Managing Service Quality at the Platform and Application Levels with rSLA

Samir Tata*, Mohamed Mohamed*, Obinna Anya*, Takashi Sakairi†
Nagapramod Mandagere*, Heiko Ludwig*, and Nathalie Baracaldo*
*IBM Research - Almaden, 650 Harry Road, San Jose, CA 95120
†IBM Research - Tokyo, Japan
Email: [stata,mmohamed,obanya,pramod,hludwig,baracald]@us.ibm.com, sakairi@jp.ibm.com

***Abstract*—Managing service quality in heterogeneous Cloud environments is complex: different providers expose different management interfaces for monitoring and configuration actions that can occur at the infrastructure, platform and application levels. To manage Service Level Agreements (SLAs) in this context, we have developed the rSLA Manager for enabling fast setup of SLA monitoring in dynamic, heterogeneous Cloud environments. In this paper, we demonstrate the use of the rSLA Manager to manage the response time behavior of an image scaling service on the IBM Bluemix platform. We show how the system enables SLA violation reduction by facilitating platform-level adaptation (e.g., adding application instances) and application-level behavior configuration (e.g., changing image scaling quality) based on different service level objectives as well as current application configuration and workload.**

I. Introduction

Managing service quality in heterogeneous Cloud environments is complex: different Cloud providers expose different management interfaces, both semantically and syntactically, for monitoring and configuration actions. Furthermore, performance adaptation can take place on the infrastructure, platform, or application level, providing an application service operator different choices depending on the provider's business objectives and commitments to service clients.

This presents tremendous challenges to SLA management in Cloud computing. Previous efforts have leveraged existing frameworks, such as Web services, e.g., WSLA, Grid services, e.g., WS-Agreement and related Grid scheduler implementations, and Web performance management. Largely, those management frameworks were focused on managing the performance of a particular platform or a service provider implementing a specific layer in the stack. Today's Cloud environments span multiple layers and typically expose heterogeneous interfaces. Cloud standards such as OCCI and associated quality definitions such as CSLA help in addressing interface heterogeneity and definition of service level objectives (SLOs); however, they fail to enable a complete control loop.

We present the rSLA Manager, an SLA service management tool that enables fast setup of SLA monitoring in dynamic, heterogeneous Cloud environments. It connects to Cloud service and application instrumentation using an adapter layer that exposes uniform monitoring information to the service that can then be aggregated for application-specific use. This demo shows the use of the rSLA Manager to specify complex Cloud quality objectives and management behavior in a formal document. In particular, we demonstrate how to specify the adaptation of both application and Platform-as-a-Service (PaaS) behavior to meet quality of service goals, thereby enabling cross-layer quality management.

II. rSLA Framework

The rSLA Manager is based on the rSLA Framework presented in [1]. It is made up of three main components: the rSLA language to formally represent SLAs, the rSLA Service, which interprets the SLAs and implements the behavior specified in them, and a set of Xlets - lightweight, dynamically bound adapters to monitoring and control action interfaces as a generic REST API, abstracting from the heterogeneity of different service interfaces.

The rSLA Manager provides users with the capability for managing the SLA lifecycle, including uploading, activating, deactivating, and deleting SLAs, and dashboards for viewing SLO compliance and action executions. During upload, the rSLA service leverages the Ruby meta-interpretation framework to achieve fast deployment and interpretation of an SLA document defining service level objectives and management behavior. During activation, the rSLA service invokes monitoring Xlets to commence collection of new observations for relevant base metrics, the rSLA Scheduler invokes specific rSLA APIs for SLO evaluation and action executions. Deactivation suspends the schedules for a SLA, while delete kills the SLA. In this demo, we illustrate these functionalities using a simple image scaling application, and highlight the typical elements of SLA management, including the use of application-level and platform-level service quality management.

III. Demo Scenarios

To illustrate the rSLA concepts, we consider an example Cloud application for image scaling. Given an image input, the application fits it within a width and height provided. The scaling methods are: SPEED, BALANCED, QUALITY and ULTRA_QUALITY. The best scaling quality is the one obtained using the ULTRA_QUALITY method, then the one obtained using the QUALITY method and so on. The ULTRA_QUALITY method requires more processing time than the QUALITY method which requires more processing time than the BALANCED method and so on.

978-1-5090-1663-1/16 $31.00 © 2016 IEEE
DOI 10.1109/WETICE.2016.69

The scaling application is provided with two interfaces: a user interface and an administrator interface. The user interface offers one operation for scaling images. Its inputs are an image and desired dimensions (width and height). Its output is a resized image. The administrator interface offers operations to get observations on the response time of previous scaling operations or to manage the scaling method used by increasing the current quality, for example from SPEED to QUALITY, or assigning a precise quality, for example ULTRA_QUALITY.

A. Experiment

The rSLA service and its components such as the scheduler and Xlets reside on the IBM Bluemix service, which is an implementation of the Cloud Foundry PaaS open source system. The image scaler application is deployed on Bluemix. The application is used to generate the image scaling workload, simulating the image service's clients. The demo setup is illustrated in Figure 1. The rSLA service in this demo uses three Xlets: one for monitoring quality and response time basic metrics and executing configuration action related to image scaling methods; one for executing configuration actions for scaling up and down the application instances; and one for reporting monitoring observations to the tenant. The demo is particularly focused on the following questions: What are the results of SLO evaluations? What and why are the configuration actions executed? And finally, is the observed behavior what we expect?

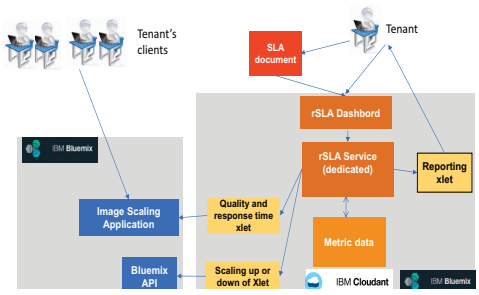

Fig. 1. rSLA Demo setup

To create workloads that cause response time variation, we considered three scenarios, where the scaling method is initialized to BALANCED. 1) Running a single tenant's client client with 100 sequential queries for image scaling, 2) Running of four parallel tenant's clients, each client runs 10 sequential queries for image scaling, and 3) Running of 80 parallel tenant's clients, where each client runs 10 sequential queries for image scaling.

B. Results and Analysis

In scenario 1, the amount of parallel queries submitted for image scaling is smaller than in scenario 2, which in turn is smaller than in scenario 3. We expect in scenario 1 that the response time rate is high (more that 90%) and the image scaling quality method will be improved (the increase_quality action executed). In scenario 2, we expect that the SLOs will be met (the response time rate is between 70 and 90% and the quality is not low) and none of the configuration actions will be executed. In scenario 3, we expect that the response time rate will be not acceptable (the rate is less that 70%) and the quality is low. Consequently the action scale_up will be executed so that the response time rate will be improved.

Figure 2 shows screen shots of the rSLA Manager user interface, a) shows the SLA lifecycle management interface, b) shows a graphical trend of observations for the response time metric, and c) summary of SLO compliance.

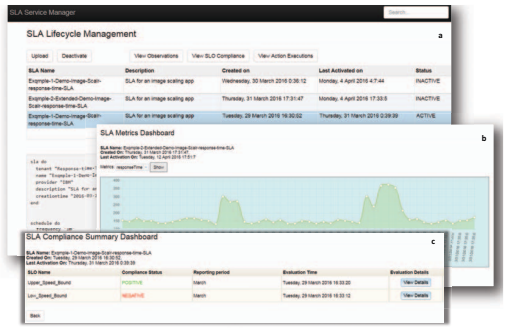

Fig. 2. SLA Management Dashboard

The observations made in our demo are measured according to frequencies defined in the schedules of the quality and response time metrics. SLOs can also be associated with a schedule that defines when to evaluate the precondition and objective expressions. For our scenarios we considered a frequency of 20s for base metrics measurement and 20s for evaluating SLOs. Based on the timestamps, we observe that the precision of metrics measurement and SLO evaluation was accurate. This was possible because in our scenario, the rSLA Manager handled a single SLA. Nevertheless, we think that the precision depends on variances of the scheduler's and the rSLA Service's load. Hence, it is important that the scheduler and the rSLA Service are scaled according to their load to ensure accurate observations and SLO evaluations.

IV. Concluding Remarks

The rSLA Manager facilitates management of SLAs specified in the rSLA language through their life cycle and viewing of compliance results. It enables dynamic setup of service quality management in a short time frame at low cost. The Xlet approach enables the rSLA service to address the issues of service monitoring and configuration in complex and heterogeneous Cloud environments. This demo illustrates that the accuracy of quality monitoring depends on the scalability of the monitoring system, implemented as a set of scalable PaaS applications, allowing for combined use of application domain and technical adaptation for service quality management.

References

[1] Heiko Ludwig, Katerina Stamou, Mohamed Mohamed, NagaPramod Mandagere, Bryan Langston, Gabriel Alatorre, Hiroaki Nakamura, Obinna Anya, and Alexander Keller. rSLA: Monitoring SLAs in Dynamic Service Environments. ICSOC, Springer, LNCS vol. 9435, pp. 139-153, 2015.