ISSN: 2321-7782 (Online)
Volume 2, Issue 1, January 2014
International Journal of Advance Research in
Computer Science and Management Studies
Research Paper
Available online at: www.ijarcsms.com

Testing Web Services by Applying Program Slicing

Harkishan Rathod[1]
Computer Engineering Department,
Government Engineering College, Modasa,
Gujarat - India

Kaushik Rana[2]
Computer Engineering Department,
Government Engineering College, Modasa,
Gujarat - India

Abstract: Service Oriented Architecture is the latest technology for the today's business or enterprise. Since SOA applications are more complex, we need the way of the assuring the quality of the Web Service. Till now Service Oriented Architecture can only be tested with traditional black box testing. In this paper we are going to apply Dynamic slicing technique for white box testing of web services.

I. Introduction

Service Oriented Architecture applications are growing in both numbers and complexity. Complex Web Services require more effort to ensure the quality of the service by testing the Web Service. In Service oriented Architecture the Web Service is tested in the IDE (i.e., netbeans or visual studio), although the testing facility is provided by the software it is the kind of the testing were we don't know what is happening in the testing, just we can check whether the output given by the Web Service is correct or not. This kind of testing test the Web Service until it is not so complex, but in case of complex Web Service it is necessary to know how the flow is going and which variable are being affected and by which statement, etc. White box testing in the Service Oriented Architecture may become a great need in developing and testing complex Web Service, because there is no any method for white box testing for the Web Service and the only method for testing the Web Service is black box testing. Here we are applying the program slicing technique for white box testing of the Web Service.

II. Black Box Testing INSOA

We have already tested simple web service in netbeans IDE by right click on the web service and click on the Test service. The service now starts of being tested in following manner. Integrated Development Environment (IDE , i.e., netbeans or eclips or visual studio, or any one) create one interface for the web service i.e., a Web page that interact with the user where it asked for the input according to the input parameters of the Web Service (textbox for the input and button for the Web method, when we click on the button the web method is invoked and the result is displayed on the web page with value its data type, it also display the input values and their data type and definition of SOAP Request and SOAP Response of the Web Service). We can check the web service only with their input values and data type and output values and their data type. No one has idea about what is going on inside, for complex problem it is impossible to find the point of the error. For such case white box testing is necessary that provide interaction with the web service and find the error in the web service.

III. Challenges in SOA White Box Testing

In Service Oriented Architecture there are various challenges to face for the white box testing. Following are the challenges facing for the white box testing for the Web Service in Service Oriented Architecture (Or dependencies)

1. *Unavailability of the Service Code*

In service Oriented Architecture Web Service is described by the Web Service Description Language (WSDL) and located using Registry (i.e. UDDI). In such environment only Web Service interface is available, Web Service code is not available to the third party (i.e. Service Consumer) in that case we have to take assumption about the availability of the Service code at the time of testing.[5] Here we are introducing the method of testing Web Service before the deploying it.

2. *User Interface*

The thing that makes the Service Oriented Architecture interesting is that Web Services do not have the user interface. So in order to test the Web Service tester would have to create its interface according to Web Service's input parameter and return value type also in form of their SOAP request and SOAP response. It is the user interface that interacts with the user, so it is very essential to create user interface of the Web Service. [3]

3. *Knowledge of SOA*

Service Oriented Architecture is having different style in programming and different representation style (in XML) different protocols used (SOAP request and SOAP response), different architecture different model (Service Provider, Service Consumer and Registry). In that case one would have the knowledge of all fundamental aspect of Service Oriented Architecture in order to understand the flow of the data and control in the testing. [5]

4. *Composition of the Web Service*

Web Service is some time composes of other Web Service that may be on the same domain or on another domain in the network. In such case we identify following type of Service Composition

- **Intra domain**

  Web Service that is used in one web service is available on the same machine.

- **Inter domain but intra network**

  Web Service that is used in one web service is available on another machine but in the same network.

- **Inter domain and inter network**

  Web Service that is used in one web service is available on another machine in different network.

In different type of service call different types challenges have to face. In Intra domain Web Service, Web Service is on the same domain, and can be used for that domain only. In inter domain but intra network web service is deployed on another machine on the same network. This web service can be access with the speed of the network speed. It is also affected by the reliability of the network. In inter domain and network the Web Service is deployed on another machine in the different network, in such case it is highly dependence on speed of network and reliability of the network, Communication way used (by fiber optics or by broad band, etc.).

5. *Reliability of the Service Provider*

When Web Service is composed of another web service that is on another domain than it is depended on reliability of that domain (i.e. in how much time it respond to the called domain).

IV. White Box Testing INSOABY Applying Program Slicing Techniques

The concept of program slicing was introduced by mark weiser. [4] Finding all statements in a program that directly or indirectly affect the value of a variable occurrence is referred to as Program Slicing. [2] Program slicing is the process of computing program slice. [6] Program slice consist of the parts of the program that affect the values computed at some point of interest. [1] In program slicing techniques slice is made with respect to slicing criterion. Slicing criterion is typically defined as combination of variable and location of statement in the program as <s, V>, where s is the statement number and V is variable. [1]

There are various types of program slicing. Among them most basic types of the program slicing are static slicing and dynamic slicing. Static slicing technique uses static analysis to derive slices. [1] That is, the source code of the program is analyzed and the slices are computed for all possible input values. Therefore static slice contain more statement than necessary. While dynamic slicing makes use of the information about a particular execution of a program. [1] Dynamic slicing is more preferable as the slice computed by dynamic slice is having less statement compared to static program slicing. [1] Here therefore we use dynamic program slicing approach in testing in Service Oriented Architecture.

Since in SOA service code is directly not available at the run time, therefore we have to make some assumption as bellow.

1. The service code is available to us and we are testing Web Service before deploying it.
2. The computer on which the Web service is published (Service Provider) is available and it is working correctly.
3. Our method of white box testing in Service Oriented Architecture is the testing before we deploy the Web Service.

A. Dynamic program slicing in Service Oriented Architecture

In dynamic program slicing the slice is computed using slicing criterion <s, V>, where s is the statement number and V is the variable. In flow back analysis, one is interested in how information flows through a program to obtain a particular value. User interactively traverses a graph. The graph is represents the data and control dependences between the statements in the program. For example, if the value computed at any statement *x* depends on the values computed at statement *y*; we may backtrack from the vertex corresponding to *x* to the vertex for *y*.

Example: Consider following code

```
            begin
S1:               read(X);
S2:               if (X < 0)
                  then
S3:                   Y := f1(X);
S4:                   Z := g1(X);
                  else
S5:                   if (X = 0)
                      then
S6:                       Y := f2(X);
S7:                       Z := g2(X);
                      else
S8:                       Y := f3(X);
S9:                       Z := g3(X);
                      end_if;
                  end_if;
S10:              write(Y);
S11:              write(Z);
            end.
```

Fig. 1. Program. [2]

In programs Program slicing is implemented using PDG. (Program dependence graph) The Program dependence graph is the graph whose nodes are the statement in the program and edge of the graph represents the dependence between them. The edges of the graph are of two types one is data dependence edge and other is control dependence. [2] A data dependence edge from vertex v_i to vertex v_j means computation is performed at vertex v_i directly depends on the value computed at vertex v_j. In other words the computation at vertex v_i uses a variable, that is defined at vertex v_j, and also there is an execution path from v_j to v_i. A control dependence edge from v_i to v_j means that node v_i may or may not be executing depending on the Boolean outcome of the predicate expression at node $v_{j.}$

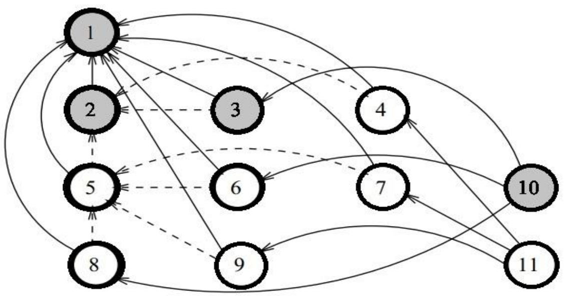


Fig. 2. Program Dependence graph for Figure 1 for input value X=5, where slice consists of statements 1, 2, 3 and 10. Here solid edge represents data dependence and dashed edge represents control dependence. And darken node represents the slice of the program. [2]

Web Services have different dependence other than traditional programs and therefore here we introduce a new type of graphs as bellow.

B. Web Service Dependence Graph (WSDG)

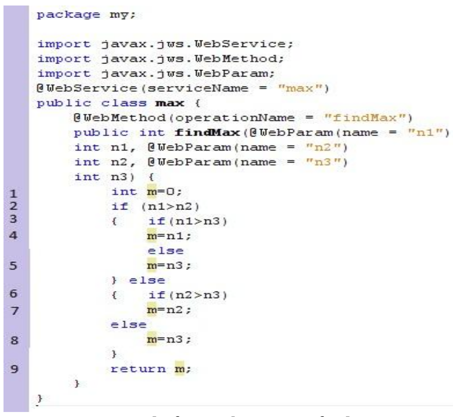
In order to compute slice of a Web Service we will propose deferent type of directed graph, Web Service Dependence Graph (WSDG). In WSDG following types of dependencies are identified.

- Data Dependence
- Flow dependence
- Callee dependence

Data Dependence is a type of dependency in which particular statement depends on any data or variable. Flow dependence is a type of dependency in which particular statement depends on the flow of the execution, in case of calling the other Web Service on the same domain or calling the Web method from the same domain. While Callee dependence is a kind of dependence in which particular statement depends on the result of the Web Service that execute on the remote machine. Here we go for the testing of the Web Service before deploying the Web Service. And therefore we don't have to access to the service code of the remote Web Service. In WSDG the nodes are the statement number, and edges are of three types. One is data dependent edge, second is flow dependent edge, and the third is callee dependent edge. The Data dependent edge is represented as solid line, flow dependent edge is represented as dashed line, and the callee dependent edge is represented as dotted line.

For Data dependence if there is a node from v_j to v_i then we can say that statement v_j depends on the variable from statement v_i. For if flow dependence, if there is a node from v_j to v_i then we can say that statement v_j depends on the flow of execution from statement v_i or in other words statement v_j depends on the result of the local called Web Service from statement v_i. For callee dependence, if there is a node from v_j to v_i then we can say that statement v_j depends on the result of Web Service called from remote machine from statement v_i.

Example: findMax Web Service (A Web Service which finds maximum number from given three numbers)



```
   package my;

   import javax.jws.WebService;
   import javax.jws.WebMethod;
   import javax.jws.WebParam;
   @WebService(serviceName = "max")
   public class max {
       @WebMethod(operationName = "findMax")
       public int findMax(@WebParam(name = "n1")
       int n1, @WebParam(name = "n2")
       int n2, @WebParam(name = "n3")
       int n3) {
1           int m=0;
2           if (n1>n2)
3           {   if(n1>n3)
4               m=n1;
                else
5               m=n3;
            } else
6           {   if(n2>n3)
7               m=n2;
            else
8               m=n3;
            }
9           return m;
       }
   }
```

Fig. 3. Code for Web Service findMax

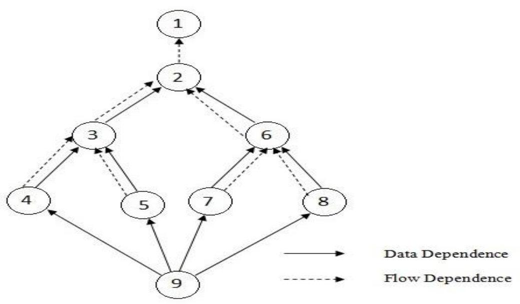


Fig. 4. WSDG for code in Figure 3

For simplicity we consider only data dependence.

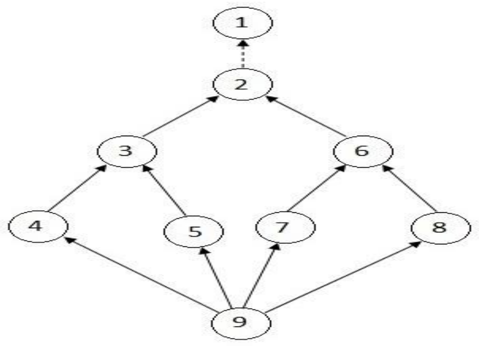


Fig. 5. simple WSDG for Figure 3

Consider a test case for input (5, 3, 2)

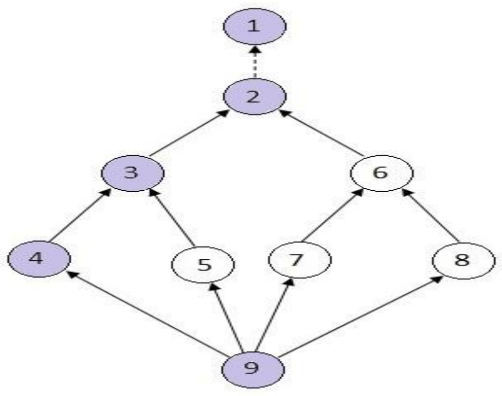
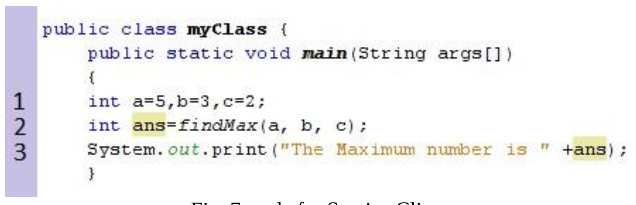


Fig. 6. Test case for value ((5, 3, 2),9)

Slice for above input will comprise of following statements (1, 2, 3, 4, and 9)

C. Service Client Dependence Graph (SCDG)

There is other kind of the directed graph at service client called Service Client Dependence Graph. The control/ flow moves from Service Client Dependence Graph to Web Service Dependence Graph when the Web Service is called. There are only two kind of the dependence in the SCDG. One is Data dependence and other is Control dependence. Data Dependence is represented as solid line. Control dependence is represented as dashed line. Control dependence occurs in the statements of looping or branching or jumping or when Web Service is called. Example: consider following code for the service client.



```
public class myClass {
    public static void main(String args[])
    {
    int a=5,b=3,c=2;
    int ans=findMax(a, b, c);
    System.out.print("The Maximum number is " +ans);
    }
```

Fig. 7. code for Service Client

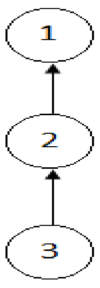


Fig. 8. SCDG for Figure 7

D. Service Dependence Graph (SDG)

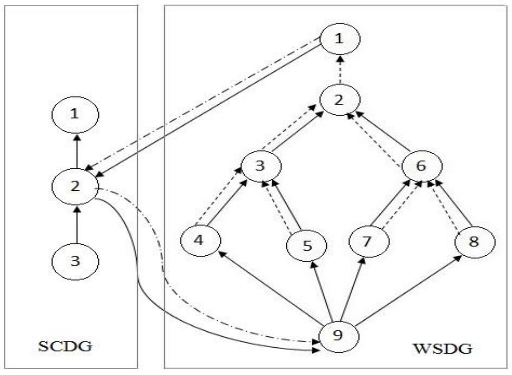


Fig. 9. Service Dependence Graph.

Here node represents statement number. Nodes in SCDG and WSDG are different. Solid edge represents data dependence, dashed edge represents flow dependence and dashed line edge represents callee dependence. So whole web slicing is represents as bellow.

The Service Dependence Graph contains Web Service Dependence Graph (WSDG) and Service Client Dependence Graph (SCDG). SDG represents the whole structure of the Web Service call, starting from Web Service Client to the Web Service.

Example: SDG for above example.

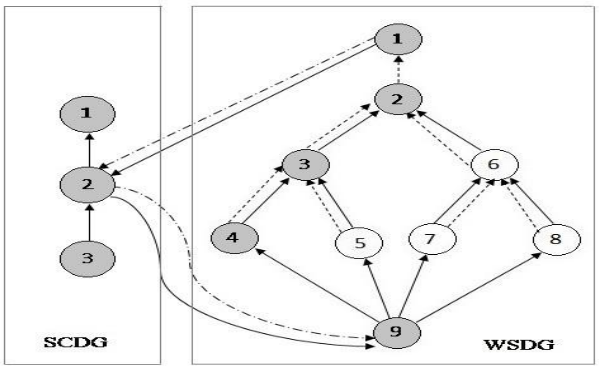


Fig. 10. Web slice for above web service. Here slice is represented as darken nodes. Left side of the graph is SCDG and right side of the graph is WSDG.

V. Algorithm for Web service Slicing(WSS)

Input: Web Service Code, Client code, Slicing Criterion (< S, V >)

Output: slice

Stage 1: Construction of SDG.

Stage 2: Compute Slice for WDSG.

Stage 1: Construction of SDG

a) Construct WSDG

```
1. Node construction
    a. Create two special nodes start stop
    b. For each statement s of Web Service do
        i. Create a node s
        ii. Initialize the node with its type, list of variables used or defined, and its scope
2. Add data dependence or control dependence or callee dependence
    For each node $u_i$ do following
    For each node $u_j$ do following
        a. Add data dependence edge ($u_i$, $u_j$), if one or more variables or data is used at node $u_j$ from node $u_i$
        b. Add control dependence edge ($u_i$, $u_j$), if control moves from node $u_i$ to node $u_j$
        c. Add callee dependence edge ($u_i$, $v_i$), if at node $u_i$ another Web Service is called to node $v_i$.
```

b) Construct SCDG

```
1. Node construction
    a. Create two special nodes start and stop
    b. For each statement s of Web Service, do
        i. Create a node s
        ii. Initialize the node with its type, list of variables used or defined, and its scope
2. Add data dependence or control dependence or callee dependence
    For each node $u_i$, do following
    For each node $u_j$, do following
        a) Add data dependence edge ($u_i$, $u_j$), if one or more variables or data is used at node $u_j$ from node $u_i$.
        b) Add control dependence edge ($u_i$, $u_j$), if control moves from node $u_i$ to node $u_j$.
```

c) Construct SDG

```
1. In SCDG if Web Service is called from node $u_i$ to node $v_i$ then add callee dependence edge ($u_i$, $v_i$)
2. If Web Service returns data from node $u_i$ to Service client node $v_i$ add callee dependence edge ($u_i$, $v_i$)
```

Stage 2: Compute Slice for WDSG.

```
1) For every variable v used at node $u_j$ from $u_i$ do
    a. Mark data dependence ($u_i$, $u_j$), if any data or variable used at node $u_j$ from node $u_i$.
    b. Mark control dependence ($u_i$, $u_j$), if flow of control depends on node $u_i$ from node $u_j$.
    c. Mark callee dependence ($u_i$, $v_i$), if Web Service is called from node $u_i$ node $v_i$.
2) Trace execution.
3) Remove unmarked node from Web Service.
```

4) Set of Marked nodes is dynamic slice.

VI. Future Work

There is different kind of program slicing techniques; among them we have explored dynamic program slicing technique to compute slice of Web Service. In future we will implement dynamic program slicing technique in SOA environment. We can apply different types of program slicing techniques to test Web Service. We can also apply hybrid slicing technique that uses more than one slicing technique to compute slice of Web Service. Moreover, we can implement different approach to implement graph for Web Service (e.g. WSDG, SCDG, and SDG). Thus we have explored Web Service quality by program slicing techniques. In future, we will implement the SDG in efficient manner to compute slice of Web Service.

VII. Conclusion

Program slicing is the technique to compute slice of the program and reduce size of the program according to slicing criterion. Applying dynamic program slicing to Web Service enables us to test Web Service internally by keeping watch on the variables used or flow of control as we do in debugging in tradition programming languages. This enables us to find errors easily in the program by applying dynamic program slicing technique to Web Service. And we can easily check errors if they exist. Thus it becomes possible to test the Web Service one by one statement. This may bring evolution in testing of Web Service.

References

1. Rajeev Kumar Durga Prasad Mohapatra, Rajib Mall. An overview of slicing techniques for object-oriented programs. Indian Institute of Technology, Kharagpur, page 2, 2005.
2. Joseph R. Horgan Hiralal Agrawal. Dynamic program slicing. ACM SIG-PLAN'90 Conference on Programming Language Design and Im-plementation, page 1, 1990.
3. Sylvia Ilieva Lachezar Ribarov, Ilina Manova. Testing in a service-oriented world. Proceedings of the International Conference on Infor-mation Technologies (InfoTech-2007), 1:4, 2007.
4. Andrea De Lucia. Program slicing: Methods and applications. University of Sannio.
5. Zayaraz Godandapani Poonkavithai Kalamegam. A survey on testing soa built using web services. International Journal of Software Engineering and Its Applications, 6, 2012.
6. Gias Uddin. Program slicing. Technical Report, CWI (Centre for Math-ematics and Computer Science) Amsterdam, The Netherlands, page 2, 1994.

Author(s) Profile

Harkishan Rathod, received the B.Tech degree in Information Technology and pursuing M.E degrees in Computer Science and Engineering from Government Engineering College, modasa in 2012 and 2013, respectively from Gujarat Technological University.

Kaushik Rana, is pursuing Ph.D degree from Gujarat Technological University, Gujarat. M.E in Computer Engineering from Dharmsinh Desai University, Nadiad. Presently he is working as Assistant Professor in Computer Engineering at Government Engineering College, Modasa, Gujarat.