A Technique to Enhanced Robustness of Web Service

Priyanka S. Mangale[1], Subhash V. Pingale[2], Vijaysinh G. Chavan[3]
[1,2,3]*Department of Computer Science & Engg, Solapur University, Solapur*
SKN Sinhgad College of Engineering, Korti, Solapur, MS, India
[1]priyanka.mangale8@gmail.com, [2]subhash.pingale@sknscoe.ac.in, [3]vijaysinh.chavan@sknscoe.ac.in

Abstract - ***Web service is medium of communication through which two applications or machine will exchange data. It is designed to support machine to machine inter-operable interaction of system over a network. There are large number of web services deployed with robustness problems. Robustness problem means unexpected behaviour of the system in the presence of invalid input. Robustness improvement approach detects robustness problems in web services using robustness testing and then reduces those issues by applying inputs verification based on well-defined parameter domains. This proposed approach can be easily used to improve the robustness of web services code.***

Keywords - Robust Web service, Service Oriented, Workload testing and debugging, Architecture.

I. INTRODUCTION

Web Services are an implementation of the Service Oriented Architecture (SOA). Many businesses use Service Oriented Architecture like banking, e-commerce online shopping, transportation, etc. The Web Services architecture is based upon the three roles that interact with each other as service provider, service registry and service requester. Here service requester is distributed application builder. This requires certain functions to be satisfied from business point of view and from architectural point of view, this is application that is looking for and invoking or initiating an interaction with service. The service requester role can be played by a browser obsessed by a person or a program without any user interface. Service provider is the owner of the service from business point of view and from architectural point of view; this is the platform that hosts access to the service. Service registry is search able registry of service description where service provider publish their service description. The service requester finds a service and obtain binding information in the service description for service during the development for static binding or during execution for dynamic binding. The service requester can obtain a service description from other source additional to a service registry. These interactions involve publish, find and bind operations.

1. Publish: A service description needs to be published to be accessible so that the service requester can find it, where it is published can vary depending upon the requirements of the application.

2. Find: In the find operation, the service requester recovers a service description directly or requests the service registry for the kind of service required.

3. Bind: A service needs to be invoked. Within the bind operation the service requester invokes or initiates AN interaction with the service at run-time victimization the binding details within the service description to find, contact and invoke the service.

The Web services background is split into three areas — communication protocols, service descriptions, and repair discovery. Web services are an implementation of SOA that is based on set of XML based technologies such as SOAP and WSDL. Simple Object Access Protocol (SOAP) which plays the role of messaging protocol for exchanging data between service provider and service requester. It is stateless and one way message exchange standard. Web Service Description Language (WSDL) is used to describe web service as collection of communication endpoints that can exchange particular type of message. It provide a formal, computer readable description of web service. It describe interface of service and provide users. Universal Description, Discovery and Integration (UDDI) that is registry of web service. It provides a mechanism for clients to discover web service. It is used to find service providers through a centralized registry [8].

Web service provides a simple interface between provider and consumer. Provider provide set of services that are used by consumer. Provider must provide robust web service to consumer.

Major faults in software causes system failure. Interface faults is related to problems in the interaction among software components/modules. Web service must provide robust interface to client application even in the presence of invalid inputs. Web service are dynamic, use an immense diversity of software system.. There is immense need of robust web service on online client application even when the input is wrong.

Robustness is defined as the degree to which system operates correctly in the presence of exceptional input. Robustness testing is testing methodology to detect the vulnerabilities of component under unexpected input. It is efficient and effective technique which characterize the behavior of the system.

Previous research on web service robustness testing shown that services are being deployed on internet with robustness problem, which require adequate robustness testing and improvement methodology and tool that can be used by developers. Code coverage analyzers, such as Cobertura [9] and Clover [10], can be used to validate and improve workloads through the automatic identification of the areas of a program that don't seem to be exercised by a collection of take a look at cases. These tools don't need access to the source code.

II. LITERATURE REVIEW

Nuno Laranjeiro, Marco Vieira and Henrique Madeira has mentioned a Technique for Deploying Robust Web Services. This technique involve

1. A service domain description language. Extended domain expression language—EDEL) that enables the inclusion of a detailed domain description into the XML Schema document (XSD) typically referenced by the web services' WSDL file. This language enables the full expression of domains, including complex parameter dependencies

2. A new and fully automatable workload generation process. The goal of this process is to create a set of service invocations capable of accurately exercising any web service implementation. The generated workload serves as a basis for testing the web services robustness and can explore the parameters domain space in an effective way.

3. An improved robustness testing approach designed for web services. Using the generated workload and fault injection techniques, apply robustness testing to identify robustness problems in web services implementations. This seminar includes the automatic detection of special external contact points and execution of tests at those points. The fault model has been extended and exceptions are now also injected at those external services invocation points. The testing results are automatically classified.

4. A technique to eliminate robustness problems. Using EDEL as a starting point, propose a technique that endows web services with the capability of performing input verification based on EDEL announced input domains. This approach does not require any access to the source code of the application [1].

Mei-Chen Hsueh, Timothy K. Tsai, and Ravishankar K. Iyer gave overview of Fault Injection Techniques and Tools that focused on Hardware fault injection technique and Software fault injection technique. This paper includes a Fault injection provides a method of assessing the dependability of a system under test. It involves inserting faults into a system and monitoring the system to determine its behavior in response to a fault.

The fault injection techniques have been recognized for a long time as necessary to validate the dependability of a system by analyzing the behavior of the devices when a fault occurs. In Hardware fault injection uses additional hardware to introduce faults into the target system's hardware. In Software fault injection, inject faults at compile-time, the program instruction must be modified before the program image is loaded and executed. Rather than injecting faults into the hardware of the target system. During run-time, a mechanism is needed to trigger fault injection [2].

M. Vieira, N. Laranjeiro, and H. Madeira have presented idea about Assessing Robustness of Web-Services Infrastructures. To reveal robustness problem in web service code described using WSDL Robustness testing approach is used. It consists of a set of erroneous web-services call parameters that are injected during the web-services execution. The robustness tests are based on combinations of exceptional and acceptable input values of parameters of web services generated by applying a set of predefined rules according to the data types of each parameter. They presented

1) Analysis of the WSDL of the services under testing in order to identify the relevant call parameters.

2) Robustness tests generation based on the information gathered in the previous step.

3) Execution of the web-services without fault injection in order to collect baseline performance metrics and to understand what behavior is expected from each service.

4) Execution of the web-services in the presence of invalid call parameters (robustness tests). The goal is to trigger faulty behaviors, and in that way expose robustness problems.

5) As a final step the results obtained in 3) and 4) are used to characterize the performance and robustness of the web-services infrastructure. In this set of test is applied to observe robustness problem when web service execute. By using this we can reveal robustness problem. But not applicable to external web service response [3].

N. Laranjeiro, M. Vieira, and H. Madeira gave Improving Web Services Robustness. This paper proposes approach used robustness testing to detect robustness issues and then mitigate those issues by applying inputs verification based on well-defined parameter domains, including domain dependencies between different parameters. This mentioned following steps to improve robustness of web service.

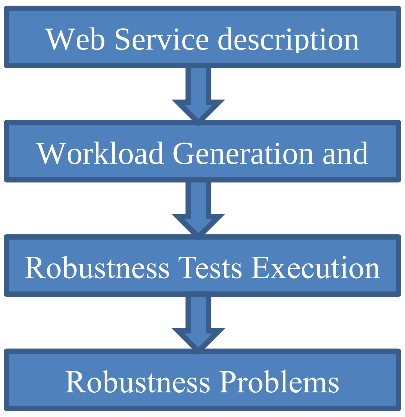
1. Gather required information and prepare the web service.
2. Workload generation and execution
3. Execute a set of robustness tests
4. Fix disclosed robustness problem and verify the service behavior.

This provide solution that closes the existing gaps. EDEL code is used to represent domain information, but this EDEL code is complex [4].

K.M. Senthil Kumar, A.S. Das, and S. Padmanabhuni provided an overview of WS-I Basic Profile: A Practitioner's View. Interoperability refers to the ability of software and hardware on multiple machines from multiple vendors to communicate with each other without significant changes on either side. To address the interoperability issues with core web services standards. This paper introduces a basic profile 1.0 that addresses the interoperability issue. They have used to examine some of the key interoperability issues that are encountered in SOAP 1.1 and WSDL 1.1 implementations by practitioners [5].

In first [1] paper there is no strategy discussed to improve the robustness of WCF Services. But in our project we are going to develop robustness improvement tool for WCF Services.

III. PROPOSED SYSTEM



In this collect information about the web service. In that first gather information about list of operations, parameters, and data types. Then gather information about all valid domains for all input and output. In first step obtaining relevant definitions about the web service. Web service interface is described using WSDL file. We obtain the list of operations, parameters, and associated data types by automatically processing XML file. The information describing the structure and type of all inputs and outputs of each operation is usually found in a XML Schema file which is referenced by the original WSDL.

The second step consists of gathering data on the valid domains for all input and output objects using XSD file. This file may also include information about valid values. The EDEL version proposed in this paper defines the dependency id attribute, the operate id attribute, and also the param index attribute as optional (index is zero by default). In practice, this results in more compact EDEL code.

Next step is to generate workload means a set of valid web service calls using one or more generation strategy.

Using the XSD file as the starting point, we are able to automatically generate a synthetic workload (stage 1).

By using XML Schema Definition Tool, it is possible to use the generated values, we need to create programming language level objects that accurately represent the structures found in the XSD file (stage 2).

The produced XML is de-serialize into the corresponding generated .net objects. This form list of object that form our final workload. (Stage 3).

This is a process that uses reflection to load classes by name and builds a list of object that are integrated into one unit test case per each service operation.(stage 4).

N Cover is coverage analysis tool to get metric of the code coverage. (Stage 5)

Measure the workload coverage using code coverage analyzer. Again generate workload or change generation strategy until acceptable percentage of workload is achieved. Add domain information about any existing external service calls. (Stage 6)

This generated workload is executed to verify service answers.

Perform set of robustness test to identify robustness issues. Parameter domain to be redefined if needed then goes to earlier step.

Fix disclosed robustness problems and verify the service behavior. Repeat the robustness tests to check if the robustness problems identified before were fixed and if new problems appeared. If new problem arise then Re-execute the workload and verify service answers to identify potential deviations

IV. CONCLUSIONS

In this paper, we have reviewed a technique to assess robustness of web service. This review shows there is no strategy discussed to improve the robustness of WCF Services. This proposed approach will be used by web service developers to improve the robustness of their service. The approach provide information about web service, generate and execute workload, robustness testing. By using this approach multiple robustness problems will be disclosed and corrected. This will improve the robustness of web service.

ACKNOWLEDGMENT

I am profoundly grateful to Prof. V. G. Chavan for his expert guidance and continuous encouragement throughout to see that this paper rights its target since its commencement to its completion.

I like to express deepest appreciation towards Prof N. M. Sawant, (Coordinator of ME CSE) SKN SINHGAD COLLEGE OF ENGINEERING whose valuable guidance supported us in completing this paper.

At last I must express our sincere heartfelt gratitude to all the staff members of Computer Engineering Department who helped me directly or indirectly during this course of work.

REFERENCES

[1] Nuno Laranjeiro, Marco Vieira and Henrique Madeira : A Technique for Deploying Robust Web Services, IEEE Transactions On Services Computing, Vol. 7, No. 1, January-march 2014

[2] Mei-Chen Hsueh, Timothy K. Tsai, and Ravishankar K. Iyer : Fault Injection Techniques and Tools, Computer, vol. 30, no. 4, pp. 75-82, Apr. 1997

[3] M. Vieira, N. Laranjeiro, and H. Madeira: Assessing Robustness of Web-Services Infrastructures, Proc. IEEE/IFIP 37th Ann. Int'l Conf. Dependable Systems and Networks, pp. 131-136, 2007.

[4] N. Laranjeiro, M. Vieira, and H. Madeira: Improving Web Services Robustness,Proc. IEEE Int'l Conf. Web Services, 2009.R. E. Sorace, V. S. Reinhardt, and S. A. Vaughn, “High-speed digital-to-RF converter,” U.S. Patent 5 668 842, Sept. 16, 1997.

[5] K.M. Senthil Kumar, A.S. Das, and S. Padmanabhuni: WS-I Basic Profile: A Practitioner's View, Proc. IEEE Int'l Conf. Web Services, pp. 17-24, 2004.

[6] V. Santiago, A.S.M.D. Amaral, N.L. Vijaykumar, M.D.F. Mattiello- Francisco, E. Martins, and O.C. Lopes, A Practical Approach for Automated Test Case Generation Using Statecharts, Proc. 30th Ann. Int'l Computer Software and Applications Conf., pp. 183-188, 2006.

[7] C. Fetzer and Z. Xiao, HEALERS: A Toolkit for Enhancing the Robustness and Security of Existing Applications, Proc. IEEE/IFIP Int'l Conf. Dependable Systems and Networks, pp. 317-322, June 2003.

[8] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi, and S. Weerawarana, Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI, IEEE Internet Computing, vol. 6, no. 2, pp. 86-93, Mar. /Apr. 2002.

[9] M. Doliner, "Cobertura," http://cobertura.sourceforge.net/, 2010.

[10] Atlassian,"Clover-Code Coverage Analysis," http://www.atlassian.com/software/clover/, 2010.