The Journal of Systems and Software 84 (2011) 603–619

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

XML-manipulating test case prioritization for XML-manipulating services☆

Lijun Mei[a], W.K. Chan[b,*], T.H. Tse[a], Robert G. Merkel[c]

[a] Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong
[b] Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Hong Kong
[c] Faculty of Information Technology, Monash University, Clayton, Victoria, Australia

ARTICLE INFO

Article history:
Received 19 January 2010
Received in revised form 13 October 2010
Accepted 18 November 2010
Available online 3 December 2010

Keywords:
Test case prioritization
Black-box regression testing
WS-BPEL
Service testing
Service-oriented testing

ABSTRACT

A web service may evolve autonomously, making peer web services in the same service composition uncertain as to whether the evolved behaviors are compatible with its original collaborative agreement. Although peer services may wish to conduct regression testing to verify the agreed collaboration, the source code of the former service may be inaccessible to them. Owing to the black-box nature of peer services, traditional code-based approaches to regression testing are inapplicable. In addition, traditional techniques assume that a regression test suite for verifying a web service is available. The location to store a regression test suite is also a problem. On the other hand, we note that the rich interface specifications of a web service provide peer services with a means to formulate black-box testing strategies. In this paper, we provide a strategy for black-box service-oriented testing. We also formulate new test case prioritization strategies using tags embedded in XML messages to reorder regression test cases, and reveal how the test cases use the interface specifications of web services. We experimentally evaluate the effectiveness of these black-box strategies in revealing regression faults in modified WS-BPEL programs. The results show that the new techniques can have a high chance of outperforming random ordering. Moreover, our experiment shows that prioritizing test cases based on WSDL tag coverage can achieve a smaller variance than that based on the number of tags in XML messages in regression test cases, even though their overall fault detection rates are similar.

© 2010 Elsevier Inc. All rights reserved.

1. Introduction

The testing and analysis of web services have posed new foundational and practical challenges, such as the non-observability problem (Canfora and Di Penta, 2006; Mei et al., 2009d), the extensive presence of non-executable artifacts within and among web services (Mei et al., 2008a, 2009b), safeguards against malicious messages from external parties (Xu et al., 2005; Martin et al., 2007), ultra-late binding (Bartolini et al., 2008), and cross-organizational issues (Ye et al., 2009). Researchers have proposed diverse techniques to address the test case selection problem (Martin et al., 2007), the test adequacy problem (Mei et al., 2008b, 2009b), the test oracle problem (Tsai et al., 2005a; Chan et al., 2007), and the test case prioritization problem (Hou et al., 2008; Mei et al., 2009c, 2009d).

Regression testing is the *de facto* activity to address the testing problems caused by software evolution (Onoma et al., 1998). It aims to detect software faults by retesting modified software versions. However, many existing regression testing techniques (such as Harrold et al., 1993; Kim and Porter, 2002; Mei et al., 2009c; Rothermel et al. 2001) assume that the source code is available for monitoring (Mei et al., 2009d), and use the coverage information of executable artifacts (such as statement coverage achieved by individual test cases) to conduct regression testing. Nonetheless, the coverage information on an external service may not be visible to the service composition that utilizes this service. Moreover, even though a technique may insert probing services to collect and compute coverage (Mei et al., 2008b, Section 5.3; Bartolini et al., 2009), the effect depends on whether the service being sampled is willing to provide such information accurately. Since such code coverage information cannot be assumed to be available, it is vital to consider alternative sources of information to facilitate effective regression testing.

Many web services (or *services* for short) use the *Web Services Description Language (WSDL)* (W3C, 2007a) to specify their functional interfaces and message parameters. They also use XML documents to represent the messages. To quantify the transfer of type-safe XML messages with external partners, the WSDL documents of a service further embed the types of such messages (Mei et al., 2008b).

☆ This research is supported in part by the General Research Fund of the Research-Grant Council of Hong Kong (project no. 717308), a strategicresearch grant of City University of Hong Kong (project no. 7002464), and a discoverygrant of the Australian Research Council (project no. DP0984760). A preliminaryversion of this paper was presented in QSIC 2009 (Mei et al., 2009c).

* Corresponding author. Tel.: +852 2788 9684.
E-mail addresses: ljmei@cs.hku.hk (L. Mei), wkchan@cs.cityu.edu.hk (W.K. Chan), thtse@cs.hku.hk (T.H. Tse), robert.merkel@benambra.org (R.G. Merkel).

0164-1212/$ – see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2010.11.905

WSDL documents are rich in interface information. Moreover, such documents are often observable by external services. Despite the richness of such documents, to the best of our knowledge, the use of WSDL documents to guide regression testing without assuming code availability has not been proposed or evaluated in the literature.

In general, different services developed by the same or multiple development teams may be modified independently of one another, and the evolution of services may not be fully known by every other service. With respect to a service maintained by a development team, a service maintained by another development team can be regarded as a service collaborator or a service consumer of the former service.

Let us consider a scenario that a service *A* (as a service consumer) would like to pair up with a service *B*, and yet the latter service may evolve over time or contain faults that lead to failures in some executions of their service collaborations. The service consumer *A* may want to execute some tests on the functions provided by *B* to ensure that *A*'s service composition has not been adversely affected (at least from *A*'s perspective). For instance, a company may want to make use of the electronic payment gateway provided by a bank to conduct payment transactions with the bank. Under this scenario, the internal service of the company is the service consumer of the payment gateway service of the bank. To the benefit of the company, the development team of the internal service would like to test its service collaboration with the payment gateway service comprehensively. In terms of testing, it typically means that many test cases will be used, which is costly to execute.

Furthermore, the program code of *B* (such as the payment gateway service of the bank in the above scenario) is generally inaccessible to *A* (the internal service of the company in the above scenario). Therefore, even though *A* may be able to discover and invoke a test suite to conduct regression testing on *B*, the above scenario makes impossible the test execution schedule that applies existing code-based regression testing techniques (Leung and White, 1989; Harrold et al., 1993) in general, and test case prioritization techniques (Rothermel et al., 2001) in particular, to improve the fault detection rate and achieve other goals.

The WSDL documents of services are accessible among peer services. It is well known, however, that black-box testing is not adequate and must be supplemented by white-box testing (Chen et al., 1998). *How well does the richness of information embedded in typical WSDL documents help alleviate this deficiency in service-oriented testing? Is it effective to use the black-box information in WSDL documents to guide regression testing to overcome the difficulties in testing services with hidden implementation details such as source code?* These questions motivate the study presented in this paper.

We observe that, in a regression test suite for service testing, existing test cases may record the associated XML messages that have been accepted or returned by a (previous) version of the target service. Because the associated WSDL documents capture the target service's functions and types of XML message, the *tags* defined in such documents and encoded in individual test cases can be filtered through all the WSDL documents. Moreover, we observe that a WSDL tag may occur several times in the same or different XML messages within a test case. For instance, to collect room booking information, multiple instances of room information often appear in the XML messages.

Following up on these observations, we propose two aspects in formulating test case prioritization techniques. The first aspect to make use of the tags in XML messages in relation to the WSDL documents of the service under test. We propose the use of WSDL tag coverage statistics and WSDL tag occurrence statistics. Based on these two statistics, we can cluster the test cases and iteratively select them from the sequence of clusters. The second aspect is to define the order of the sequence of clusters. There are many ways to do so, including simple orderings such as randomization, sorting, as well as more advanced sampling strategies. To facilitate further comparison of future research, we choose a simple strategy (namely, sorting according to the count statistics) so that researchers can easily compare it with their own strategies in the context of service regression testing.

Following these directions, we formulate four prioritization techniques as proofs of the concepts. We further conduct an empirical study on a suite (from Mei et al., 2009d) of WS-BPEL applications (OASIS, 2007) using both adequate test suites and random test suites to verify the effectiveness of our techniques. The results show that our techniques can have high chances of outperforming random ordering. Moreover, our experiment shows that prioritizing test cases based on WSDL tag coverage by regression test cases can achieve a smaller variance than that based on the number of WSDL tag occurrences resulting from regression test cases, even though their overall fault detection rates are similar. Our experiment also shows that the fault detection rates of our techniques on the subject applications can be less effective than white-box techniques, but this finding is *not* statistically significant.

Techniques for the construction of effective regression test suites are not within the scope of this paper. We appreciate that invalid test cases can be costly because they still require the execution of the service under test despite the lack of fruitful results. We assume that all the test cases in a given regression test suite are valid. Invalid regression test cases can be removed, for instance, using the information in the fault handler messages returned by the service, or using the WSDL documents of the services to validate the format of the test cases in advance. Once a test case has been identified to be invalid, it can be permanently removed from the regression test suite for that service. Thus, during the next regression testing of the service (without knowing in advance whether the service has evolved), the test case does not need to be considered in test case prioritization.

Existing techniques, such as Rothermel et al. (2001) and Mei et al. (2009d), assume that information on regression test cases is already available. In service-oriented applications, however, since the coordination program may use both in-house and external services to implement the functionality, the test suite for evaluating a service composition needs to be determined dynamically before each round of testing. We will, therefore, discuss how to model the entire testing procedure in this paper.

The main contribution of this paper with its preliminary version (Mei et al., 2009c) is threefold: (i) We propose a new set of black-box techniques to prioritize test cases for regression testing of services having observable and rich content interface. It eases the problem of autonomous evolution of individual services in service compositions, so that peer services can gain confidence on the service under test with lower cost in regression testing. Our technique is particularly useful when the source code of the service under test is not available or is too costly to obtain, (ii) We address the challenges in performing black-box regression testing for service-oriented applications, and develop a strategy to facilitate such testing, (iii) We report the first controlled experimental evaluation of the effectiveness of black-box regression testing in the context of service testing. Our empirical results indicate that the use of the information captured in WSDL documents (paired with regression test suites) is a promising way to lower the cost of quality assurance of workflow services. Our empirical results also indicate that the different partitions generated according to the different perspectives (white-box coverage or black-box coverage information) have different effects on the fault detection rates for different kinds of faults.

The rest of the paper is organized as follows: Section 2 introduces the foundations of test case prioritization. Section 3 introduces the preliminaries of our approach through a running

Download English Version:

https://daneshyari.com/en/article/458817

Download Persian Version:

https://daneshyari.com/article/458817

Daneshyari.com