International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 4 Issue: 2 263 - 267

Software Performance Engineering for Cloud Applications – A Survey

P.Ganesh[1], D EvangelinGeetha[2], TV Suresh Kumar[3]
1(Associate Professor , Dept. of MCA, BMSIT, ambikaganesh@gmail.com)
2(Professor, Dept. of MCA, MSRIT, devangelin@gmail.com)
3(Professor&HOD, Dept. of MCA, MSRITT, tvsureshkumar@msrit.edu)

Abstract—Cloud computing enables application service providers to lease their computing capabilities for deploying applications depending on user QoS (Quality of Service) requirements.Cloud applications have different composition, configuration and deployment requirements.Quantifying the performance of applications in Cloud computing environments is a challenging task. Software performance engineering(SPE) techniques enable us to assess performance requirements of software applications at the early stages of development. This assessment helps the developers to fine tune their design needs so that the targeted performance goals can be met. In this paper, we try to analyseperformance related issues of cloud applications and identify any SPE techniques currently available for cloud applications.

Keywords—*Software Performance Engineering(SPE), Service Level Agreement(SLA), Software Development Life Cycle (SDLC)*

*****

I. INTRODUCTION

Cloud computing is a new paradigm in which computing resources such as processing, memory, and storage are not physically present at the user's location. Instead, a service provider owns and manages these resources, and users access them via the Internet. Regarded as a paradigm shift to make software moreattractive as a service and shape the way IT hardware is provided [1], cloud computing moves from concept to realitywith staggering rapidity in recent years. The distinguishingfeature of cloud computing is to offer computing resources ina pay-as-you-go manner, which is expected to relieve theservice developers from the concerns about computingresource planning and management.

Before the emergence of cloud computing, one of themost challenging tasks for web application developers is capacity planning [2]. In order to guarantee the performanceof the web application, they need to determine how muchcomputing resources of each category (e.g., CPU, memoryand hard disk) should be purchased based on the design andimplementation of the web application and an estimation offuture workload. The former can be figured out by staticanalysis, testing and monitoring, but the latter is hard todetermine due to its unpredictability and variability, whichthus sometimes leads to bad performance of webapplications. In cloud computing, the developers do not needto do capacity planning any more. They purchase moderateresource from a cloud provider according to varyingworkload, and commit to an agreement, which specifies theirrequirement of service quality, with the cloud provider. The agreement is referred to as Service Level Agreement (SLA) [3]. Then, the job of allocating and manipulating resources ispassed to the cloud provider and becomes transparent to thedevelopers.

Cloud computingservices can be classified as follows.

1) *Infrastructure as a service (IaaS):* Users rent virtual serversfor their needs instead of buying real machines and software,and thus save the cost required for both equipmentand maintenance.

2) *Platform as a service (PaaS):* Service providers support anenvironment in which all software and runtime are readyfor use; program developers then simply upload theirdata, such as Web application codes and database data.

3) *Software as a service (SaaS):* Instead of buying software andinstalling it on a local machine, the software-as-a-servicemodel provides users with software-on-demand. Whenusers need to use applications, they use Web interfaceapplications or a providers program, and pay only forthe cost of the time they use it.

The development of efficient cloud applications inherits the challenges posed by the natural imbalance between computing, I/O and communication bandwidths of physical systems; these challenges are greatly amplified due to the scale of the system, its distributed nature, and by the fact that virtually all applications are data-intensive. Though cloud computing infrastructures attempt to automatically distribute and balance the load, the application developer is still left with the responsibility to place the data close to the processing site and to identify optimal storage for the data.Performance isolation is a critical aspect for the Quality of Service guarantees in shared environments. Indeed, if the run-time behavior of an application is affected by other applications running concurrently and thus, competing for CPU cycles, cache, main memory, disk and network access, it is rather difficult to predict the completion time; moreover, it is equally difficult to optimize the application [33].

One of the main advantages of cloud computing, the shared infrastructure, could also have a negative impact as perfect performance isolation is nearly impossible to reach in a real system, especially when the system is heavily loaded. The performance of virtual machines fluctuates based on the load, the infrastructure services, and the environment including the other users.Many applications consist of multiple stages; in turn, each stage may involve multiple instances running in parallel on the systems of the cloud and communicating among them. Thus, efficiency, consistency, and communication scalability are major concerns for an application developer. Indeed, due to shared networks and unknown topology, cloud infrastructures exhibit inter-node latency and bandwidth fluctuations which affect the application performance.

The data storage plays a critical role in the performance of any data-intensive application; the organization of the storage, the storage location, as well as, the storage bandwidth must be carefully analyzed to lead to an optimal application performance.Due to the complex architecture of clouds and the interaction between various applications sharing computing resources, it is much harder to enforce the SLA of each application. How to manage the resources in the cloud to have a balance between various applications and meet all the developers' requirements specified in SLAs, becomes a challenging problem for cloud providers. One common approach is to monitor the operation of the whole cloud, collect necessary runtime data, and adjust the resource provision manually. But here two new problems will arise in cloud environment. The first is the runtime monitoring data in cloud environment is too complex for human to comprehend and analysis. The second is that it is difficult to make an optimal adjustment for all applications. The cloud provider needs a performance model to help them make decisions, instead of just relying on the experience of system administrators.

Performance is a runtime attribute of a software system. Performance generally refers to system responsiveness, either the time required to respond to specific events, or number of events processed in a given time interval.The performance objectives are specified in three ways: response time, throughput, and constraints on resource usage. These objectives are used for the performance assessment. Performance responsiveness and scalability are make-or-break qualities for software. Poor performance costs the software industry, millions of dollars annually, in lost revenue, decreased productivity, increased development and hardware costs and damaged customer relations.

Nearly, everyone runs into performance problems at one time or another. Today's software development organizations are being asked to do more with less. In many cases, this means upgrading legacy applications to accommodate a new infrastructure, improve response time or throughput or both. Performance is an important but often neglected aspect of software development methodologies. Traditional software development methods have taken a "fix-it-later" approach to performance. "Fix-it-later" advocates concentrating on software correctness; performance considerations are deferred until the later phases of the software process (i.e., integration and testing). If performance problems are discovered at this point, the software is "tuned" to correct them and/or additional hardware is used. The 'fix-it-later' approach is clearly undesirable. Performance problems may, for example, be so severe that they require extensive changes to the system architecture. If these changes are made late in the development process, they can increase development costs, delay deployment, or adversely affect other desirable qualities of a design, such as understandability, maintainability, or reusability. Finally, designing for performance from the beginning produces systems that exhibit better performance than can be achieved using a "fix-it-later" approach. Performance analysis requires suitable descriptions of the software runtime behaviour (software dynamics).

Software Performance Engineering (SPE) is the process of predicting and evaluating based on performance models, whether the software system satisfies the user performance goals throughout the SDLC. It can be related to methodology for constructing software systems that meet performance goals. SPE includes techniques for gathering data, coping with uncertainty, constructing and evaluating performance models, evaluating alternatives and verifying models and validating results. It also includes strategies for the effective use of these techniques. SPE is a set of techniques that can be used in the software performance management. SPE can be integrated into any software development process, such as waterfall or iterative and incremental process.

Software Performance Engineering (SPE) has evolved over the past years and has been demonstrated to be effective during the development of many large systems [4]. Although the need for SPE is generally recognized by the industry, there is still a gap between the software development and the performance analysis domains. The main reasons for this gap has been the lack of a simple, inexpensive, and scalable techniques and tools for building performance models of complex, distributed and real-time software systems, and for solving these models to obtain the systems performance properties.

In the current practice, constructing performance models of complex systems is expensive to develop and validate. To construct performance models, analysts inspect, analyze and translate 'by hand' software specifications into models. Then solve these models under different workload factors, in order to diagnose performance problems and recommend design alternatives for performance improvement. This performance analysis cycle, when done properly starting at the early stages of design and continuing through all software development stages is time consuming, and thus expensive. Automated techniques are therefore needed to ease and accelerate the process of building and solving performance models.

Nowadays, cloud computing is an emerging technology and most of the developers are developing cloud applications. Performance analysis is a crucial process for cloud applications. Hence, we have reviewed the literature available in the area of SPE for cloud applications. In this paper, we have presented the observations made from the related literature.

II. REVIEW ON CLOUD PERFORMANCE

Survey on the literature available in the area of Cloud computing and SPE is carried out to identify the research issues still opening in these areas.

Cloud computing, a paradigm shift that offercomputing resources in a pay-as-you-go manner, is expected tominimize service operators' cost without sacrificing theperformance of services. However, it is much harder toguarantee the performance of cloud applications due to thecomplex architecture of cloud, the interaction between co-deployedapplications and the unpredictability of workload.Some research work has reported that existed cloud is notperforming as satisfactorily as expected [5]. How to guaranteethe stable performance of cloud applications turns

into a majorconcern of both the cloud providers and the service providers.

The work proposed in [6], present an approach to guarantee the performance of cloud applications based on a performance model. This modelis built by applying a series of machine learning algorithmson mass of data collected during the long-term execution of areal cloud. Then the model can be leveraged for futureperformance prediction and for resource allocationadjustment under a given performance requirement.

C-Meter, a portableframework for performance analyze of cloud computing environments is proposed in [7]. But this framework has many problems to beaddressed, such as job monitoring and controlling, andimportant issues like dynamic scheduling. D-Cloud [8] is asoftware testing environment using the cloud computingtechnology, which permits automatic configuration, testing withfault injection along the description of the testing scenario.The authors of [9] proposed Cloud-basedPerformance Testing System (CPTS) which is a portable,extensible and easy-to-use framework for generating andsubmitting test workloads to computing clouds.CPTS, is used to build elastic resource infrastructures and provide various kinds of testing services to testing users.

HaiboMi, Huaimin Wang et.al [10] proposes an approach called *Magnifier*to rapidly diagnose the source of performance degradation in large-scale non-stop cloud systems.InMagnifier, the execution path graph of a user request ismodeled by a hierarchical structure including componentlayer, module layer and function layer, and anomaliesare detected from higher layer to lower layer separately. Ji Ho Kim, Sang Min Lee et. al.[11] proposes the significance to analyze theeffect of failure in Cloud systems to performance. In this paper, they propose a method to analyze performability ofInfrastructure as a Service (IaaS) Cloud by introducing failuremode in CloudSim which is an event based simulator and doesnot take into account failure mode.

III. REVIEW ON SPE

The above mentioned works, for evaluating or testing the performance in the cloud environment, rely on runtime information and measured response times.Software Performance Engineering (SPE) provides capability to construct software and systems to meet performance objectives in every aspect of software development. The SPE approach proposed by Connie U. Smithwas the first methodology to integrate software performance analysis into the software engineering process, from the earliest stages to the end [12], [13]. It consists of two models: the software execution model and the system execution model. The first takes the form of Execution Graphs (EG) that represent the software execution behaviour; the second is based on QN models and represents the dynamic behaviour of the system, including hardware and software components. The SPE process requires additional information that includes software resource requirements for processing steps and computer configuration data. The analysis of the software model gives information about the resource requirements of the software system. The obtained results, together with information about the execution environment, are the input parameters of the system execution model.

An extension of the SPE approach was developed by Cortellessa and Mirandola in [14], [15]. The proposed methodology, called PRIMA-UML, makes use of information from different UML diagrams to incrementally generate a performance model representing the specified system. This model generation technique was initially proposed in [16]. In [17], Cortellessa et al. discuss the derivation of a LQN model from a software architecture specified by means of a Class diagram and a set of Sequence diagrams, generated using standard CASE tools. The approach clearly identifies all the supplementary information needed by the method to carry out the LQN derivation. A software performance framework for Model Driven Architecture (MDA) is given in [18].

An approach based on architectural patterns for client/server systems is presented in [19], [20], where Gomaa and Menasce investigate the design and performance modelling of component interconnection patterns, which define and encapsulate the way client and server components of SA communicate with each other via connectors. Menasce and Gomaa presented in [21], [22] an approach to the design and performance analysis of client/server systems. The approach introduced by Kahkipuro in [23], [24] is quite different from the others described in this section. The proposed framework consists of three different performance model representations and of the mappings among them. This approach proposes a UML-based performance modelling notation. The approach, proposed by Arief and Speirs in [25], presents a simulation framework named Simulation Modelling Language (SimML) to automatically generate a simulation Java program by means of the JavaSim tool, from the UML specification of a system that realizes a process oriented simulation model. The approach proposed by Balsamo et al for quantitative evaluation of the performance of the software systems is presented in [26], [27], [28]. They developed a prototype tool for automatic translation of the software model into a process-oriented simulation model.

HamzehKhazaeiet. al.[12], in their paper, describe an approximate analytical model for performance evaluation of cloud server farms and solve it to obtain accurateestimation of the complete probability distribution of the request response time and other important performance indicators. Aprocess model, Hybrid Performance Prediction Process (HP^3) model for modelling and evaluation of distributed systems during feasibility study and early phases of SDLC based on SPE is presented in [30]. Methodologies for assessing the performance during the feasibility study is discussed in [31],[32].

IV. OBSERVATIONS

From the above literature, the following open issues can be identified.

- Performance is unpredictable and is a major concern for cloud applications.
- Performance is a runtime attribute and hence to guarantee better performance, it is required to predict and evaluate performance throughout the SDLC.

- The performance assessment for cloud applications during the early stages of development is not discussed in the literature.
- Most of the software performance prediction methodologies are concentrating on modelling of software specifications, transformation techniques, and solving the performance models.
- Performance requirement data uncertainty is ignored in the methodologies. These performance requirement data are highly dynamic and uncertain, and gathering these data in early stages of development is difficult, in particular for cloud applications.Uncertainty in these data can be minimized by using suitable estimation techniques.

The below tables summarize the literature available on cloud application performance in particular and SPE approach to guarantee performance in general.

#	Author	Paper	Observation
1	M. Armbrust et al.	Above the clouds: A Berkeley view of cloud computing	Performance is unpredictable. Performance is a major obstacle for cloud computing.
2	Schad, J. Dittrich et al.	Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance	Existing cloud is not performing as satisfactorily as expected. Desirable to have SLAs based on performance
3	C. Binning et al.	How is the weather tomorrow? : Towards a Benchmark for the Cloud	Traditional benchmarks like TPC are not sufficient for analyzing the cloud services performance
4	Jin Shao et al.	A Performance Guarantee Approach for Cloud Applications Based on Monitoring	Based on the performance model, an approach to guarantee the cloud performance is presented

Table 1: Summary of papers wrt Cloud application performance

#	Author	Paper	Observation
1	C.U. Smith et al.	Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software.	Software performance engineering methodology at early stages of development introduced
2	Simonetta Balsamo et al.	Model based performance prediction in software development : A Survey	Performance is a runtime attribute of a software system. Model based approach is preferred for performance analysis
3	C.U. Smith et al.	Performance Engineering of Software Systems	Analytical model for performance evaluation of cloud server proposed
4	EvangelinGeetha D etal.	Framework for Hybrid Performance Prediction Process (HP3) Model: Use Case Performance Engineering Approach.	Process model for distributed systems developed

Table 2: Summary of papers wrt Software Performance Engineering

V. CONCLUSION

A process model for performance modeling and evaluation of cloud applications during the early phases of development is needed to be designed. Also, a framework for describing the elements of the process model are to be proposed. The framework that provides generic methodologies to implement the process model has to be proposed.

The main elements of the framework should be:

- Methodologies to gather data required for performance assessment in cloud environment.
- Techniques to model the software specifications with performance requirement data using UML models.
- Optimization techniques for resource allocation.
- Solving the models to obtain the performance metrics by simulation.

REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the clouds: A Berkeley view of cloud computing" *Computing*, vol. 53, 2009, pp. 07-013.

[2] D.A. Menascé and V.A.F. Almeida, *Capacity planning for Web services: metrics, models, and methods*, Prentice Hall, 2002.

[3] A. Chazalet, "Service Level Agreements Compliance Checking in the Cloud Computing: Architectural Pattern, Prototype, and Validation," *2010 Fifth International Conference on Software Engineering Advances*, IEEE, 2010, pp. 184-189.

[4] Connie U .Smith , Performance Engineering of Software Systems , Reading , MAAddision-Wesley, 1990

[5] J. Schad, J. Dittrich, and J.A. Quiane-Ruiz, "Runtime Measurements in the Cloud: Observing, Analyzing, and Reducing Variance," *Proceedings of the VLDB Endowment*, vol. 3, 2010, pp. 460-471.

[6] Jin Shao and Qianxiang Wang, "A Performance Guarantee Approach for Cloud Applications Based on Monitoring",

Proceedings of the 35th IEEE Annual Computer Software and Applications Conference Workshops, 2011.

[7] N. Yigitbasi, A.Iosup, D.Epema and S.Ostermann, "C-Meter: A Framework for Performance Analysis of Computing Clouds," in Cluster Computing and the Grid. Shanghai, pp. 472–477, June 2009.

[8] Takayuki Banzai, Hitoshi Koizumi, Ryo Kanbayashi, Takayuki Imada, Toshihiro Hanawa,andMitsuhisa Sato, "D-Cloud: Design of a software testing environment for reliable distributed systems using cloud computing technology," in IEEE/ACM International Conference on Cluster, Cloud and Grid Computing. Melbourne, Australia, pp. 631–636, June 2010.

[9] Li Zhang,YinghuiChen,FanTang,XiongAo, " Design and Implementation of Cloud-based Performance Testing System for Web Services", *Proceedings of the 6th International ICST Conference on Communications and Networking in China (CHINACOM),2011.*

[10] HaiboMi, Huaimin Wang, Gang Yin, HuaCai, Qi Zhou, Tingtao Sun, YangfanZhou,"Magnifier: Online Detection of Performance Problems in Large-Scale Cloud Computing Systems", *Proceedings of the 2011 IEEE International Conference on Services Computing.*

[11] Ji Ho Kim, Sang Min Lee, Dong Seong Kim, JongSouPark, "Performability Analysis of IaaS Cloud",*Proceedings of 2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing.*

[12] C.U. Smith, Performance Engineering of Software Systems. Addison Wesley, 1990.

[13] C.U. Smith and L.G. Williams, Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software. Addison Wesley, 2002.

[14] V. Cortellessa and R. Mirandola, "Deriving a Queueing Network Based Performance Model from UML Diagrams," ACM Proc. Int'l Workshop Software and Performance, pp. 58-70, 2000.

[15] Vittorio Cortellessa, RaffaelaMirandola, 'PRIMA-UML: A Performance Validation Incremental Methodology on Early UML Diagrams**',** Science of Computer Programming, Elsevier, 44, (1), 2002, pp. 101-129.

[16] V. Cortellessa, G. Iazeolla, and R. Mirandola, "Early Generation of Performance Models for Object-Oriented Systems," IEE Proc. - Software, vol. 147, no. 3 pp. 61-72, 2000.

[17] V. Cortellessa, A. D'Ambrogio, and G. Iazeolla, "Automatic Derivation of Software Performance Models from CASE Documents," Performance Evaluation, vol. 45, pp. 81-105, 2001.

[18] V Cortellessa, A.D. Marco, P. Inverardi, "Software Performacne Model-Driven Architecture", SAC '06, Dijon, France, April 23-27, 2006.

[19] H. Gomaa and D.A. Menasce´, "Design and Performance Modeling of Component Interconnection Patterns for Distributed Software Architectures," ACM Proc. Int'l Workshop Software and Performance, pp. 117-126, 2000.

[20] H. Gomaa and D. Menasce´, "Performance Engineering of Component-Based Distributed Software Systems," Performance Eng., R. Dumke et al., eds. pp. 40-55, 2001.

[21] D.A. Menasce´ and H. Gomaa, "On a Language Based Method for Software Performance Engineering of Client/Server Systems," ACM Proc. Int'l Workshop Software and Performance, pp. 63-69, 1998.

[22] D.A. Menasce´ and H. Gomaa, "A Method for Design and Performance Modeling of Client/Server Systems," IEEE Trans. Software Eng., vol. 26, no. 11, pp. 1066-1085, Nov. 2000.

[23] PekkaKahkipuro (2000) "Performance Modeling Framework for CORBA Based Distributed Systems", University of Helsinki, Finland. Report A-2000-3.

[24] P. Kahkipuro, "UML-Based Performance Modeling Framework for Component-Based Distributed Systems," Proc. Performance Eng. Conf., pp. 167-184, 2001.

[25] L.B. Arief and N.A. Speirs, "A UML Tool for an Automatic Generation of Simulation Programs," ACM Proc.Second Int'l Workshop Software and Performance, pp. 71-76, 2000.

[26] Simonetta Balsamo, Moreno Marzolla, Simulation-Based Performance Modeling of UML Software Architectures, Ph.D thesis, Ca' Foscari University of Venice, Italy.

[27] Simonetta Balsamo, Roberto Mamprin, Moreno Marzolla: 'Performance Evaluation of Software Architectures with Queuing Network Models', Proc. ESMc '04, Paris, France, Oct2004.http://www.moreno.marzolla.name/publications/papers/marzolla-esmc04-final.pdf, accessed May 2010.

[28] Simonetta Balsamo Moreno Marzolla: 'Performance Evaluation of UML Software Architectures with Multiclass Queueing Network Models', Proc. 5th International Workshop on Software and Performance, Spain, July, 2005, pp. 37 – 42.

[29] HamzehKhazaei, JelenaMisic and Vojislav B. Misic, "Performance Analysis of Cloud Computing Centers Using M/G/m/m+r Queuing Systems", IEEE Transactions on Parallel and Distributed systems, vol. 23, no. 5, May 2012, pp 936-943.

[30] EvangelinGeetha D, Suresh Kumar T V and RajaniKanth K. Framework forHybrid Performance Prediction Process (HP3) Model: Use Case PerformanceEngineering Approach. *ACM SIGSOFT Software Engineering Notes, May 2011Volume 36 Number 3, doi:10.1145/1968587.1968607.*

[31] EvangelinGeetha D, Suresh Kumar T V and RajaniKanth K. Predicting theSoftware Performance during Feasibility Study. *IET Softw. April 2011, Volume 5,Issue 2, pp.201–215.* doi:10.1049/iet-sen.2010.0075.

[32] EvangelinGeetha D, Suresh Kumar T V and RajaniKanth K. Determining suitable execution environment based on dynamic workload during early stages of software development life cycle: *a simulation approach. Int. J. Computational Science and Engineering, Inderscience, Vol. X, No. Y, 200X*

[33] Dan C. Marinescu, "Cloud Computing and Computer Clouds", Department of Electrical Engineering and Computer Science, University of Central Florida, Orlando, USA.

[34] Mouline, Imad. "Why Assumptions About Cloud Performance Can Be Dangerous." *Cloud Computing Journal*. May, 2009. www.cloudcomputing.sys-con.com/node/957492

[35] C. Binning, D. Kossmann, T krasa and S Loesing, "How is the weather tomorrow? : Towards a Benchmark for the Cloud", *In TestDB Workshop*, 2009.

[36] Simonetta Balsamo, Antinisca Di Marco, Paola Inverardi, Model based performance prediction in software development : A Survey,*IEEE Transactions on software engineering*, Vo. 30, No.5, May 2004.