Optimization of Cloud-Based Applications Using Multi-site QoS Information

Hong Thai Tran[1(✉)] and George Feuerlicht[1,2,3]

[1] Faculty of Engineering and Information Technology, University of Technology, Sydney, Ultimo, Australia
{hongthai.tran, george.feuerlicht}@uts.edu.au
[2] Unicorn College, V Kapslovně 2767/2, 130 00 Prague 3, Czech Republic
[3] Department of Information Technology, University of Economics, Prague, W. Churchill Sq. 4, Prague 3, Czech Republic

Abstract. With rapid increase of the use of cloud services, the availability of Quality of Service (QoS) information is becoming of utmost importance to assist application managers in selection of suitable services for their enterprise applications. Due to different characteristics of cloud and on-premise services, monitoring and management of cloud-based enterprise applications requires a different approach that involves the monitoring of QoS parameters such as availability and response time in different geographic locations. In this paper, we propose a multi-site model for the monitoring and optimization of cloud-based enterprise applications that evaluates the availability and response time of cloud services concurrently across different geographic locations. Our preliminary results using eWay and PayPal payment services monitored in eleven sites across four geographic regions indicate that location-based information can be used to improve the reliability and performance of cloud-based enterprise applications.

1 Introduction

SOA (Service Oriented Architecture) is evolving towards a more flexible, dynamically scalable cloud-based computing architecture for enterprise applications. Typically, multiple cloud and on-premise services are composed using different protocols and integration methods to provide the required enterprise application functionality. As cloud services are sourced from different cloud providers their QoS (Quality of Service) characteristics can substantially differ depending on the geographical location and on the provider cloud infrastructure. While most cloud service providers publish QoS information on their websites, it often does not accurately reflect the values measured at the consumer site as the performance of cloud services is impacted by numerous factors that include dynamic changes in network bandwidth and topology and transmission channel interference [1]. Additionally, changes in provider internal architecture and method of service delivery can significantly impact on QoS characteristics of cloud services. Consequently, consumer monitoring and optimization of the runtime behaviour of cloud services has become critically important for the management of enterprise applications [2].

A.M. Tjoa et al. (Eds.): CONFENIS 2016, LNBIP 268, pp. 325–338, 2016.
DOI: 10.1007/978-3-319-49944-4_24

Service monitoring is a run-time activity that involves recording the values of response time, availability and other non-functional service parameters in order to enable predictive analysis and proactive service management. Service monitoring and service management in cloud computing environments presents a particular challenge to application administrators as the enterprise application is dependent on the performance and availability of third-party cloud services. The traditional approach to QoS monitoring is based on continuously sending test messages to critical services to check their availability and performance. This approach is not suitable for the monitoring of cloud services as it increases service costs and generates unnecessary data traffic.

Monitoring and optimization of QoS of cloud services presents an important and challenging research problem. Although some research work on monitoring of QoS characteristics of cloud services is available in the literature, there is currently lack of detailed information about the assessment of run-time behaviour of cloud services that includes location-based QoS information [3], making informed decisions about the selection and composition of cloud services difficult in practice [4, 5].

In our earlier work we have described the features of the Service Consumer Framework (SCF) designed to improve the reliability of cloud-based enterprise applications by managing service outages and service evolution. We have implemented and experimentally evaluated availability and response time characteristics of payment services (PayPal and eWay) using three separate reliability strategies (Retry Fault Tolerance, Recovery Block Fault Tolerance, and Dynamic Sequential Fault Tolerance) and compared these experimental results with theoretically predicted values [6].

In this paper we extend this work by focusing on improving the estimates of availability and response time of cloud services by introducing location-based QoS information. We monitor QoS characteristics of eWay and PayPal services across eleven locations in four geographical regions to obtain a more accurate estimate of response time and availability for specific deployment locations of consumer enterprise application. We collect the QoS information independently of the information published by cloud service providers by recording payment transaction log data in a monitoring database. In the next section (Sect. 2) we review related literature dealing with monitoring the performance of cloud-based services, and in Sect. 3 we discuss service optimization using multi-site monitoring. Section 4 describes our experimental setup for multi-site monitoring of cloud services and gives experimental results of availability and response time for eWay and PayPal payment services measured at eleven geographic locations. Section 5 contains our conclusions and proposals for future work.

2 Related Work

Optimization techniques to improve reliability and performance of enterprise applications that include fault prevention and forecasting have been the subject of research interest for a number of years [7]. Such techniques have been recently adapted for web services and cloud-based enterprise applications. Using redundancy-based fault tolerance strategies, Zibin and Lyu [8] propose a distributed replication strategy evaluation and selection framework for fault tolerant web services. Authors compare various

replication strategies and propose a replication strategy selection algorithm. Adams et al. [9] describe fundamental reliability concepts and a reliability design-time process for organizations, providing guidelines for IT architects to mitigate potential failures of cloud-based applications.

Developing reliable cloud-based applications involves a number of new challenges, as enterprise applications are no longer under the full of control of local developers and administrators. In response to such challenges, Zibin et al. [10] present a FTCloud component ranking framework for fault-tolerant cloud applications. Using structure-based component ranking and hybrid component ranking algorithms, authors identify the most critical components of cloud applications and then determine an optimal fault-tolerance strategy for these components. Based on this work, Reddy and Nalini [11] propose the FT2R2Cloud framework as a fault tolerant solution using time-out and retransmission of requests for cloud applications. FT2R2Cloud measures the reliability of software components in terms of the number of responses and throughput. Authors propose an algorithm to rank software components based on reliability as calculated using number of service outages and service invocations over a period of time.

Other authors focus on QoS optimization, for example Deng and Xing [12] proposed a QoS-oriented optimization model for service selection. This approach involves developing a *lightweight* QoS model, which defines functionality, performance, cost, and trust as QoS parameters of a service. Authors have verified the validity of the model by simulation of cases that show the effectiveness of service selection based on these QoS parameters. Leitner et al. [13] formalize the problem of finding an optimal set of adaptations, which minimizes the total cost arising from Service Level Agreement (SLA) violations and the cost of preventing the violations. Authors present possible algorithms to solve this complex optimization problem, and describe an end-to-end approach based on the PREvent (Prediction and Prevention based on Event monitoring) framework. They discuss experimental results that show how the application of their approach leads to reduced service provider costs and explain the circumstances in which different algorithms lead to satisfactory results. Other authors have focused on predicting future QoS values using service performance history records. Wenmin et al. [1] present a history record-based service optimization method, called HireSome that aims at enhancing the reliability of service composition plans. The method takes advantage of service QoS history records collected by the consumer, avoiding the use of QoS values recorded by the service provider. Authors use a case study of a multimedia delivery application to validate their method. Lee et al. [14] present a QoS management framework that is used to quantitatively measure QoS and to analytically plan and allocate resources. In this model, end users quality preferences are considered when system resources are apportioned across multiple applications, ensuring that the net end-user benefit is maximized. Using semantically based techniques to automatically optimize service delivery, Fallon and O'Sullivan [15] introduce the Semantic Service Analysis and Optimization (AESOP) approach and a Service Experience and Context COllection (SECCO) framework. The AESOP knowledge base models the end-user service management domain in a manner that is aware of the temporal properties of the services. The autonomic AESOP Engine runs efficient semantic algorithms that implement the Monitor, Analyze, Plan, and Execute (MAPE) functions using temporal properties to operate on small partitioned subsets of the

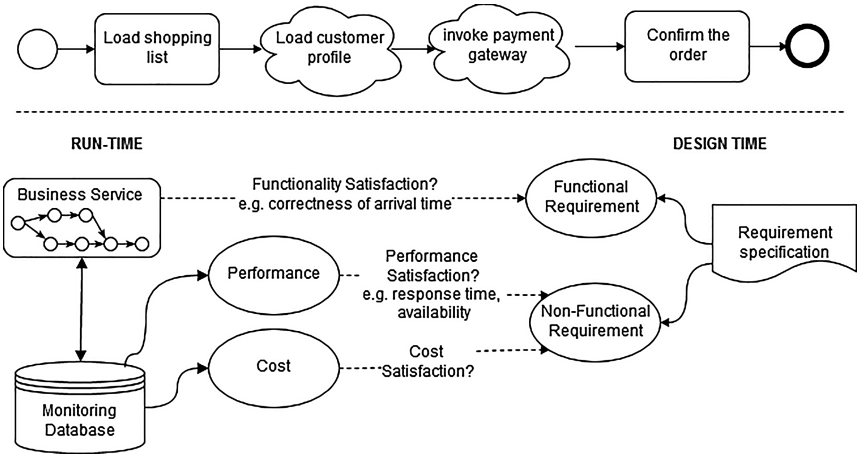

Fig. 1. Online shopping check out optimization scenario

knowledge base. A case study is used to demonstrate that AESOP is also applicable in the Mobile Broadband Access domain.

So far only a very limited attention has been paid to using location-based QoS information for the optimization of cloud-based enterprise applications.

3 Service Optimization Using Multi-site Monitoring

Service optimization is concerned with continuous service improvement and aims to optimize performance and cost of business services. Consider, for example, the situation illustrated in Fig. 1 that shows an Online Shopping Check Out service that includes a cloud-based payment gateway. At design time, the service consumer needs to select a suitable payment service to integrate into the business workflow ensuring that both the functional and non-functional requirements are satisfied. Making this selection decision requires the knowledge of QoS parameters at the site where the enterprise application is deployed.

Typically, both the service provider and service consumer perform service monitoring independently, and both parties are responsible for resolving service quality issues that may arise. Service providers maintain transactions logs and make these logs available to service consumers who can use this information to calculate service costs and to estimate service QoS. Provider QoS data is collected continuously at the provider site irrespective of any connectivity issues and includes information about planned and unplanned outages. However, the QoS values published by service providers may not accurately reflect the values measured at the service consumer site as QoS depends on the deployment location of the enterprise application and is affected by the quality of the network connection, provider location, and service configuration. With some global cloud service providers, the actual location from which the service is delivered may not be known to service consumers, making it difficult to optimize the

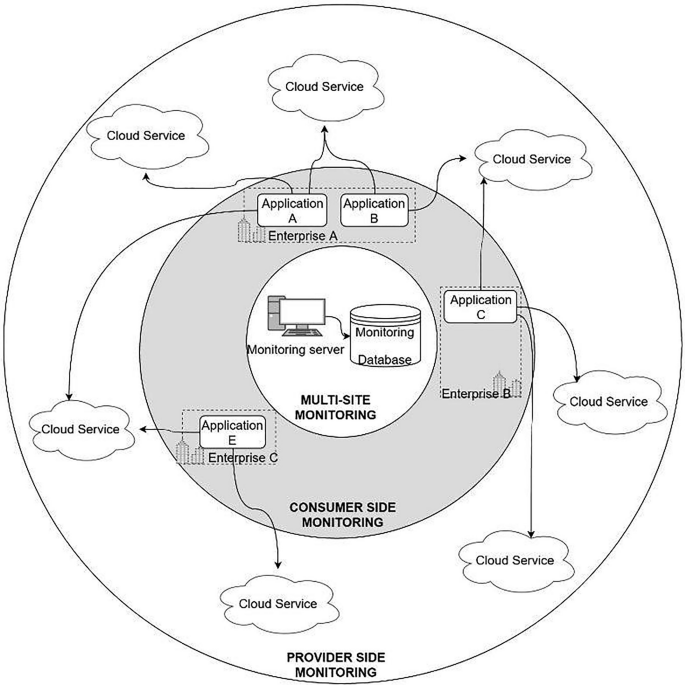

Fig. 2. Multi-site cloud service monitoring

performance of the enterprise application based on QoS values published by the provider. The QoS values measured at the consumer site are impacted by connectivity issues, and while these values may not fully reflect provider site QoS measurements they are important indicators of enterprise application performance. Multi-site monitoring can be used to overcome the limitations of single-site (provider or consumer) QoS monitoring by mapping the behaviour of cloud services across different sites and geographical regions. We argue that in order to fully optimize cloud service selection and deployment and to ensure that the non-functional requirements are met at run-time, the service consumer needs to know the runtime QoS values of cloud services as measured in different geographic locations. To accomplish this, we propose a model that uses a centralized monitoring database to collect service QoS data from multiple service consumer locations and making this data available for analysis by service consumers (Fig. 2). This can be achieved by collaboration among different service consumers who record their local monitoring data in a global QoS database and share this information with other consumers of cloud services. The implementation of such a shared QoS monitoring database would enable accurate real-time QoS analysis and real-time notifications of QoS issues. Runtime performance information (i.e. response time, availability and various types of error messages) recorded in the database can be used by application administrators to monitor service utilization, plan maintenance activities, and to perform statistical analysis of response time and throughput for individual cloud services.

3.1 Enterprise Application Optimization Strategies

Optimization of enterprise applications that use cloud services may involve a number of different strategies that range from using alternative cloud services to migrating the servers that run the application to a different cloud infrastructure. With increasing availability of alternative cloud services with equivalent functionality, service consumers can chose services to use in their enterprise applications based on the cost and QoS characteristics. This may involve deployment of a new version of an existing service or replacement of the service with an alternative from a different provider, if the original service becomes obsolete or too costly. Service consumers can also optimise application performance by re-locating the application to a different cloud infrastructure selecting a more suitable geographic location, taking into account both end-user connectivity and connectivity to third-party cloud services. Finally, QoS characteristics of cloud-based enterprise applications can be improved by using various reliability strategies, re-configuring cloud services to provide higher levels of fault tolerance [6]. These fault tolerance strategies include Retry Fault Tolerance (RFT), Recovery Block Fault Tolerance (RBFT) and Dynamic Sequential Fault Tolerance (DFST) strategies. Using RFT strategy, cloud services are repeatedly invoked following a delay period until the service invocation succeeds. RFT helps to improve reliability, in particular in situations characterized by short-term outages. The RBFT strategy relies on service substitution using alternative services invoked in a specified sequence. This *failover* configuration includes a primary cloud service used as a default (active) service, and stand-by services that are deployed in the event of the failure of the primary service, or when the primary service becomes unavailable because of scheduled/unscheduled maintenance. The DFST strategy is a combination of the RFT and RBFT strategies that deploys an alternative service when the primary service fails following RFT retries [16]. The choice of an optimal strategy for the deployment of cloud services must be based on in-depth knowledge of QoS characteristics including their dependence on the geographical location.

4 Experimental Setup for Multi-site Monitoring

In order to evaluate the proposed location-based QoS approach to optimization of cloud-based enterprise applications we have implemented an experimental multi-site monitoring environment for two payment services: PayPal Pilot service (pilot-payflowpro.paypal.com) and eWay Sandbox (https://api.sandbox.ewaypayments.com). The QoS data was collected using Amazon Elastic Compute Cloud (AWS EC2) servers deployed at eleven sites (Mumbai, Seoul, Singapore, Sydney, Tokyo, Frankfurt, Ireland, Sao Paulo, California, Oregon and Virginia) across four different geographic regions (Asia Pacific, Europe, South America and the US). The monitoring database was implemented using Microsoft SQL Server Amazon Relational Database (AWS RDB). The QoS data was collected in each site by monitoring payment transactions and removing private data such as customer information before recording the information in the monitoring database. Simulating over 200,000 payment transactions initiated by 300 users, payment services were invoked using the SCF (Service Consumer

Framework) payment service adaptor that logs the service name, location, start time, end time, result, and error code for each payment transaction [17].

The payment service response time for a transaction (T_T) was calculated as:

$$T_T = \mathrm{T}_E - \mathrm{T}_s \tag{1}$$

where T_E is the *end time* of transaction and T_S is the *start time* of a transaction, and the average response time (T_S) of a service was calculated as:

$$T_s = \frac{\sum_1^n \mathrm{T}_T}{n} \tag{2}$$

where n is number of transactions, and T_T is response time of a transaction in Eq. (1). Similarly, an inactive time or *downtime* of a service (T_I) is calculated as:

$$T_I = \mathrm{T}_{IS} - \mathrm{T}_{As} \tag{3}$$

where T_{IS} is the *start time* of a failed transaction and T_{As} is the start time of the next successful transaction. Then, the availability of a service (A_s) is calculated as:

$$D = \mathrm{T}_{LE} - \mathrm{T}_{FS} \tag{4}$$

$$PF_s = \frac{\sum T_I}{D} \tag{5}$$

$$A_S = 1 - PF_S \tag{6}$$

where D is the duration of test period that is calculated using the *end time* of last transaction (T_{LE}) and the *start time* of first transaction (T_{FS}). PF_s is probability of failure of a service, and T_I is a downtime in Eq. (3) and A_S is the availability of a service.

Table 1 shows the response time of eWay and Paypal payment services as measured in different geographical locations over the monitored period 20th to 28th August 2016. The table shows that the response time of the eWay service is better (in most cases less than half) than the response time of the PayPal service, while the availability of both services is approximately the same. Both response time and availability are influenced by two major factors: provider QoS characteristics and the reliability of the network connection. In order to optimize the consumer side QoS characteristics it is important to identify which of these factors plays a dominant role. If network connectivity is the dominant factor that impacts on service quality, then using the RFT fault tolerance strategy described in Sect. 3.1 above may improve consumer side QoS, but only for situations characterized by short-term outages or latency fluctuations. When network connectivity suffers from long-term outages, the solution may involve migrating the service to a different cloud infrastructure in a different geographical location. However, if network connectivity is not a dominant factor and QoS degradation is caused by provider related issues, then RBFT and DFST fault tolerant strategies may provide a solution by substituting alternative services at runtime.

Table 1. QoS data for eWay and PayPal payment services

Region	Location	Service	Number of transaction	Number of fails	Average response time (s)	Availability
Asia Pacific	Mumbai	eWay	8328	52	1.48	99.37%
		PayPal	8328	43	3.37	99.47%
	Seoul	eWay	9599	55	1.25	99.42%
		PayPal	9601	53	2.71	99.44%
	Singapore	eWay	8900	57	1.33	99.35%
		PayPal	8900	52	2.97	99.41%
	Sydney	eWay	9530	57	0.95	99.40%
		PayPal	9531	53	2.73	99.44%
	Tokyo	eWay	9635	60	1.17	99.37%
		PayPal	9636	57	2.73	99.41%
Europe	Frankfurt	eWay	9092	52	1.59	99.42%
		PayPal	9091	49	2.76	99.45%
	Ireland	eWay	9322	56	1.56	99.40%
		PayPal	9323	56	2.86	99.39%
South America	Sao Paulo	eWay	8790	51	1.58	99.41%
		PayPal	8790	46	2.94	99.47%
US	California	eWay	11029	72	1.26	99.34%
		PayPal	11029	59	2.00	99.46%
	Oregon	eWay	16538	101	1.18	99.39%
		PayPal	16538	93	2.17	99.43%
	Virginia	eWay	10310	61	1.39	99.40%
		PayPal	10310	61	2.37	99.41%

In order to differentiate between network connectivity and cloud service provider issues we analyse the level of dependence between QoS parameters for the two payment services (eWay and PayPal) at each location by calculating the correlation coefficient for response time and availability. High level of dependence indicates that both payment services fail or suffer from increased response time at the same time, identifying network connectivity as the main source of the problem. Low levels of correlation indicate independent modes of failure for the two payment services, pointing to the service provider as the cause of QoS fluctuations.

Table 2 shows the values of correlation coefficients of eWay and PayPal payment services calculated for different locations. The correlation coefficient $C_{(T_e,T_p)}$ [18] of response time between eWay and PayPal services is calculated as:

$$C_{(T_e,T_p)} = \frac{\sum (T_e - \overline{T_e})(T_p - \overline{T_p})}{\sqrt{\sum (T_e - \overline{T_e})^2 \sum (T_p - \overline{T_p})^2}} \tag{7}$$

Table 2. Response time and availability correlation coefficients for eWay and PayPal

Region	Location	Response time	Availability
Asia Pacific	Mumbai	−0.0317	−0.0229
	Seoul	0.0896	−0.0861
	Singapore	0.0027	0.0868
	Sydney	0.0947	0.0577
	Tokyo	−0.0867	0.1273
Europe	Frankfurt	−0.1168	−0.2012
	Ireland	−0.0169	0.0536
South America	Sao Paulo	0.0648	0.0415
US	California	0.0924	−0.0612
	Oregon	−0.0331	−0.0275
	Virginia	−0.0523	0.0137

where T_e is response time for eWay transaction, T_p is the response time for a concurrent PayPal transaction, $\overline{T_e}$ is the average response time of the eWay service and $\overline{T_p}$ is the average response time of the PayPal service. The correlation coefficient $C_{(T_e,T_p)}$ for the availability of eWay and PayPal is calculated as:

$$C_{(A_e,A_p)} = \frac{\sum (A_e - \overline{A_e})(A_p - \overline{A_p})}{\sqrt{\sum (A_e - \overline{A_e})^2 \sum (A_p - \overline{A_p})^2}} \tag{8}$$

where A_e is average availability of the eWay service, A_p is average availability of the PayPal service computed for one hour, $\overline{A_e}$ is the average availability of eWay service during the monitoring period, and $\overline{A_p}$ is the average availability of PayPal service during monitoring period.

It is evident from the low correlation coefficient values in Table 2 that the underlying factors affecting response time and availability of the two payment services are mutually independent over the monitored period. As the two payment services share the same network connections, this indicates that the source of QoS variability is the service provider system, rather than the network. This implies that improved QoS values may be achievable by deploying RBFT and DFST service substitution fault tolerant strategies [17]. We also note that in an environments characterized by reliable low latency network connectivity the QoS values observed at the service consumer site will approximate those published by the service provider.

Figures 3 and 4 show the hourly average response time and availability values for eWay and PayPal services during the monitored period between 20th and 28th August 2016 for eleven geographic locations across the globe. Figure 3 shows that the response time of eWay services is generally better than for PayPal and that the response time of PayPal deployed in the US and Europe is better than those deployed in Asia Pacific. Figure 4 shows that the availability of both services varies from 98.8 to 99.8% with PayPal availability slightly better than that of eWay.

Fig. 3. Hourly average response times of eWay and PayPal services (20th to 28th August 2016)

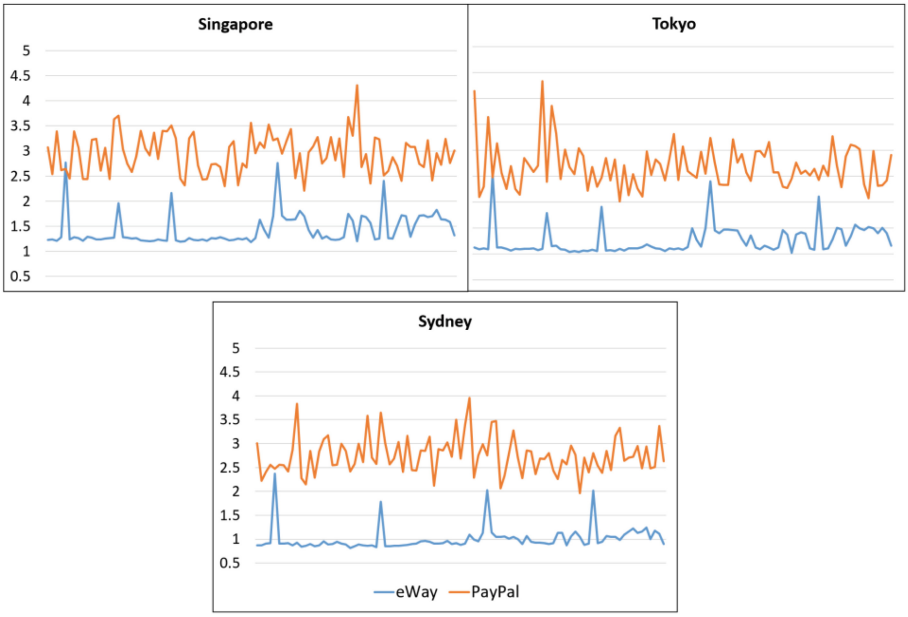

Fig. 3. (continued)

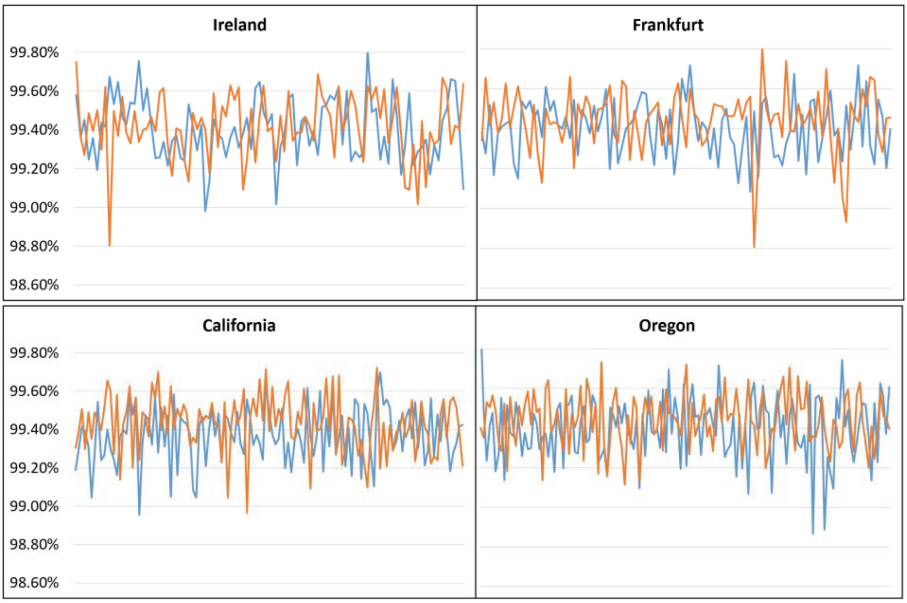

Fig. 4. Hourly average availability of eWay and PayPal services (20th to 28th August 2016)

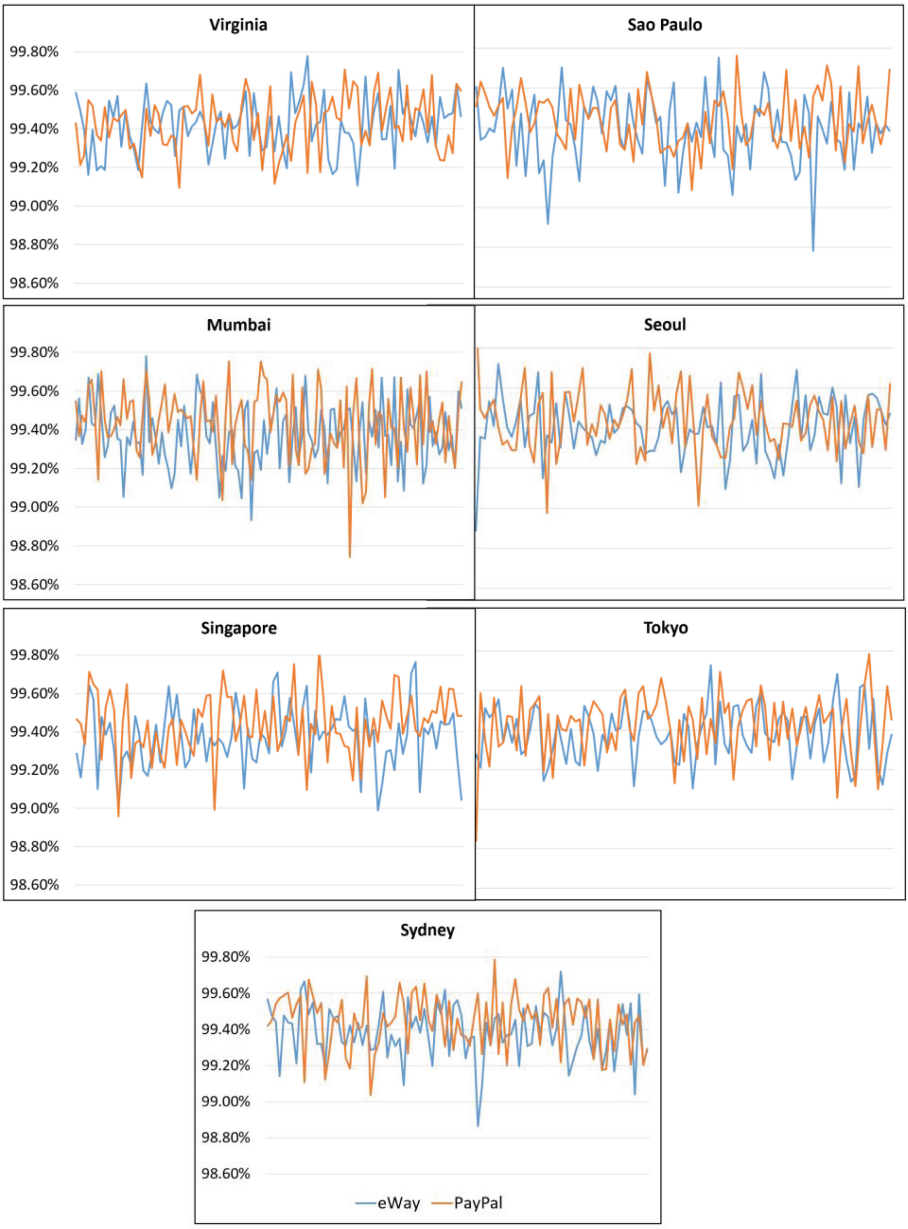

Fig. 4. (continued)

5 Conclusion

In this paper we have argued that consumer-side monitoring of QoS characteristics of cloud services is essential to enable service consumers to make informed decisions about service selection at design-time, and to maintain good run-time performance of cloud-based enterprise applications. Service consumers need to supplement QoS information published by cloud providers with data obtained independently using consumer side monitoring taking into account location-based information, as the QoS values measured at the consumer deployment site (i.e. at the site where the enterprise application is running) may vary from those published by cloud service providers.

Our results obtained using AWS (Amazon Web Services) platforms deployed in eleven sites across four geographic regions to monitor eWay and PayPal payment services indicate that both services achieved availability values above 99.9% during most of the measurement period 20th to 28th August 2016. It is evident from the low correlation coefficient values that the underlying factors affecting response time and availability of the two payment services are mutually independent. As the two payment services share the same network connections, this indicates that the source of QoS variability is the service provider system, rather than the network. This implies that improved QoS values may be achievable by deploying RBFT and DFST service substitution fault tolerant strategies. Using a combination of QoS information published by cloud service providers and QoS data measured at different geographic locations by service consumers, improves the understanding of performance and reliability trade-offs and can facilitate the selection of more effective optimization strategies.

In our future work we plan to collect QoS data over an extended period of time to give more reliable estimates of service availability and response time. We also plan to make our monitoring database publicly available to cloud service consumers to enable sharing of QoS information and to promote a collaborative effort with the aim to improve the accessibility of cloud QoS information.

References

1. Wenmin, L., Wanchun, D., Xiangfeng, L., Chen, J.: A history record-based service optimization method for QoS-aware service composition. In: 2011 IEEE International Conference on Web Services (ICWS) (2011)
2. Safy, F.Z., El-Ramly, M., Salah, A.: Runtime monitoring of SOA applications: importance, implementations and challenges. In: 2013 IEEE 7th International Symposium on Service Oriented System Engineering (SOSE) (2013)
3. Aceto, G., Botta, A., De Donato, W., Pescapè, A.: Cloud monitoring: a survey. Comput. Netw. **57**(9), 2093–2115 (2013)
4. Lu, W., Hu, X., Wang, S., Li, X.: A multi-criteria QoS-aware trust service composition algorithm in cloud computing environments. Int. J. Grid Distrib. Comput. **7**(1), 77–88 (2014)
5. Noor, T.H., Sheng, Q.Z., Ngu, A.H., Dustdar, S.: Analysis of web-scale cloud services. IEEE Internet Comput. **18**(4), 55–61 (2014)

6. Tran, H.T., Feuerlicht, G.: Improving reliability of cloud-based applications. In: Aiello, M., Johnsen, E.B., Dustdar, S., Georgievski, I. (eds.) ESOCC 2016. LNCS, vol. 9846, pp. 235–247. Springer, Heidelberg (2016). doi:10.1007/978-3-319-44482-6_15
7. Tsai, W.T., Zhou, X., Chen, Y., Bai, X.: On testing and evaluating service-oriented software. Computer **41**(8), 40–46 (2008)
8. Zibin, Z., Lyu, M.R.: A distributed replication strategy evaluation and selection framework for fault tolerant web services. In: IEEE International Conference on Web Services, ICWS 2008 (2008)
9. Adams, M., Bearly, S., Bills, D., Foy, S., Li, M., Rains, T., Ray, M., Rogers, D., Simorjay, F., Suthers, S., Wescott, J.: An introduction to designing reliable cloud services. Microsoft Trustworthy Computing (2014). https://www.microsoft.com/en-au/download/details.aspx?id=34683
10. Zibin, Z., Zhou, T.C., Lyu, M.R., King, I.: Component ranking for fault-tolerant cloud applications. IEEE Trans. Serv. Comput. **5**(4), 540–550 (2012)
11. Reddy, C.M., Nalini, N.: FT2R2Cloud: Fault tolerance using time-out and retransmission of requests for cloud applications. In: 2014 International Conference on Advances in Electronics, Computers and Communications (ICAECC) (2014)
12. Deng, X., Xing, C.: A QoS-oriented optimization model for web service group. In: 8th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2009. IEEE (2009)
13. Leitner, P., Hummer, W., Dustdar, S.: Cost-based optimization of service compositions. IEEE Trans. Serv. Comput. **6**(2), 239–251 (2013)
14. Lee, C., Lehoezky, J., Rajkumar, R., Siewiorek, D.: On quality of service optimization with discrete QoS options. In: Proceedings of 5th IEEE Real-Time Technology and Applications Symposium. IEEE (1999)
15. Fallon, L., O'Sullivan, D.: The AESOP approach for semantic-based end-user service optimization. IEEE Trans. Netw. Serv. Manag. **11**(2), 220–234 (2014)
16. Zheng, Z., Lyu, M.R.: Selecting an optimal fault tolerance strategy for reliable service-oriented systems with local and global constraints. IEEE Trans. Comput. **64**(1), 219–232 (2015)
17. Feuerlicht, G., Tran, H.T.: Service consumer framework: managing service evolution from a consumer perspective. In: 16th International Conference on Enterprise Information Systems, ICEIS-2014. Springer, Portugal (2014)
18. Microsoft: CORREL function (2016). https://support.office.com/en-us/article/CORREL-function-995dcef7-0c0a-4bed-a3fb-239d7b68ca92. Cited 22 Aug 2016