Future Generation Computer Systems 28 (2012) 1017–1029

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Towards autonomic detection of SLA violations in Cloud infrastructures

Vincent C. Emeakaroha [a,*], Marco A.S. Netto [b], Rodrigo N. Calheiros [c], Ivona Brandic [a], Rajkumar Buyya [c], César A.F. De Rose [b]

[a] Vienna University of Technology, Vienna, Austria

[b] Faculty of Informatics, Catholic University of Rio Grande do Sul, Porto Alegre, Brazil

[c] CLOUDS Laboratory, Department of Computer Science and Software Engineering, The University of Melbourne, Australia

ARTICLE INFO

Article history:
Received 9 November 2010
Received in revised form
28 April 2011
Accepted 11 August 2011
Available online 6 November 2011

Keywords:
Service level agreement
Resource monitoring
SLA violation detection
SLA enactment
Cloud architecture

ABSTRACT

Cloud computing has become a popular paradigm for implementing scalable computing infrastructures provided on-demand on a case-by-case basis. Self-manageable Cloud infrastructures are required in order to comply with users' requirements defined by Service Level Agreements (SLAs) and to minimize user interactions with the computing environment. Thus, adequate SLA monitoring strategies and timely detection of possible SLA violations represent challenging research issues. This paper presents the Detecting SLA Violation infrastructure (DeSVi) architecture, sensing SLA violations through sophisticated resource monitoring. Based on the user requests, DeSVi allocates computing resources for a requested service and arranges its deployment on a virtualized environment. Resources are monitored using a novel framework capable of mapping low-level resource metrics (e.g., host up and down time) to user-defined SLAs (e.g., service availability). The detection of possible SLA violations relies on the predefined service level objectives and utilization of knowledge databases to manage and prevent such violations. We evaluate the DeSVi architecture using two application scenarios: (i) image rendering applications based on ray-tracing, and (ii) transactional web applications based on the well-known TPC-W benchmark. These applications exhibit heterogeneous workloads for investigating optimal monitoring interval of SLA parameters. The achieved results show that our architecture is able to monitor and detect SLA violations. The architecture output also provides a guideline on the appropriate monitoring intervals for applications depending on their resource consumption behavior.

1. Introduction

Cloud computing represents a novel paradigm for the implementation of scalable computing infrastructures combining concepts from virtualization, distributed application design, Grid, and enterprise IT management [1–3]. Service provisioning in the Cloud relies on Service Level Agreements (SLAs) representing a contract signed between the customer and the service provider including non-functional requirements of the service specified as Quality of Service (QoS) [4,5]. SLA considers obligations, service pricing, and penalties in case of agreement violations.

Flexible and reliable management of SLA agreements is of paramount importance for both Cloud providers and consumers. On the one hand, prevention of SLA violations avoids penalties providers have to pay and on the other hand, based on flexible and timely reactions to possible SLA violations, user interaction with the system can be minimized, which enables Cloud computing to take roots as a flexible and reliable form of on-demand computing.

Although, there is a large body of work considering development of flexible and self-manageable Cloud computing infrastructures [6–8], there is still a lack of adequate monitoring infrastructures able to predict possible SLA violations. Most of the available monitoring systems rely either on Grid [9,10] or service-oriented infrastructures [11], which are not directly compatible to Clouds due to the difference of resource usage model, or due to heavily network-oriented monitoring infrastructures [12]. In Grids [13] resources are mostly owned by different individuals/enterprises, and in some cases, as desktop Grids for instance, resources are only available for usage when the owners are not using them [14]. Therefore, resource availability varies much and this impacts its usage for application provisioning, whereas in Cloud computing, resources are owned by an enterprise (Cloud provider), provisioning them to customers in a pay-as-you-go manner. Therefore, availability of resources is more stable and resources can be provisioned on-demand. Hence, the monitoring strategies used for detection of SLA violations in Grids cannot be directly applied to Clouds.

Furthermore, another important aspect for the usage of SLAs is the required elasticity of Cloud infrastructures. Thus, SLAs are not only used to provide guarantees to end user, they are also used by

* Corresponding author.
E-mail address: vincent@infosys.tuwien.ac.at (V.C. Emeakaroha).

0167-739X/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2011.08.018

providers to efficiently manage Cloud infrastructures, considering competing priorities like energy efficiency and attainment of SLA agreements [15,16] while delivering sufficient elasticity. Moreover, SLAs are also recently used as part of novel Cloud engineering models like Cloud federation [17,18] where provider can in- or outsource their infrastructure depending on the current load. Thus, since SLA parameters are usually defined by Cloud providers and can comprise various user-defined attributes, current monitoring infrastructures lack appropriate solutions for adequate SLA monitoring. The first challenge is to facilitate mapping of measured metrics by low level tools to application based SLAs. The second challenge is to determine appropriate monitoring intervals at the application level keeping the balance between the early detection of possible SLA violations and system intrusiveness of the monitoring tools.

In this paper we present the novel concept for mapping low-level resource metrics to high-level SLAs—*LoM2HiS* [19], where system metrics (e.g., system up and down time) are translated to high-level SLAs (e.g., system availability). Thus, LoM2HiS facilitates efficient monitoring of Cloud infrastructures and early detection of possible SLA violations. Furthermore, LoM2HiS framework enables user-driven mappings between the resource metric and SLA parameters by utilizing mapping rules defined with Domain Specific Languages (DSLs). However, determination of optimal measurement intervals of low-level metrics and their translation to SLAs is still an open research issue. Short measurement intervals may negatively affect the overall system performance, whereas long measurement intervals may cause heavy SLA violations.

In order to assist Cloud providers in detecting SLA violations through resource monitoring, we developed the DeSVi architecture [20]. This architecture represents a core step towards achieving flexible and autonomic SLA management. The main components of the DeSVi architecture are: (i) *the automatic VM deployer*, (ii) *application deployer*, and (iii) *the LoM2HiS framework*. Based on user requests, the *automatic VM deployer* allocates necessary resources for the requested service and arranges its deployment on a virtual machine (VM). After service deployment, LoM2HiS framework monitors the VMs and translates the low-level metrics into high-level SLAs using the specified mapping rules. To realize autonomic SLA management DeSVi utilizes a knowledge database for the evaluation of the monitored information in order to propose reactive actions in case of SLA violation situations.

The main contributions of the paper are: (i) definition of the motivation scenario for the development of the architecture aimed at detecting SLA violations, (ii) conceptual design of the *DeSVi* architecture for the prediction of SLA violations, (iii) discussion of the implementation choices for the DeSVi, and (iv) extensive evaluation of the architecture in a real computing infrastructure using various SLA parameters and two Cloud applications: an image rendering service based on POV-Ray[1] and the TPC-W transactional web e-Commerce benchmark.[2]

The rest of this paper is organized as follows: Section 2 presents the related work. Section 3 presents the architecture for the autonomic management of Cloud services and the motivating scenario for the development of the *DeSVi* architecture. Section 4 introduces the *DeSVi* architecture. In particular we discuss the automatic VM deployer, application deployer, and the monitoring components. Section 5 discusses our implementation choices, whereas Section 6 discusses experimental evaluation of the *DeSVi* architecture. Section 7 presents our conclusions and describes future work.

[1] http://www.povray.org.
[2] http://www.tpc.org/tpcw/.

2. Related work

We classify related work into (i) resource monitoring [21,12,22], (ii) SLA management including violation detection [23–27], and (iii) mapping techniques of monitored metrics to SLA parameters [28,11]. Currently, there is little work in the area of resource monitoring, low-level metrics mapping, and SLA violation detection in Cloud computing. Because of that, we look into the related areas of Grid and Service-Oriented Architecture (SOA) based systems.

Fu et al. [21] propose GridEye, a service-oriented monitoring system with flexible architecture that is further equipped with an algorithm for prediction of the overall resource performance characteristics. The authors discuss how resources are monitored with their approach in Grid environment but they consider neither SLA management nor low-level metric mapping. Gunter et al. [12] present NetLogger, a distributed monitoring system, which can monitor and collect information of networks. Applications invoke NetLogger's API to survey the overload before and after some request or operation. However, it monitors only network resources. Wood et al. [22] developed a system, called Sandpiper, which automates the process of monitoring and detecting hotspots and remapping/reconfiguring VMs whenever necessary. Their monitoring system is reminiscent of our in terms of goal: avoid SLA violation. Similar to our approach, Sandpiper uses thresholds to check whether SLAs can be violated. However, it differs from our system by not considering the mapping of low level metrics, such as CPU and memory, to high-level SLA parameters, such as response time for SLA enactment.

Boniface et al. [23] discuss dynamic service provisioning using GRIA SLAs. The authors describe provisioning of services based on agreed SLAs and the management of the SLAs to avoid violations. Their approach considers only Grid environments and not Clouds. Moreover, they do not detail how the low-level metric are monitored and mapped to high-level SLAs to enforce the SLA objectives at runtime. Koller and Schubert [24] discuss autonomous QoS management using a proxy-like approach. Their implementation is based on WS-Agreement. Thereby, SLAs can be exploited to define certain QoS parameters that a service has to maintain during its interaction with a specific customer. However, their approach is limited to Web services and does not consider other applications types. Frutos and Kotsiopoulos [25] discuss the main approach of the EU project BREIN [29] to develop a framework that extends the characteristics of computational Grids by driving their usage inside new target areas in the business domain for advanced SLA management. BREIN applies SLA management to Grids, whereas we target SLA management in Clouds. Dobson and Sanchez-Macian [27] present a unified QoS ontology applicable to QoS-based Web services selection, QoS monitoring, and QoS adaptation. However they do not consider application deployment and provisioning strategies. Comuzzi et al. [26] define the process for SLA establishment adopted within the EU project SLA@SOI framework. The authors propose the architecture for monitoring SLAs considering two requirements introduced by SLA establishment: the availability of historical data for evaluating SLA offers and the assessment of the capability to monitor the terms in an SLA offer. But they do not consider monitoring of low-level metrics and mapping them to high-level SLA parameters for ensuring the SLA objectives.

Rosenberg et al. [28] deal with QoS attributes for Web services. They identify important QoS attributes and their composition from resource metrics. They present mapping techniques for composing QoS attributes from resource metrics to form SLA parameters for a specific domain. However, they do not deal with monitoring of resource metrics. D'Ambrogio and Bocciarelli [11] introduce a model-driven approach for integrating performance prediction