SOFTWARE: PRACTICE AND EXPERIENCE
Softw. Pract. Exper. 2014; **44**:771–775
Published online 20 May 2014 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/spe.2266

EDITORIAL

A note on software tools and techniques for monitoring and prediction of cloud services

Rajiv Ranjan*,†, Rajkumar Buyya, Philipp Leitner, Armin Haller and Stefan Tai

[1]*Senior Research Scientist and CSIRO Julius Fellow, CSIRO CCI Division, GPO Box 664, Canberra, ACT 2601*
[2]*Director, Cloud Computing and Distributed Systems (CLOUDS) Lab, Department of Computing and Information Systems, The University of Melbourne, Australia*
[3]*University of Zurich, Department of Informatics, Binzmühlestrasse 14, CH-8050 Zurich, Switzerland*
[4]*Research Scientist, CSIRO CCI Division, GPO Box 664, Canberra, ACT 2601*
[5]*Institute AIFB - Building 11.40, Karlsruhe Institute of Technology, D-76128 Karlsruhe*

1. INTRODUCTION

Cloud computing is the latest computing paradigm that transparently delivers Information and Communication Technology resources as services, freeing the users of Cloud applications from dealing with low-level implementation and system administration details. Cloud provides the promise of on-demand access to affordable large-scale computing (e.g., multi-core CPUs, GPUs, and clusters of GPUs), storage (such as disks), and software (e.g., databases, application servers, and data processing frameworks) resources without substantial up-front investment. Cloud resources are hosted in large datacenters, often referred to as virtualized data farms, operated by companies such as Amazon, Apple, GoGrid, and Microsoft.

While the growing ubiquity of Cloud computing is having a significant impact in many applications domains, there are still significant problems that exist with regard to efficient provisioning and delivery of applications using its Information and Communication Technology resources. These barriers are due to resource uncertainties [1] that have degradable effect on the run-time Quality of Service (e.g., access latency and number of requests being successfully served per second) of software applications deployed in the Cloud. There are many reasons for such uncertainties including (i) unpredictable application workload types (enterprise, scientific, and streaming big data analytics), (ii) fluctuations in resource capacity demands (i.e., bandwidth, memory, storage, and CPU), (iii) abrupt failures (e.g., failure of a network link), (iv) stochastic access patterns (e.g., number of end-users and their geo-location), (v) heterogeneity in device types (e.g., mobile phone, laptop, and smart TV), (v) heterogeneous resource types and their providers, and (vi) heterogeneity in data types (3D images, videos, audios, text, etc.) and network types (e.g., wired and wireless).

These Cloud resource uncertainties need to be managed optimally to maintain contractual requirements defined in Service-Level Agreements (SLAs) that underlie most Cloud computing contracts. Basically, SLAs are legal documents (paper and/or electronic) that encode the nature and scope of QoS parameters (e.g., ensure availability 99.99% and ensure web application server latency to be less than 100 ms). To tackle uncertainties, recent research and industry efforts [2] have

*Correspondence to: Dr. Rajiv Ranjan, Senior Research Scientist and CSIRO Julius Fellow. CSIRO CCI Division, GPO Box 664, Canberra, ACT 2601
†E-mail: Raj.Ranjan@csiro.au

focused on developing monitoring techniques and frameworks that can assist cloud providers and application owners in (i) keeping their resources and applications operating at peak efficiency, (ii) detecting variations in resource and application performance, (iii) accounting the SLA violations of certain QoS parameters, and (iv) tracking the leave and join operations of cloud resources due to failures and other dynamic configuration changes.

The rest of this editorial note is organized as follows: Section 2 gives a brief overview of the research and development work carried out for monitoring application QoS over cloud resources; Section 3 summarizes the research contributions that were accepted for this special issue; Section 3 concludes the paper with some future remarks.

2. RELATED WORKS IN RESOURCE AND APPLICATION SERVICE MONITORING LANDSCAPE: FROM COMPUTATIONAL GRIDS TO CLOUD DATACENTERS

In last 20 years, a large body of research has focused on developing tools and techniques for monitoring the QoS status of resources and applications over distributed systems (e.g., grids, clusters, and clouds). Some QoS monitoring techniques have been investigated and implemented in computational grids, such as Network Weather Service (NWS) [3], which monitors the network and computing resource QoS and periodically forecast the QoS in a future arrival of a application workload. The current version of NWS gathers the operating system level metrics such as available CPU percentage, available non-paged memory, and TCP/IP Performance. Other monitoring tools [4, 5] that were popular in grid and cluster computing era included R-GMA, Hawkeye, Ganglia, MDS-I, and MDS-II. Aforementioned monitoring techniques and tools were designed for managing static system configuration, where numbers of hardware and software resource types were assumed to remain constant over lifecycle of an application. In other words, these tools did not consider the issue of auto scaling and de-scaling primitives supported by virtualized cloud resources. These tools were only concerned about monitoring the QoS parameters for the hardware resources (CPU, storage, and network), while being completely agnostic to application-specific QoS parameters and SLA requirements. The performance of these tools was optimized for monitoring the QoS of only one type of application (e.g., high performance computing application). On the other hand, in cloud computing datacenters, multiple application instances can be multiplexed and co-allocated on single physical resource. Clearly, the monitoring tools developed in grid and cluster computing era (while being innovative and useful) is not suitable to tackle the challenges on cloud computing environments and hosted application types.

Current cloud resource and application QoS monitoring frameworks (e.g., Amazon CloudWatch [6], Azure Fabric Controller) typically monitor the entire virtual machine (VM, a software implementation of a physical CPU resource) as a black box and lacks ability to inter-operate across cloud datacenters managed by different providers (e.g., Amazon, Microsoft, GoGrid, and CA). This means that QoS of software resources (e.g., web server, and database server) contained in the application stack is not properly monitored and managed. While frameworks such as Monitis [7] and Nimsoft [8] overcome the aforementioned limitations of CloudWatch and Fabric Controller, they lack ability to monitor and enforce application-specific QoS requirements. Further, all of the aforementioned frameworks lack ability to predict and detect faults before they occur.

Some of the recent research works [9] have also focused on applying large-scale data and pattern mining to the QoS monitoring history and event log data. Authors in [10] evaluated the prediction capability of Support Vector Machine, Neural Network, and Linear Regression techniques for learning the QoS behavior of cloud hosted applications. To predict the CPU usage of VMs, authors in [11] applied Markov Chain model. Authors in [12] applied prediction techniques such as Moving Average, Auto Regression, Neural Networks, Support Vector Machines, and Gene Expression Programming for predictive VM QoS monitoring and provisioning. Most of these techniques focused on monitoring and predicting QoS of VMs rather than individual application components. Further, these approaches did not reason about the interplay of QoS parameters and SLA requirements across multiple layers (software as a service, platform as a service, and infrastructure as a service) of cloud application stack.

3. SUMMARY OF CONTRIBUTIONS

In this special issue, we present seven articles that tackle several aspects of the aforementioned resource uncertainties for monitoring QoS of applications hosted on Cloud resources. In particular, Ryckbosch and Diwan propose a Temporal Pattern Analyzer system in their paper [13] *Analyzing Performance Traces Using Temporal Formulas* that uses formulas in linear-temporal logic extended with variables to analyze traces to investigate long-tail performance problems at Google and reduce the manual labor involved in analyzing traces. The technique is applied on user request logs, which contain events at each stage of processing of a user request to Gmail. The authors show that the system can scale to large traces, a prerequisite considering that Gmail produces a million or more events a second. Two of the case studies presented in the paper have directly contributed to improving the performance of Google.

Cao *et al.* also use execution trace information, in this case, CPU load traces and propose [14] a novel method for *CPU load prediction for cloud environment based on a dynamic ensemble model* to obtain better performances. The ensemble model proposed consists of two layers, a predictor optimization layer that can continuously incorporate new predictor instances and remove those ones with a poor performance and an ensemble layer that is responsible for producing the final prediction based on the results of multiple predictor instances.

The four papers are all concerned with monitoring Cloud applications, ranging from a model and language to define design-time adaption techniques in the paper [15] by Inzinger *et al.* on a *Generic Event-Based Monitoring and Adaptation Methodology for Heterogeneous Distributed Systems*, to better visualization techniques in the monitoring process in the paper [16] *A Novel Monitoring Mechanism by Event Trigger for Hadoop System Performance Analysis* by Chang *et al.*, to adapting to failed application service in a distributed environment by introducing fault avoidance service that can be called instead of the failed service by Gülcü *et al.* in their paper [17] *Fault Masking as a Service*, to a feature-based high availability mechanism that monitors data streams for a quantile feature in the paper [18] by Ding *et al.* on a *Feature-based High Availability Mechanism for Quantile Tasks in Real-time Data Stream Processing*.

In particular, Inzinger *et al.* present [15] a novel domain-specific language termed MONINA that allows specification of system components and their monitoring and adaptation-relevant behavior for controlling Cloud systems. The authors propose a mechanism for optimal deployment of the defined control operators onto available computing resources by monitoring the cloud environment with complex-event processing queries and adapt to problems by condition action rules performed on top of a distributed knowledge base.

Chang *et al.* propose [16] a system called Event Trigger that provides an automatic recording mechanism on the Hadoop Cloud Computing system, to check the system performance at every static time interval, and compares the variation. The performance parameters are collected during the system monitoring process and are applied onto an easy-understandable visual graph for users to adjust the hardware deployment in order to refine the Hadoop system.

Gülcü *et al.* propose [17] an approach to prevent the occurrence of errors that result from the unavailability of partner services in the first place. They introduce a fault avoidance service to which composite services can register at will. After registration, this fault avoidance service periodically checks the partner links, detects unavailable partner services, and updates the composite service with available alternatives. Thus, in case of a partner service error, the composite service will have been updated before attempting an ill-destined request.

Ding *et al.* focus [18] on the monitoring of data streams on the quantile tasks, a typical summary-oriented operation for aggregation, and propose a feature-based high availability mechanism to reduce related overhead and latency. With the help of a monitor module, the quantile feature is maintained incrementally through histogram synopsis over a time-based sliding window. Consequently, failed tasks can be recovered precisely with a high probability in an efficient way.

Finally, the special issue is rounded off by a paper [19] on *Design and Implementation of Task Scheduling Strategies for Massive Remote Sensing Data Processing Across Multiple Data Centers* by Zhang *et al.* that proposes scheduling strategies for data processing workflows. In particular, they propose scheduling strategies in massive remote sensing data processing to reduce the total task execution time. The authors divided the data processing workflows into two categories, namely, Bag of Tasks applications that consist of a large number of independent tasks and Direct Acyclic Graph applications that contain a large number of interdependent tasks. They propose two strategies to deal with issues in either of the two categories, a Partitioning Group based on Hypergraph algorithm that partitions data into several groups to minimize the amount of sharing data transferring and an Optimized Task Tree strategy to find the key workflow path, which would be endowed with a high priority in the execution.

This special issue presents, through these seven papers, several techniques that can dynamically predict and capture the relationship between an application performance targets, current hardware resource allocation, and changes in workload patterns, in order to adjust resource configuration at design-time and run-time. More work in this area is rapidly emerging, further improving the availability of massively distributed Cloud applications. This will further improve the economies of scale of Cloud applications making Cloud Computing an even more compelling paradigm in comparison to traditional in-house hosted applications.

4. CONCLUSION

Application QoS monitoring will continue to remain an important research area for cloud-based systems. More tangible efforts are needed for developing monitoring tools and techniques that can specify, reason, and monitor QoS related to a variety of application (enterprise, scientific, and streaming big data analytics) and cloud datacenter types (private and public). Further, research should also aim to correlate events with data from many different sources (e.g., holiday schedules, job schedules, and trends from social media about application usage sentiment) in order to predict how external events can impact an application QoS. In this special issue, we have selected research papers that aim to address some of these challenges. We hope that the readers will find the articles of this special issue informative and useful.

REFERENCES

1. Schad J, Dittrich J, Quiané-Ruiz J. Runtime measurements in the cloud: observing, analyzing, and reducing variance. *Proceedings of the VLDB Endowment* 2010; **3**(1-2): 460–471.
2. Alhamazani K, Ranjan R, Mitra K, Jayaraman P, Rabhi F, Khan SU, Guabtni A, Bhatnagar V. An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art. *Springer Journal of Computing*, Elsevier, 2014.
3. Wolski R, Spring N, Hayes J. The network weather service: a distributed resource performance forecasting service for metacomputing. *Future Generation Computer Systems* 1999; **15**(5-6): 757–768.
4. Zhang X, Freschl J, Schopf JM. A performance study of monitoring and information services for distributed systems. In *Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing (HPDC '03)*. IEEE Computer Society: Washington, DC, USA, 2003; 270–281.
5. Zanikolas S, Sakellariou R. A taxonomy of grid monitoring systems. *Future Generation Computer Systems* 2005; **21**(1):163–188.
6. Amazon Cloud Watch, 2014. Available at: http://aws.amazon.com/cloudwatch/.
7. Monitis, 2014. Available at: http:// portal.monitis.com/.
8. Nimsoft, 2014. http://www.nimsoft.com/solutions/nimsoft-monitor/cloud.
9. Hormozi E, Hormozi H, Akbari MK, Javan MS. Using of machine learning into cloud Environment (A Survey): managing and scheduling of resources in cloud systems. *2012 Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, 2012; 363–368.
10. Ajila SA, Bankole AA. Cloud client prediction models using machine learning techniques. *37th IEEE Annual Computer Software and Applications Conference (COMPSAC)*, IEEE Computer Society, 2013; 134–142.
11. Mallick S, Hains G, Deme CS. A resource prediction model for virtualization servers. *2012 International Conference on High Performance Computing and Simulation (HPCS)*, IEEE Computer Society, 2012; 667–671.

12. Jiang Y, Perng C, Li T, Chang R. ASAP: a self-adaptive prediction system for instant cloud resource demand provisioning. *2011 IEEE 11th International Conference on Data Mining (ICDM)*, IEEE computer society, 2011; 1104–1109.
13. Ryckbosch F, Diwan A. Analyzing performance traces using temporal formulas. *Journal of Software: Practice and Experience* 2014; **44**(7):777–792. DOI: 10.1002/spe.2256.
14. Cao J, Fu J, Li M, Chen J. CPU load prediction for cloud environment based on a dynamic ensemble model. *Software: Practice and Experience* 2013; **44**(7):793–804. DOI: 10.1002/spe.2231.
15. Inzinger C, Hummer W, Satzger B, Leitner P, Dustdar S. Generic event-based monitoring and adaptation methodology for heterogeneous distributed systems. *Journal of Software: Practice and Experience* 2014; **44**(7):805–822. DOI: 10.1002/spe.2254.
16. Liao C-S, Chuang C-P, Chang R-S. A novel monitoring mechanism by event trigger for Hadoop system performance analysis. *Journal of Software: Practice and Experience* 2013; **44**(7):823–834. DOI: 10.1002/spe.2230.
17. Gülcü K, Sözer H, Aktemur B, Ercan AÖ. Fault masking as a service. *Journal of Software: Practice and Experience* 2014; **44**(7):835–854. DOI: 10.1002/spe.2255.
18. Ding W, Han Y, Wang J, Zhao Z. Feature-based high-availability mechanism for quantile tasks in real-time data stream processing. *Journal of Software: Practice and Experience* 2013; **44**(7):855–871. DOI: 10.1002/spe.2244.
19. Zhang W, Wang L, Ma Y, Liu D. Design and implementation of task scheduling strategies for massive remote sensing data processing across multiple data centers. *Journal of Software: Practice and Experience* 2013; **44**(7):873–886. DOI: 10.1002/spe.2229.