resident time for applications with low priority is represented by Tsl, can be approximated as (5), where values of φn represent mean resident time for applications with lower priorities. The value can be obtained through model calibration. Combining (4) and (5) the mean response time can be represented by aggregating resident times over all resourses by higher priority and lower priority applications. Thus once the values of average demand and mean resident time for applications with lower priorities are obtained, mean response time can be predicted from equation (6).

$$T_{sl} = \frac{1}{R}\sum_{n=1}^{N} \varphi_n * R_n \quad (5)$$

$$MRT = T_{sh} + T_{sl} \quad (6)$$

The linear regression method is used to estimate the mean demand from (3) and response time from equation (6) for a given arrival rate of request. The estimate considered average user request rate of 1800- 2400 per minute and a mean response time target of 35ms to 45ms.

The measured values of MRT were compared with those estimated using (6) for evaluation. The performance result in the test for normal and high utilization of resources is depicted along with estimated values in figure 6 and figure 7.The model is valid for the present case as it captures the effect of resource capacity, time varying demand and response with respect to complexity in applications. The estimate considered the same entitlement and user request rate as used in the test cases. The error between the measured and estimated values of response times for various cases is negligible. The result justified the architectural performance.

The modeling used in the work is application oriented. The model can predict the performance of the cloud environment with various applications, running on the virtualized servers having different computational needs. The modeling considered here needs to include more parameters for a complex networked structure. In such cases, advanced non linear solution methods provide precise results than regression method.

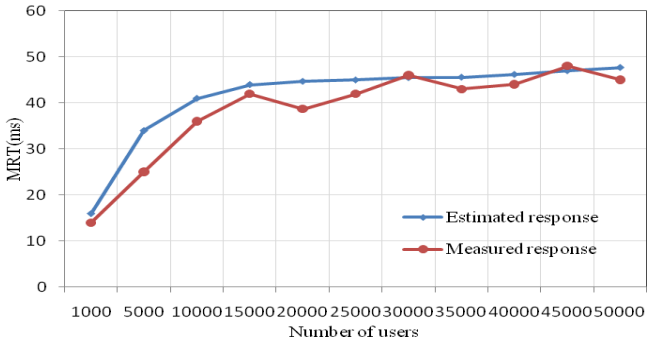

Fig -6: Response under normal utilization

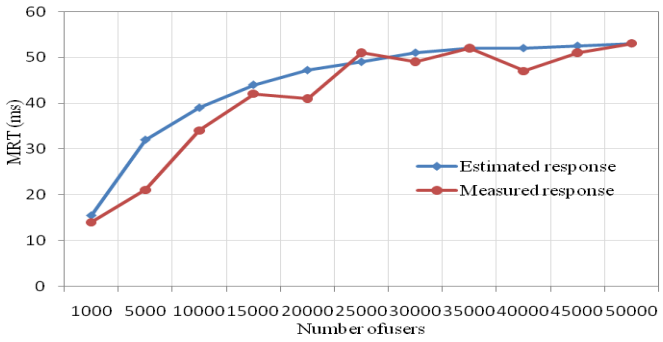

Fig -7: Response under high utilization

CONCLUSIONS

The architecture explores a simplified approach through functional building blocks; all aimed at delivering real time services in service oriented structure. It support creation, execution and evolution of service oriented solutions. A priority scheduling algorithm was applied to the architecture for performance management.

The proposed algorithm was tested with the application services relating to power grid which has time varying processing and storage need. The result showed that the architecture have better performance with the algorithm. Analytical model based on queuing theory to capture the performance of the architecture has been developed for evaluation. The result justified that the proposed architecture is effective for real time applications services in smart grid environment.

The work is continuing with the modifications in the architecture and algorithm to develop load management system for smart grid that can optimally coordinate timely demand side management.

REFERENCES

[1] G.T.Heydt, B.H.Chowdhury, M.L.Crow, D.Haughton,B.D.Kiefer,F. Meng and B.R.Sathyanarayana “Pricing and Control in the Next generation Power Distribution System” IEEE Transactions on Smart Grid,Vol. 3,No. 2,June 2012,pp.907-914

[2] F.Katiraei and M.R.Iravani, “Power Management Strategies for a Microgrid With Multiple Distributed Generation Units”, IEEE Transactions on Power Systems,Vol.22,No.4,November 2006,pp.1821-1831

[3] Soma Shekara Sreenadh reddy, Depura, Lingfeng wang, vijay Devabhaktuni, “Smart meters for Power grid Challenges,issues,advantages and Status”, Renewable and Sustainable energy Reviews, Vol.15,2011,pp.2736-2742.

[4] R Rusitschka, K. Eger, C. Gerdes, "Smart Grid Data Cloud: A Model for Utilizing Cloud Computing in the Smart Grid Domain", in Proceedings IEEE International conference on Smart Grid Communications,2010., pp.-483-488

[5] Amir-Hamed Mohsenian-Rad, Albert Leo-Garcia , “Cordination of Cloud Computing and Smart Power grids”, in: Proceedings IEEE International conference on Smart grid communications, 2010, pp. 368-372.

[6] Rajeev T., Ashok S, A Cloud Computing Approach for Power Management of Micro grids, in: Proceedings IEEE PES Innovative Smart grid Technologies, India (ISGT-India) 2011, pp. 49-52

[7] Pinal Salot, “A survey of various scheduling algorithms in Cloud Computing Environment”, International Journal of Research in Engineering& Technology(IJRET), Vol. 2,No. 2,February 2013,pp.131-135

[8] Zhikui Wang, Yuan Chen, Daniel Gmach, Sharad Singhal, Brian J. Watson,Wilson rivera,Xiaoyun Zhu, and Chris D. hyser, “AppRAISE:Application-Level Performance Management in Virtualized Server Environments,” IEEE Transactions on Network and service management ,Vol. 6, No.4.Dec 2009.pp.240-253.

[9] Ruben Van den Bossche , Kurt Vanmechelen, Jan Broeckhove, “Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds”, Future Generation Computer Systems, Vol.29, May 2013,pp. 973-985

[10] Brijesh Goyal,Pallavi Jain, “Reminiscing Cloud Computing Technology”, International Journal of Research in Engineering& Technology(IJRET),Vol. 1,No. 2,November 2012,pp. 364-367

[11] Wei-Tek Tsai, Xin Sun, Janaka Balasooriya, “ Service-Oriented Cloud Computing Architecture”in:,Proceedings on Seventh International Conference on Information Technology,April,2010.pp. 684-689.

[12] OpenStack Beginner's Guide(for Ubuntu - Precise) v3.0, 7 May 2012

[13] B.Urgaonkar,G.Pacifici,P.Shenoy,M.Spreitzer,and A.Tantawi, “An analytical model for multi-tier internet services and its applications,”ACM

SIGMETRICS Performance Evaluation Review,. Vol 33,June 2005,pp.291-302.

[14] E.D.Lazowska, J.Zahorjan, G.S.Graham, and K.C.Sevcik, uantitative Sysyem Performance: Computer System Analysis Using Queuing Network Models.upper Saddle River,N.J:Prentice-Hall,Inc.,1984.