ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 8, February 2015

Implementation of Monitoring System in Cloud Computing Using Agents

Sonali Agrawal, Deepak Choubey (asst. prof.)
Dept of Computer Science & Engg.
SRIT Jabalpur (M.P.), India

Abstract— ***Scheduling and monitoring is the main challenge in cloud computing paradigm. Various methods have been proposed but they have their own pros and cons. The main challenge is distributive nature of the cloud computing that create the problem to implement the proper scheduling of the service (mainly SaaS) and their efficient monitoring. Various users have access the service of the cloud simultaneously and scheduling monitoring of at the same time is bit challenging. Proposed system has adopted the idea of software agent to gain the high efficiency in the same direction (scheduling and monitoring).This article presents the method for computing number of resources used and the solution for better elasticity and their efficient monitoring of the resources in the cloud which helps to gather analytical statistics of the resources currently held and will be used such a memory, number of instances and CPU. Proposed mechanism has influences from the working of Aneka framework. For evaluation of the proposed work, the components has been used, first the data set which is the web application (jsp) developed for testing in cloud environment. Then for monitoring and scheduling with software agent New Relic service has been used to customized the agent functionality to meet the propose systems requirement.***

Keywords- **Agent, Cloudbees, Cloud computing, Codenvy, Monitoring, Public Cloud, Provisioning, New Relic, Scheduling.**

I. INTRODUCTION

Heterogeneity in the computational requirement, dynamic choice and infrequent usages types of resources of the users in modern era has main challenge for service provider (application developer and hardware manufacturer). Secondly, Now computing power or connected computing power (with network) has more demanding and significant role in almost all areas of epoch including market analysis, searching, map, accounting, medical, trading, shopping, rescue operations and many more, the list is endless. Various devices (computing) and application has been developed and developing to fulfill the common users need. However different users have different requirements of computational power and application and systems software. Hence demand of users is heterogeneous in nature so that varieties of application (hardware & software) have been developed to achieve the highest user satisfaction. Advancement of electronics and telecommunication field has done the job. Specialization has more promising than generalization due to expertise in specific job/function but it also has dark sides. Various requirements require numerous specialized devices (CPU, storage etc.) and software tools. Purchasing or licensing of all such required items (devices & applications) is not feasible to the organization or individuals in terms of the cost and installation. Secondly most of the resources are idle i.e. frequently not used. Hence the utility types of computing paradigm will play an import role. Cloud computing is a new computing paradigm based on utility computing model which will fulfill the user's requirement dynamically on rent basis. According to the Lewis Chunningham [36] "Cloud computing is the internet to access someone else's software running on someone else's hardware in someone else's data center". More comprehensive concept about cloud computing has been narrated and drafted by National Institute of Standard & Technology (NIST): According to NIST- "Cloud computing is a paradigm for facilitating expedient, on-demand network access to a shared cluster (pool/collection) of configurable computing power and resources (like applications, services, networks, servers, and storage,) that can be expeditiously provisioned and exemption with least management endeavor or without service provider interaction. This cloud paradigm endorsed availability and is possessed of five imperative characteristics, three service models, and four deployment models." Cloud computing is fast growing as an alternative to conventional computing. However, the paradigm is somehow same as, utility computing, grid computing, cluster computing and distributed computing in approximately .Cloud computing fabricate a virtual paradigm for sharing data and computations over a scalable network of nodes. Examples of such nodes include end user computers, data centers, and web services. Such a scalable network of nodes is called cloud. An application based on such clouds is taken as a cloud application. Cloud computing is modern TCP/IP integrations of computer and network technologies such as fast micro processor, gigantic memory, high-speed network and reliable system architecture [4]. Normally cloud computing services are organized into three groups:

a. Software-as-a-Service (SaaS)
b. Platform-as-a-Service (PaaS) and
c. Infrastructure-as-a-Service (IaaS)

Cloud computing also is divided into five layers including clients, applications, platform, infrastructure and servers. The five layers look like more reasonable and clearer than the three categories [11]. An Amazon EC2 instance is a virtual processing resource (VM) in the Amazon cloud. The progression of instantiating latest (fresh) Virtual Machine might take as long as few minutes. The new VMs originate either as fresh boots or replicas of an existing virtual Machine (template VM), unconscious of the existing application state. For monitoring the resources various methods and tools has been used like CloudWatch popular in Microsoft Azure and Amazon EC2 cloud. CloudWatch is a type of web service application that monitored the instances (resource provision), as well as also responsible for elasticity for the subscribed services i.e. Auto Scaling [4] as per the need of the subscribed application by the cloud subscriber. For instance, checking condition has been set (like threshold condition) whenever there is need of additional resources or less number of resource. For example, if the mean CPU utilization (of particular subscribers application) is greater than 70% add more resources automatically, or remove the excess resources when mean CPU utilization in below 10%. It takes an action based on statistics collected and exposed by CloudWatch contrary to our work. These metrics take a purely system view such as utilization, but not the application view such as average response time of a request, or an associated Service Level Objective (SLO). Further, Auto Scaling is atheist to the need for provisioning data resources desirable for workload execution. Amazon claims that the latency and throughput of the volumes are designed to be significantly better than the instance's local store. Conversely, a volume can only be attached to only single instance. Microsoft Windows Azure does not offer automatic scaling, but it is the primary tool for provisioning. Subscriber can provision any number of instances that they craving to have available for deployed application. Similar to Amazon EC2, the instances are virtual processing resources. Effectively, Azure provides provisioning mechanisms which can be used by a management function to improve application and system metrics. Systems that jointly employ scheduling and provisioning techniques have been explored in grids. The Falkon scheduler triggers a provisionary component for host increase or decrease. This host variation has also been explored during the execution of a workload, hence providing dynamic provisioning. The scheduling and monitoring is the main challenge in cloud computing paradigm. Various methods have been proposed but they have their own pros and cons. The main challenge is distributive nature of the cloud computing that create the problem to implement the proper scheduling of the service (mainly SaaS) and their efficient monitoring. Various users have access the service of the cloud simultaneously and scheduling monitoring of at the same time is bit challenging. Propose system has adopted the idea of software agent to gain the high efficiency in the same direction (scheduling and monitoring).

II. RELATED WORK

Cloud computing is fast growing as an alternative to conventional computing. However, the paradigm is somehow same as, utility computing, grid computing, cluster computing and distributed computing in approximately .Cloud computing fabricate a virtual paradigm for sharing data and computations over a scalable network of nodes. Examples of such nodes include end user computers, data centers, and web services. Such a scalable network of nodes is called cloud. According to author [3], cloud computing depicts- The modern trends Cloud computing designated to provision application (SaaS and PaaS) and efficient data center service using in conjunction with hardware and software mutually delivered via internet (network in case of private cloud within campus) on rent basis. Virtualization is the core concept behind cloud computing. One of the main key function of the cloud computing is elasticity (shrink-in and shrink-out) whereabouts required (variable) number of in instances of VM (Virtual Machine) has been created dynamically to fulfill the users heavy (shrink out) and light (shrink in) on the users application demands [1,2]. The cloud applications themselves have long been known as Software-as-a-Service (SaaS). SaaS is a software delivery paradigm where the software is hosted off-premises, developed by service providers and delivered via Internet and the payment mode follows a subscription model [3]. For SaaS providers, having the power to scale up or down an application to only consume and pay for the resources that are required at that point in time is an attractive capability and if done correctly it will be less expensive than running on regular hardware from traditional hosting [1]. Author [3] found that, However, in spite of the advantages of using cloud computing to create highly scalable applications, solving performance problems through cloud computing is not a trivial decision if involved costs are analyzed [4]. For example, Amazon Web Services charges by the hour for the number of instances you occupy, even if your machine is idle. In 2008, the image-processing Animoto application deployed over Amazon EC2 infrastructure [5] experienced a demand surge that resulted in growing from 50 servers to 3500 servers in three days; after the peak collapsed, network traffic knock over to a point so that it was well beneath the max point. Increasing the number of resource (positive scalability) to achieve shrink-out i.e. scale- up elasticity was not a cost expansion approach although it's an viable prerequisite, whereas shrink-in i.e. scale-down elasticity permitted the steady-state expenditure to more closely match the steady-state workload. Indeed, Animoto's provider charges by 3500 virtual instances because a peak load occurred at a certain time frame and when this peak disappeared, it would pay for unused resources [4]. This effect is still a barrier for SaaS providers, whose applications have different peak loads and they are highly prone to suffer over and under provisioning of resources [6,7]. A computing cloud is a gigantic network of nodes. Hence, elasticity or extensibility i.e. scalability ought to be an eminence feature of the cloud. The paramount scalability is Horizontal scalability, that is the capability to connect and incorporate various clouds to endeavor as

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 8, February 2015

single virtual/logical cloud [3,4]. For instance, a cloud providing calculation services (calculation cloud) can access a cloud providing storage services (storage cloud) to maintain transitional outcomes. Two computational clouds can also join together into a bigger computational or calculation cloud. Scalability should be transparent to users. For illustration subscriber users can cache their information in the cloud devoid of the necessitate to discern where it stores the data or how it reprieve the data. For instance every cloud has only a fixed volume of physical storage units. Therefore, a cloud a1 may inquire about facilitate from a different cloud a2 for shared storage units to perform some demands on related to storage. Such sharing requirement may result in the data to migrate among multiple clouds. However, the cloud subscriber ought to not be conscious of the scattered storage structure of the information of his/her interest i.e. distributed [5]. Suppose whilst subscribers needs to retrieve the cached data, the user may perhaps exactly accessed it from the subscribed cloud i.e. cloud a1. Then c1 is responsible for gathering the data from both a1 and a2, and returns the collected data to the user. Consequently cloud offers location transparency to subscribed applications i.e. seems too accessed from local (subscribed) cloud.

III. EXISTING WORK IN CLOUD MONITORING

The work in [37] describes a distributed monitoring service, implemented in Java and JINI (with WS-* bindings), called MonaLISA (Monitoring Agents in A Large Integrated Services Architecture). An agent in MonaLISA represents a service (i.e., that can be used by other services or clients) that is discoverable, self-describing and able to collaborate and cooperate with other services in various monitoring tasks. Collected data is stored, per service, in a local relational database. The Data Collection Engine directs MonaLISA's function. Clients may request both real-time and historical data through use of various filtering mechanisms (e.g., predicates, Agent Filter). Clayman et al. [38] [39] describe the Lattice monitoring framework, designed to be a base framework on top of which monitoring systems may be built. Though in agreement with most of the requirements we specify here, their focus is more on the actual probes for sensing low-level metrics (e.g. CPU utilization probe). Authors are more interested in the collection, aggregation, and distribution of application- and system-level metrics from third-party probes. The work in [40] introduces an architecture for and implementation of a private cloud monitoring system. The architecture is quite high-level and is composed of three layers: an Infrastructure layer, an Integration layer and a View layer. The implementation is modular in design and consists of several components that are mostly focused on the integration layer of the architecture. Currently, it is compatible with Eucalyptus (as a IaaS implementation); however, it is mentioned that it could be extended to work with alternative IaaS implementations in the future. It appears to rely quite heavily on Nagios for its monitoring functionality. In [41], Kanstre and Savola define a set of requirements for a distributed monitoring framework and a reference architecture that satisfies those requirements. The requirements include scalability, correctness, security, adaptation and intrusiveness. The architecture is a conceptual layered architecture and there is no reported realization of it. An implementation of a distributed network monitoring framework was proposed in [42]. The authors showed how a three tier layered framework can be used for monitoring computer networks in geographically distributed locations. Compared with the above two approaches, our approach is better-suited to federated systems of clouds, though we share some common requirements such as scalability. A cloud monitoring framework was proposed by Sun et al. [43]. The authors use a conceptual Service Oriented Architecture and focus mostly on message interchange among entities and on the integration of the framework with the existing system management processes. There is no implementation or evaluation of the architecture. Following works has need the next conceptual step, recognizing the importance of scalability and intercloud monitoring on a loosely-coupled publishes-subscribe architecture. Lahmadi et al. [44] present a benchmark effort for defining metrics for evaluating a performance management framework. Their metrics include overhead, delay and scalability in the context of networks and services. Balis et al. [45] described Gemini2, a monitor for grids built on a complex event processing (CEP) system called Esper (stream processing is in the same space as complex event processors). They suggest CEP is well-suited for monitoring as it enables access to streams of data in real time. They propose what amounts to substantial shift in how monitoring information is consumed: the end user writes and submits an SQL-like query requesting information, and the system deploys sensors to acquire this monitoring information which is then pushed to the end user. Support for existing monitoring services or even conventional monitoring paradigms is not included. The idea of using public clouds to enhance the capability of grid resources has been explored theoretically in different works. Assunçaoetal. [46] present a simulation-based analysis of different algorithms for provisioning of resources both in a local cluster and in the cloud. Such an analysis is based on common grid and cluster workloads. Kondo et al. [47] present a cost-analysis study of mixed cloud and desktop grid environments for high-throughput, CPU intensive applications. Such a study shows that hybrid approaches where servers for the desktop grid are hosted in the cloud enable savings in infrastructure costs. Regarding actual implementations of systems supporting hybrid clouds for scientific applications, Comet Cloud [48] is an autonomic engine for hybrid grids and cloud

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 8, February 2015

systems, which supports the execution of workflow applications. Aneka, on the other hand, provides support for different programming models such as workflow, MapReduce, threads, and Actors oriented programming. Moreover, it can also exploit resources from idle desktop machines, including those running the Windows operating system. The ASKALON grid environment has been extended [49] to support the execution of workflow applications in both grids and clouds (either public or private). The CaGrid Workflow Toolkit [50] performs discovery, data access, service invocation, and execution of workflows in multiple types of resource. Both systems support only workflow applications and limited types of resource, whereas Aneka supports different programming models and computing environments. GridWay [51] supports the execution of applications both in local grids and in different cloud providers with the help of Globus Nimbus. It supports any type of local resource that can be managed by the Globus middleware, and also supports programming models supported by the latter. Therefore, both GridWay and Aneka are able to provision any type of resource to applications, even though Aneka supports more application models than GridWay. Finally, Elastic Site Manager [52] is a resource manager that is able to dynamically provision resources from private and public clouds to scientific applications. Open Nebula combined with Haizea [53] supports the dynamic provision of virtualized resources from private and public Clouds. Resources managed by these systems are virtualized resources only, whereas Aneka is able to leverage applications with both virtualized and non-virtualized resources simultaneously, due to its provisioning capabilities. Problem identified by [1]: From this scenario, the cloud data storage and access may need not only intra-cloud interactions; however it can also inter-cloud interactions. That is to say, the cloud data haven't only be retrieved in a LAN, but also roamed in WAN. In LAN environment, cloud computing system can use Remote Procedure Call (RPC) or Remote Method Invocation (RMI) as the intrinsic capacity, to implement the service directory coherence and service migration. RPC and RMI can accomplish excellent efficiency in Local Area Network, but inappropriate for Internet or Wide Area Network [6]. Mobile agents on the Internet or WAN have the characteristics as follows: Autonomy, Personality, Communication, Mobility and High Performance and Fault tolerance [7]. Mobile agents are mainly intended to be used for applications distributed over wide area (slow) networks because they can save communication costs by moving the resource and service to the remote target environment which is near the user. A mobile agent based cloud computing system for WAN (SaaAS) is presented by the [1]. According to author [1], with the help of mobile agent rather than RPC/RMI as the underlying facility to implement the service directory coherence and service migration, SaaS is more suitable to work in Internet. In the article [1], author presents a code and data of service load mechanism based mobile agent and divided-cloud and convergent coherence mechanism of SaaAS, which can effectively reduce the heavy communication overhead in Internet. Problem identified by [2]: In article [2], author has surveyed many problem associated with cloud service delivery especially while talking about the service interoperability and portability of the data in the cloud.

IV. ENHANCED AGENT BASED SCHEDULING & MONITORING SYSTEM IN CLOUD COMPUTING

Modern era is reflection of human creative thinking and application of optimize solution for the problems mapped and simulated into the machines using technological skills and advancement on it. Cloud computing is another example of technological advancement which offers dynamic provisioning of the utility on rent basis to the subscriber. Cloud offers instant service (software, platform or infrastructure) to requisite dynamically. Sometime's subscriber need more resource (like network bandwidth, CPU or memory) or sometimes it require to less. Cloud service provider has deploy and manage to sufficient number of resource that has been shared to all the subscriber as per the load requirement of the individuals. To achieve this task cloud service provider required highly efficient scheduling approach and the proper monitoring of the services provisioned or will be provision to subscriber. All the scheduling and monitoring assured the high reliability, automatic scalability, fault tolerance services in secure manner. In the cloud computing. Agents are the self executable code work on behalf of the humans. They are able to communicate i.e. social in nature, mobile i.e. can roam in the network, perform the task at remote stations and send back the results to source platform (where they been originated), agents are also clone themselves and one of the core property of the agent is autonomy i.e. autonomous and distributive in nature. For implementation and evaluation of proposed approach public cloud has been chosen due to cost effective experimental setup. For developing proposed agent based system three types of public cloud and their services has been selected as test bed for better evaluation and measurement of the accuracy of the propose system. They are following with respective functionality in the proposed system-

1. Codenvy – To develop an/are application i.e. SaaS (Software as a service). For the proposed system an java web application using jsp (Java Server pages) application has been chosen to develop on to the codevny SaaS cloud service provider.
2. Cloudbees – To deploy and test our SaaS application onto the cloud, propose system needs a platform i.e. Platform as a Service (PaaS). For this Cloudbees service provider has been integrated onto the developed SaaS application.

3. New Relic – To develop the core functionality of the proposed system .i.e. monitoring and scheduling using software agent New Relic service has been subscribed. In this the java agent has been customized to meet the monitoring and scheduling of the SaaS services.

A. Problem Identification and Proposed Solution

Propose system has surveyed and identified the problem domain that must be addressed in context of the cloud computing and consequently present the idea of agent integration. These are following-

1. Service scheduling delay
2. Elasticity optimization
3. Better Provisioning of the SaaS
4. Fault tolerance

While author [1] and [2] proposed an agent based solution to solve the above listed QoS parameter that greatly affect the performance of cloud service especially SaaS. But the main problem while looking [1] and [2] is the realization and effectiveness of the agent with cloud for better optimization of the service delivery. The main lacking point in the article [1] and [2] is validation of the proposed mechanism. Additionally the requirements for such fast provisioning of the cloud has been discuss in the recent year in the article [3]. Our main research work is to enhance the agent based model for SaaS delivery in the cloud as depicted in the [1] and [2]. Following goals has been set during experimental setup as a objective to solved with integrating of the Mobile Agent to Cloud Computing service realization-

- To Evaluate and delivered the cloud computing services (SaaS) using agent (for better and fast delivery) using public cloud such as “New Relic and cloud bees”.
- Deploying a web services under SaaS paradigm and evaluate the effectiveness of the web application in the cloud environment with the help of agent. For SaaS development Codenvy has been subscribed. In which jsp based application has been develop and deployed on cloud bees PaaS.
- Evaluation and Public PaaS (platform as a Service) of the Cloudbees service integrating a SaaS deployment on it and delivering through agent.
- Measuring the performance of the proposed analytical approach (influenced from Aneka) in cloud services such as public Cloud bees.

B. Proposed Algorithm

Provisions of service and resources in cloud PaaS is an important function that provides analytical statistics about the current view of cloud (running instance for a user or group of users).

Model for Proposed Work:

Our proposed work is to schedule and monitor cloud SaaS application onto the cloud and evaluate the performance of the same using proposed agent based.

Proposed Algorithm for Provisioning Application and Resources

Algorithm for Scheduling (influenced from Aneka) developed onto the Cloudbees-

```
1. Initialize agent to continuous monitor the resources to cl
   send report to the Monitoring_Agent –
   1. Relaese_resources_Agent = List all the resources_avaia
   2. Memory_agent = Calculated the free_space()
   3. Throughput_Agent = Check and monitor the requested_
   4. Req_res_agent = check the request and reply
   5. Network_usages
   6. CPU_Agent = Calculate the total free capacity of C
      subscription)
2. Monitoring Agent – It Checks the required resource
   (Quality of Service) and load requirement to the subscrib
  for each subc_request with QoS constraints:

   {
    resources = available_resources for the requested SaaS appli
    call Relaese_resources_Agent();
    call Memory_agent();
    call CPU_Agent();
  Jobs_pending = number of jobs in the queue;
  effort = (Jobs_pending /resources)× averageJobsRuntime;
  call Req_res_agent();
    if (effort > Remaining_Time_application)
     {
               additionalResources = (Jobs_pending×averag
                                            Remaining_Time_
          Call CPU_agent();
          call Relaese_resources_Agent();
           CALL_Monitoring_Agents(job_Id);   // for resource p
      }
     else
          toRelease = 0;
          call Relaese_resources_Agent();
      if (Jobs_pending < resources)
      {
          toRelease ← Jobs_pending − resources;
          call Relaese_resources_Agent();
      }
     else
     {
        Call CPU_agent();
        Jobs_pending = Jobs_pending + Jobs_running;
        Effort = (Jobs_pending /resources)× averageJobsRuntim
        if (effort < Remaining_Time_application)
         toRelease ←resources –(Jobs_pending×averageJobsRun
                          Remaining_Time_application

     }
     CALL_Relaese_resources_Agent(job_Id);
   end
end
```

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 8, February 2015

C. Monitoring of SaaS using Java Agent

Set of agent has been customized and configured as a java agent onto the new relic to perform the monitoring of the SaaS application.

Set of Agents-

7. Monitoring_Agent
8. Relaese_resources_Agent
9. Memory_agent
10. Throughput_Agent
11. Req_res_agent
12. Network_usages
13. CPU_Agent etc.

Brief Summary of the task assigned to Agent-

1. Resource utilization and monitoring like network, memory, I/O request has been monitored by the **Monitoring_Agent** the calculation of the required resources has been evaluated using above mentioned algorithm to ensure elasticity.

```
Monitoring_Agent(Job_id)
 {
    Check the resources required to the job pending
    Call Memory_Agent(); //calculate the size required to store the job in memory
    Call CPU_Agent()
    {
        Check the priority and computational power required for the job
            Call Req_res_Agent();
    Call Network_Agent()
        {
            Check the achieved Throughput during provisioning
            Call Throughput_Agent();
        }
}
```

2. Error monitoring and page load response of the SaaS while accessing from the browser and respective users.
3. Relaese_resources_Agent

V. CONCLUSION

This paper the enhanced agent based solution to ensure better elasticity and monitoring solution. Analytical analysis is to collect statistics to check the required number of resources needs or used and provides dynamic indication to better elasticity achievement. Proposed agent based solution for guaranteed better elasticity and their efficient monitoring of the resources in the cloud which helps to gather analytical statistics of the resources currently held and will be used such a memory, number of instances and CPU. To deploying created SaaS application in the cloud a PaaS service has been required to be subscribed, for this cloud bees PaaS service has been chosen. Then for monitoring and scheduling with software agent New Relic service has been used to customized the agent functionality to meet the propose systems requirement. In this article, the fundamental of cloud computing with their latest functionality has been presented.

REFERENCES

[1] Gaoyun Chen, Jun Lu, Jian Huang and Zexu Wu "SaaAS - The Mobile Agent based Service for Cloud Computing in Internet Environment", IEEE, Sixth International Conference on Natural Computation (ICNC 2010), pp. 2935 – 2939, 2010.

[2] Zehua Zhang and Xuejie Zhang "Realization of Open Cloud Computing Federation Based on Mobile Agent", IEEE, pp. 642 – 646, 2009.

[3] Javier Espadas, Arturo Molina, Guillermo Jiménez, Martín Molina, Raúl Ramírez and David Concha "A tenant-based resource allocation model for scaling Software-as-a-Service applications over cloud computing infrastructures", Elsevier Science Direct, Future Generation Computer Systems 29 (2013), pp. 273–286, 2013.

[4] IBM Blue Cloud project [URL]. http://www-03.ibm.com/ press/us/en/pressrelease/22613.wss/.

[5] Rizwan Mian, Patrick Martin and Jose Luis Vazquez-Poletti "Provisioning data analytic workloads in a cloud", Elsevier, Future Generation Computer Systems, pp. The Characteristics of Cloud Computing, 2012.

[6] Chunye Gong, Jie Liu, Qiang Zhang, Haitao Chen and Zhenghu Gong "The Characteristics of Cloud Computing", 39th International Conference on Parallel Processing Workshops, pp. 2010.

[7] Amit Nathani, Sanjay Chaudhary and Gaurav Soman "Policy based resource allocation in IaaS cloud", Elsevier, Future Generation Computer Systems 28 (2012) 94–103

[8] Naidila Sadashiv and S. M Dilip Kumar "Cluster, Grid and Cloud Computing: A Detailed Comparison",IEEE, The 6th International Conference on Computer Science & Education (ICCSE 2011) August 3-5, 2011. SuperStar Virgo, Singapore.2011.

[9] Bernd Grobauer, Tobias Walloschek and Elmar Stöcker Siemens "Understanding Cloud Computing Vulnerabilities", IEEE, Copublished By The IEEE Computer And Reliability Societies, 2011

[10] Jianwei Yin, Yanming Ye, Bin Wu and Zuoning Chen "Cloud Computing Oriented Network Operating System and Service Platform", 1st IEEE Workshop on Pervasive Communities and Service Clouds, IEEE, 2011

[11] A

[12] Dan Svantesson and Roger Clarke "Privacy and consumer risks in cloud computing", Elsevier, compute r law & securi ty rev iew 26 (2010) 391e397.

[13] L.M. Vaquero, L.R. Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," ACM SIGCOMM Computer Communication Review, v.39 n.1, 2009.

[14] D. Chappell, Introducing Windows Azure, David Chappell & Associates, 2009.

[15] I. Raicu, Y. Zhao, C. Dumitrescu, I. Foster, M. Wilde, Falkon: a fast and light-weight tasK executiON framework, in: Proceedings of the 2007 ACM/IEEE Conference on Supercomputing, ACM, Reno, Nevada, 2007.

[16] Frederick Chong and Gianpaolo Carraro, "Architecture Strategies for Catching the Long Tail," Microsoft Corporation, April 2006. http://msdn.microsoft.com/enus/library/aa479069.aspx.

[17] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," 2008, IETF RFC 5246, http://www.ietf.org/rfc/rfc5246.txt.

[18] A. Baumann, P. Barham, P.-E. Dagand, T. Harris,R. Isaacs, S. Peter, T. Roscoe, A. Schüpbach, and A. Singhania, "The multikernel: a new OS architecture for scalable multicore systems". In Proceedings of the ACM SIGOPS 22nd Symposium on OS Principles, 2009, pp. 29–43.

[19] A. Schüpbach, S. Peter, A. Baumann, T. Roscoe, P. Barham, T. Harris, and R. Isaacs. "Embracing diversity in the Barrelfish manycore operating system". In Proceedings of the Workshop on Managed Many-Core Systems (MMCS) 2008. ACM, June 2008.

[20] S. Peter, A. Schüpbach, P. Barhamy, A. Baumann,R. Isaacsy, T. Harrisy and T. Roscoe,"Design Principles for End-to-End Multicore Schedulers". In 2nd Workshop on Hot Topics in Parallelism, Berkeley, CA, USA, June 2010.

[21] D. Malcolm, "The five defining characteristics of cloud computing," http://news.zdnet.com/2100-9595_22-287001.html.

[22] P. Sharma, "What kind apps are best suited for 'Cloud deployment': 4 Solutions," http://www.techpluto.com/cloud-computing-characteristics/.

[23] D. Amrhein, "Forget Defining Cloud Computing," http://ibm.ulitzer.com/ node/1018801.

[24] D. Wentzlaff and A. Agarwal. "The Case for a Factored Operating System (fos)", MIT CSAIL Technical Report, MIT-CSAIL-TR-2008-060, October 2008.

[25] D. Wentzlaff and A. Agarwal. "Factored Operating Systems (fos): The Case for a Scalable Operating System for Multicores". ACM SIGOPS Operating System Review (OSR), April 2009.

[26] Sean Marston, Zhi Li, Subhajyoti Bandyopadhyay,Juheng Zhang, and Anand Ghalsasi "Cloud computing—The business perspective", Elsevier, Decision Support Systems 51 (2011) 176–189.

[27] M. Armbrust, et al. Above the clouds: a Berkeley view of cloud computing, electrical engineering and computer sciences, Technical Report No. UCB/EECS-2009-28, University of California at Berkeley, February 2009.

[28] Dimitris Zeginis, Francesco D'andria, Stefano Bocconi, Jesus Gorronogoitia Cruz, Oriol Collell Martin, Panagiotis Gouvas, Giannis Ledakis And Konstantinos A. Tarabanis "A User-Centric Multi-Paas Application Management Solution For Hybrid Multi-Cloud Scenarios", Scalable Computing: Practice and Experience (SCPE), Volume 14, Number 1, pp. 17–32.avaialable at http://www.scpe.org.

[29] C. S. Yeo, R. Buyya, M. Dias de Assuncao, J. Yu, A. Sulistio, S. Venugopal, and M. Placek. Utility Computing on Global Grids. In H. Bidgoli, editor, Handbook of Computer Networks, Wiley Press, Hoboken, NJ, USA, 2008.

[30] R. Buyya, J. Broberg, and A. Goscinski (eds). Cloud Computing: Principles and Paradigms. ISBN-13: 978-0470887998, Wiley Press, USA, February 2011.

[31] C. S. Yeo and R. Buyya. Integrated Risk Analysis for a Commercial Computing Service. Proceedings of the 21st IEEE International Parallel and Distributed Processing Symposium (IPDPS 2007), Long Beach, CA, USA, March 2007.

[32] M. Crouhy, D. Galai, and R. Mark. The Essentials of Risk Management. McGraw-Hill, New York, NY, USA, 2006.

[33] R. R. Moeller. COSO Enterprise Risk Management: Understanding the New Integrated ERM Framework. John Wiley and Sons, Hoboken, NJ, USA, 2007.

[34] R. Mian, P. Martin, A. Brown, M. Zhang, in: S. Fiore, G. Aliosio (Eds.), Managing Data Intensive Workloads in the Cloud. Grid and Cloud Database Management, Springer Publishing, 2011, pp. 235–260.

[35] J. Schad, J. Dittrich, J. Quiane-Ruiz, Runtime measurements in the cloud: observing, analyzing and reducing variance, in: Proceedings of 36th International Conference on Very Large Databases, Singapore, September 2010.

[36] Lewis Chunningham, Oracle data architect, available at http://www.slideshare.net/jeetraj17/cloud-computing-it703-unit-1-2.

[37] H. Newman, I. Legrand, P. Galvez, R. Voicu, C. Cirstoiu, Monalisa: a distributed monitoring service architecture, 2003. Arxiv Preprint.

[38] S. Clayman, A. Galis, C. Chapman, G. Toffetti, L. Rodero-Merino, L.M. Vaquero, K. Nagin, B. Rochwerger, Monitoring service clouds in the future Internet, in: Towards the Future Internet—Emerging Trends from European Research, IOS Press, 2010, pp. 115–126.

[39] S. Clayman, A. Galis, L. Mamatas, Monitoring virtual networks with lattice, in: Network Operations and Management Symposium Workshops, NOMS Wksps, 2010 IEEE/IFIP, pp. 239–246.

[40] S. De Chaves, R. Uriarte, C. Westphall, toward architecture for monitoring private clouds, IEEE Communications Magazine 49 (2011) 130–137.

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 4, Issue 8, February 2015

[41] T. Kanstrén, R. Savola, Definition of core requirements and a reference architecture for a dependable, secure and adaptive distributed monitoring framework, in: 2010 Third International Conference on Dependability, DEPEND, pp. 154–163.

[42] S. Hongjie, F. Binxing, Z. Hongli, A distributed architecture for network performance measurement and evaluation system, in: Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies, PDCAT 2005, 2005, pp. 471–475.

[43] Y. Sun, Z. Xiao, D. Bao, J. Zhao, An architecture model of management and monitoring on cloud services resources, in: 2010 3rd International Conference on Advanced Computer Theory and Engineering, ICACTE, vol. 3, pp. V3-207–V3-211.

[44] A. Lahmadi, L. Andrey, O. Festor, Performance of network and service monitoring frameworks, in: IFIP/IEEE International Symposium on Integrated Network Management, 2009, IM'09, pp. 815–820..