Proceedings Paper:
Webster, D, Townend, P and Xu, J (2014) Restructuring web service interfaces to support evolution. In: Proceedings - IEEE 8th International Symposium on Service Oriented System Engineering, SOSE 2014. Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on, 7-11 April 2014, Oxford, UK. IEEE Computer Society , 158 - 159. ISBN 9781479925049

https://doi.org/10.1109/SOSE.2014.66

Restructuring Web Service Interfaces to support Evolution

David Webster, Paul Townend, Jie Xu
School of Computing,
University of Leeds
Leeds, UK
{D.E.Webster, P.M.Townend, J.Xu}@leeds.ac.uk

Abstract— This paper presents an overview of a scheme (RES-WS) to enable Web Service providers to be able to evolve their service interface in a non-backwards compatible way and still maintain compatibility with existing consumers. The need for a non-passive approach to evolving services while still remaining backwards compatible is highlighted and presented based on a clear need identified within the literature. Based on graph rewriting theory, the set of refactorings was formally represented as transformations through rewriting rules and enables a set of preconditions to be defined for each transformation. A demonstrator has been developed to implement the RES-WS scheme as a message mediator which interprets the chain of primitive refactoring transformations required to carry out a particular complex transformation. Experimental validation was performed to demonstrate the feasibility and effectiveness of the message mediator for an upgraded Web Service against its QoS constraint.

Keywords-Web Services, Restructuring, Refactoring

I. INTRODUCTION

Just as traditional software evolves over time, there is a need to evolve Web Services and their interfaces similarly. Evolution of services must be undertaken in a controlled manner to avoid an interruption in the ability for a consumer to be able to interact with the service and contributes towards their overall dependability.

Whilst passive strategies for maintaining backwards compatibility are considered best practice at the current time [1,2], there are still challenges for situations when the service provider aims to make an incompatible change for various reasons, including dealing with an interface becoming too complex or due to the maintenance cost of supporting concurrent service versions.

A research challenge to address, therefore, is resolving the conflict between a service provider that needs to break backwards compatibility to evolve its interface in order to accommodate new requirements and the assumption from a set of service consumers who either assume that the service interface will remain fixed or are unable to adapt due to constraints such as lack of development agility.

II. CONTRIBUTION

This short paper expands on a solution to the problem of evolving Web Service interfaces through the use of a scheme that we call RES-WS to permit the restructuring of a Web Service's interface elements to support the automated generation of a SOAP message mediator. This work builds upon our previously published work in this domain [3] to offer verification and validation through empirical experimentation. In order to be successful this restructuring process needs to occur without affecting the functional compatibility with a service's consumers in addition to preserving the service's ability to operate within a minimal real-time performance overhead constraint.

III. EXPERIMENTATION AND RESULTS

The objective for this experimentation was to assess the overhead of applying RES-WS to a use-case and provide an empirical measure of the performance overhead incurred. To assess the relative effect of applying the RES-WS message mediator a sensor service integration demonstrator developed for a previous project [4] was benchmarked for performance prior to refactoring and the application of the message mediator. For this demonstrator the QoS for a response from the sensor integration workflow was 2000ms. The second objective was to assess the overhead of applying RES-WS to a Web Service providing service to multiple consumers in parallel using a serial and parallel mode message mediator. The following questions were asked:

1 Does the application of the message mediator increase the average response time above a baseline range?
2 Does the application of the message mediator result in an increase in response time as the number of parallel sensors increases?

In the experiments the number of sensors generated within the sensor simulation started at 1 and then increases in increments of 5 sensors from 5 onwards. 200 samples were recorded from multiple executions of the sensor detection workflow. The experiments performed are discussed below:

A. Baseline Experiment

The results retrieved from running the baseline experiments from unmodified services can be seen in Figure 1. In terms of response time performance, despite an increase in the sensor count, the results demonstrate that as the number of sensors increases past 10 sensors, there is no clear decrease in response time.

B. Initial Second Stage Experiment for Message Mediation

The purpose of this experiment was to assess the increased response time when the modified sensor retrieval workflow was not producing parallel requests and hence not expecting parallel invocations of the sensor services. The results of this experiment are presented in Figure 2.

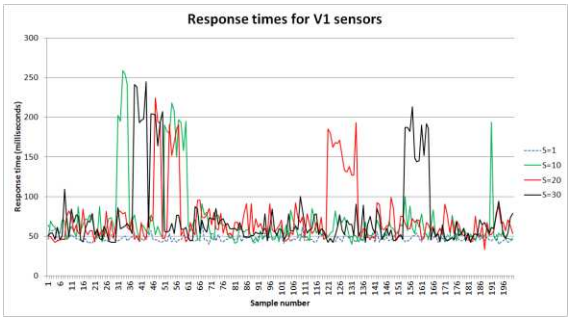

Figure 1: Results of baseline response times for unmodified V1 sensors.

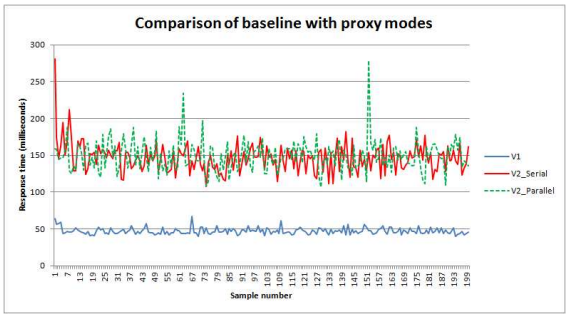

Figure 2: Raw response times (milliseconds) for Initial Second Stage Experiment.

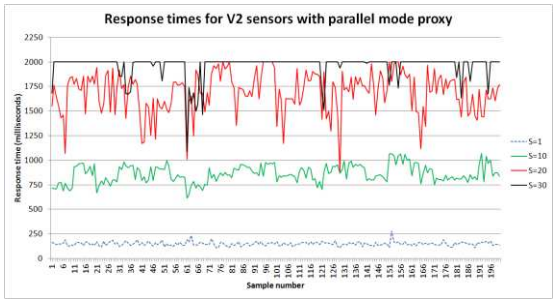

Figure 3: Raw results of Message Mediation in Parallel mode.

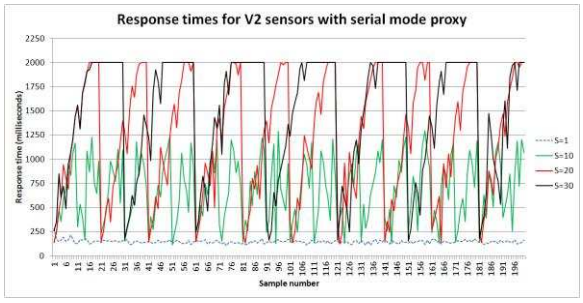

Figure 4: Raw results of Message Mediation in Serial mode.

C. Second Stage Experiment for Serial and Parallel mode Message Mediation

The next part of the second stage experiment was to determine the performance impact on the distributed system when the message mediator was used to handle parallel requests that needed a response within the QoS time frame. Given that the Workflow Service expects that instances of the Sensor Service would be invoked in parallel, the parallel execution mode for the message mediator provided the most natural fit. The results of this experiment are presented in Figure 3 and Figure 4.

IV. CONCLUSIONS

Although message mediator did not provide any value faults within the mediated messages, the results demonstrate that the probability of introducing a late timing fault increased as the number of parallel service invocations that the message mediator needed to deal with increases. The parallel operating mode produced a higher average response time than the serial mode, however, the standard deviation of the average response time was much lower. With the serial operating mode whilst messages entered into the message queue later had a high probability of generating a late timing fault under high loads, there was a high probability that earlier messages would return with a smaller response time than the parallel operating mode.

It is extremely difficult, however, to be able to give a system agnostic measure to help conclude when the message mediator will cease to be useful under domain specific constraints. Given that each scenario can provide variances in: message size; the number of restructuring operations; the number of parallel sensors and the QoS limit the evaluation of the message mediator must be evaluated against the system within which it is being used. Experimentation, however, has revealed issues that would not have been addressed through purely theoretical consideration of the problems associated with Web Service restructuring. Although the sensor integration demonstrator is a relatively small-scale system developed for research, the issues revealed will apply to commercial systems when scaled up and demonstrate that potential degradation of the QoS provided needs to be addressed.

ACKNOWLEDGMENT

The work reported in this paper has been supported in part by the NECTISE programme jointly funded by BAE Systems and the U.K. EPSRC Grant EP/D505461/1, the UK EPSRC WRG platform project (No. EP/F057644/1), the National Basic Research Program of China (973) (No. 2011CB302602), the UK TSB STRAPP project (No. 1926-19253), and the Major Program of the National Natural Science Foundation of China (No. 90818028).

REFERENCES

[1] T. Erl, A. Karmarkar, P, Walmsley, H. Haas, L. Yalcinalp, K. Liu, D. Orchard, A. Tost, J. Pasley, "Web Service Contract Design and Versioning for SOA," Prentice Hall PTR, Upper Saddle River, NJ, USA, 2009.

[2] V. Andrikopoulos, S. Benbernou, M. Papazoglou, " Evolving Services from a Contractual Perspective," in CAiSE '09: Proceedings of the 21st International Conference on Advanced Information Systems Engineering, vol. 5565, pp. 290-304, Springer-Verlag, 2009.

[3] D. Webster, P. Townend, J. Xu. "Interface refactoring in performance-constrained web services." Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), 2012 IEEE 15th International Symposium on. IEEE, 2012.

[4] D. Russell, L. Liu, Z. Luo, C. Venters, D. Webster, J. Xu, "Realizing Network Enabled Capability Through Dependable Dynamic Systems Integration," in proceedings of International Conference on Computer and Information Technology, pp. 1269-1274, 2010.