Advances in Computer Science Research, volume 71
4th International Conference on Machinery, Materials and Information Technology Applications (ICMMITA 2016)

A Constraint-Based Approach to Web Service Simulation

Hui Zhou [a], Han Wu [b]

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
Nanjing 210016, China.

[2] Collaborative Innovation Center of Novel Software Technology and Industrialization
Nanjing 210023, China.

[a]zh_zabulon@163.com, [b] HanFangWu@foxmail.com

Keywords: Web service, constraint solver, simulation testing

Abstract. Web services have been widely used in complex software systems. To guarantee software quality, many researchers have been devoted to the testing of web services. Traditional web service testing approaches mainly focus on the automation of testing process, test case generation etc. However, the web service simulation testing is still far from mature. This paper proposes a constraint-based web service simulation approach. The approach uses first order logic formulas to specify the relationships between service outputs and inputs and can simulate a service with a constraint solver. Compared to other approaches, our approach can support complex first order logical constraints and thus are capable of specifying various correlations between variables. Researchers can quickly and efficiently do simulation testing of web services by using the approach.

Introduction

Service-oriented software architecture has been widely used in many field of society in recent years. The size of Web service software is getting bigger and bigger, but at the same time, there are more and more defects in Web services. In order to guarantee the quality of Web services, it is necessary to test it in detail. All along, the researchers also widely discussed various Web service testing technology [1][2]. However, most of the existing service-oriented software testing tools focus on the automation of testing process, test case generation[5][6], test results analysis, etc.. However, the Web service simulation test is far from mature. At present, many researchers use the WSDL interface to create simulation service. Generating output value randomly and setting program pile are two main ways. However, the former way may be difficult to match the actual business logic, and the latter way takes a lot of time to develop the pile module.

With the development of constraint solving technology, constraint system has made great progress in solving skills and abilities, and there have been a lot of sophisticated constraint solving tools .We can generate simulation service and test Web service better with the help of the constraint solver. Based on the above, this paper proposes a simulation test approach for Web service based on constraint solver. In this approach, the first order logic is used as the main expression of constraints, and the constraint solver is used to support the data solution under complex constraints. Through establishing the association between Web service input and output operation with the help of the constraints, and using constraint solver to solve the constraints. Then using the generated data to build the simulation services. Last, through running the simulation services to test the robustness of Web services and interfaces. With strong constraint solving ability of solver tools, the approach can not only support the linear operation general integer types, but also be able to float type on the test data generating and even support the complex computing formula may be aroused in the constraint system such as bit operation. We can expand the scope of the test functions to support the Web service type. By coding string type to integer type, and applying the regular expression to generate string type data.

Copyright © 2017, the Authors. Published by Atlantis Press.
This is an open access article under the CC BY-NC license (http://creativecommons.org/licenses/by-nc/4.0/).

Constraint Management

The approach supports most of the first order logic constraint formulas which can be processed by solver. In order to express the constraint relationship, we define our own constraint expression language, the language will be translated to constraint language for constraint solving, and the solution will be mapped to Web service input data to meet the constraints or not.

Constraint Variables. The constraint formula is based on the constraint variables, and the constraint system supports value constraint variable. Its expression form is

value (*svc\operation\message_path*),

Value is a variable name, used to distinguish among other types of constraints. *svc* is a service identifier, which can be used to distinguish different services. The specific meaning of *svc* name can be defined in advance by the way of macro definition. for example

svc = http://test.com/axis2/services/Add,

Constraint Formula. The constraint formula is essentially a first-order logic formula supported by the constraint solver, which can be established between the different parameters of a single service operation. Constraint formula could be used to restrict the input data and also could be used to describe a possible connection between the output and the input of a service operation. Constraint formulas can also be established between different services, which are mainly used to describe the relationship between service invocation and timing. Typical examples of constraint formula are as follows.

1) value constraint formula. Like

value (*svc\add\a*) < value (*svc\add\b*) (1)

The expression express the relationship that the value of *a* should be less than the value of *b* in *add* services.

2) timing constraint formula. Like

startTime (*svc\add*) < startTime (*svc\multiple*) (2)

The expression express that *add* operation should be performed before the *multiple* operation.

We use the AND, OR, NOT relationship to the organize the structure of the constraints. The leaf node of the tree is a basic constraint, and each constraint contains a constraint formula. A more complex constraint system is composed of AND, OR and NOT relations. Each constraint node can be named in order to visually express the constraints of meaning.

Simulation Test Framework

Simulation Service. A Web service often depends on a number of third party services, these third party services constitute the runtime environment of the service. In order to check the correctness and the performance of the the services in different environments, we often need to configure the runtime environment of by setting up working status and QoS of the deployed third party services. Because the third party service itself is difficult to modify the code, and we need use simulation technology to deploy these services. we can create a simulation implementation of third party services Through the simulation. The simulation can simulate the behavior of third party services and control the performance of third party services. We call these services implemented by simulation technology simulation services . The message SOAP package will be intercepted by a proxy gateway when a service is ready to call the third party service, and then the SOAP package will be forwarded to the simulation service. The simulation service decide whether forwarding the package to the original service or request the package itself according to the simulation configuration. The role of simulation services can be expressed in the Fig. 1.

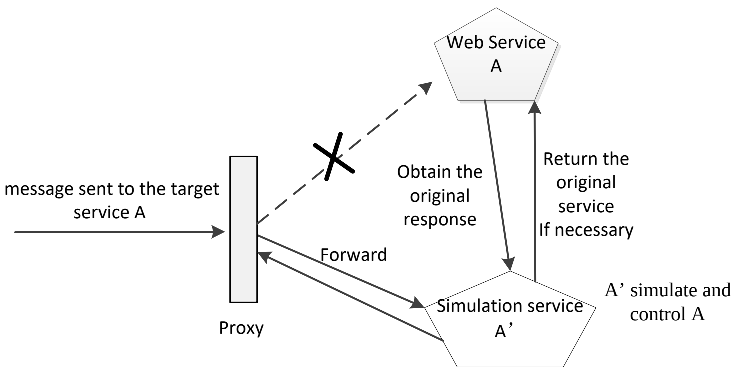
Web Service A
message sent to the target service A
Obtain the original response
Return the original service If necessary
Forward
Proxy
Simulation service A'
A' simulate and control A

Fig. 1. The simulation service function

Implementation of constraint-based simulation service. In order to guarantee the expansibility and flexibility of the approach, this paper uses the dynamic approach to construct the simulation service. When we need to create a simulation service for a A service we will only create a shell for A in the first time, and its interface type, behavior, and QoS control will be injected at runtime. The main advantage of this approach is the speed of the generation of simulation service, and the simulation service logic and QoS control scheme are easy to be changed. In addition, the approach is easy to avoid many kinds of defects.

The specific structure of simulation service is shown in the Fig. 2. The shell can be achieved by creating a framework for service class 'VirtualServiceMessageReceiverInOut'. Once the framework service receives any SOAP messages, it can feedback any SOAP packets. The injection of business logic controls the input and output SOAP message.

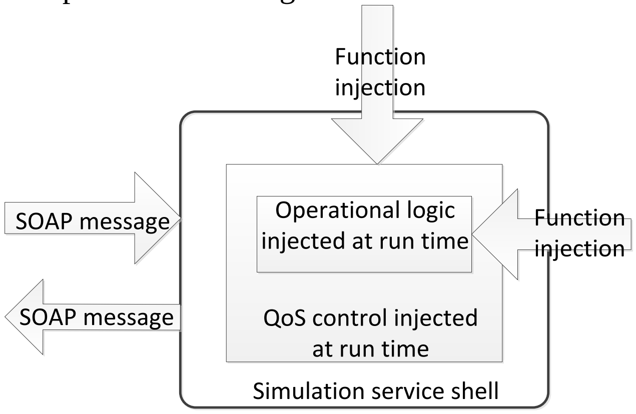

Fig. 2. The specific structure of simulation service

The injection of QoS is consist of a QoS quality configuration and a section code which were used to control response time, the maximum amount of concurrency, calling success rate. And the business logic operation is injected with WSDL and constraint information. The simulation service decides what information should be output according to the SOAP information. Including the output data format and value of each output data.

Fig. 3 shows the procedure of simulation service test, the first step is access the constraint between input and output of simulation service, and then replace the partial constraint variables with the real parameters values, and then mapping the output of the service to constraint variables, then solving the constraint formula. After obtaining the output variable value, we can build the SOAP package of the service output.

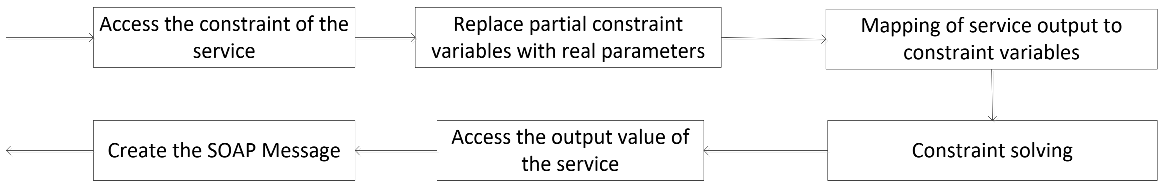
Access the constraint of the service
Replace partial constraint variables with real parameters
Mapping of service output to constraint variables
Constraint solving
Access the output value of the service
Create the SOAP Message

Fig. 3. The procedure of simulation service test

Simulating the operation of a *middle* service as an example, the specific process is as follows.

First, the input and output constraint of *middle* service is as follows:

(value(*svc\middle\result*) > value(*svc\middle\x*)
and value (*svc\middle\result*) < value(*svc\middle\y*))
or (value(*svc\middle\result*) > value(*svc\middle\y*)
and value(*svc\middle\result*) < value(*svc\middle\x*))。 (3)

Then, when the simulation service is called, it will obtain the the input parameters from caller, for example, {x = 10, y=20}. and put the real argument into the constraint system, the new constraint formula is as follows:

(value(*svc\middle\result*) > 10 and value(*svc\middle\result*) < 20)
or (value(*svc\middle\result*) > 20 and value(*svc\middle\result*) < 10) (4)

Next, a *middle* service can obtain an output value by the solver, result = 15. The output value meets the business logic setting, which is more reasonable than the random approach , and the process is smaller than developing a pile module.

Conclusion

In this paper, we propose a constraint-based simulation testing approach for web services. A constraint solver is used to build Web service simulations and we can test Web service better and easily. Compared with the existing service simulation testing techniques, this approach can handle various types of constraints, and can deal with correlation constraints between service inputs and outputs. The approach lets business logic restrictions be expressed and treated effectively and easily, which can improve the quality of test data, and high quality service simulations with better fault revealing ability can be built.

Acknowledgments

This work was supported by the China Defense Industrial Technology Development Program (JCKY2016206B001, JCKY2015206C002) and the Science and Technology Planning Project of Jiangsu Province (BY2016003-02)

References

[1] Bozkurt M, Harman M, Hassoun Y. Testing and Verification in Service-Oriented Architecture: A Survey[J]. Software Testing, Verification and Reliability, 2013, 261-313.

[2] Dong Qiu, Bixin Li, Shunhui Ji and Hareton Leung. Regression Testing of Web Service: A Systematic Mapping Study[J]. ACM Computing Surveys, 2014,47(1),21-46.

[3] JIN Jiwei, MA Feifei, ZHANG Jian. Brief introduction to SMT solving. Journal of Frontiers of Computer Science and Technology, 2015, 9(7):769-780.

[4] De Moura L, Rner N. Z3: an efficient SMT solver[C]// Theory and Practice of Software, International Conference on TOOLS and Algorithms for the Construction and Analysis of Systems. Springer-Verlag, 2008, 337-340.

[5] S. Hanna and M. Munro, Fault-based Web services testing, 5th International Conference on Information Technology[C]//New Generations (ITNG). USA, 2008, 471–476.

[6] Li Z J, Zhu J, Zhang L J, et al. Towards a Practical and Effective Approach for Web Services Test Case Generation[J]. Automation of Software Test, 2009,106–114.