A Business Process of Web Services Testing Method Based on UML2.0 Activity Diagram *

Zhang Guangquan[1,3] Rong Mei[2] Zhang Jun[1]
1. School of Computer Science and Technology, Suzhou University, Suzhou 215006, China
2.Shenzhen Tourism College,Jinan University，Shenzhen 518053，China
3.School of Mathematics and Computer Science Chongqing Normal University, Chongqing400047 ,China
Email: zgqxyz@sina.com; gqzhang@suda.edu.cn

Abstract:

In order to increase the reliability of web services business process, a BPEL modeling and testing approach based on UML2.0 activity diagram is presented. The key idea of the approach is to transform the business process interaction via BPEL code to visual and dynamic testing model. The approach extends UML2.0 activity diagram to describe the syntax and behaviors of BPEL. The formal specification and the definition of test coverage criteria of UML2.0 activity diagram are proposed. It enhances the restriction of test cases generation method and promotes the efficiency and accuracy of testing by reducing the amount of test cases reasonably. To illustrate this more thoroughly, this paper discusses the feasibility of automated testing using web services business process testing model with an example.

1. Introduction

Web services are applications that can be published, located and invoked across the Web. The web services framework consists of a set of XML standards and specifications for describing these services. The Business Process Execution Language for Web Services (WS-BPEL or BPEL for short) is a XML based programming language. The goal of BPEL is to make it possible to write a business process once in BPEL and then run it anywhere. A key research challenge in web services concerns web services testing. The main characteristics of web services influence testing activities are their asynchronous behaviors and distribution.

Bearing in mind these features, the goal of this paper is to describe a testing model to obtain test cases of BPEL [1]. The approach builds web services business process model by extending UML2.0 activity diagram and proposes the formal specification. It enhances the restriction of business process testing method and promotes the efficiency and accuracy of testing.

2. Business process testing model based on UML2.0 activity diagram

UML2.0 activity diagram is a special form of state machine, which can be used to model computing flow and work flow. In the activity diagram, the state machine and metamodel of activity is separated, the semantic of activity is based on the semantic of Petri Net. But UML2.0 activity diagram cannot build the business process testing model sufficiently. This paper uses stereotype and tagged value, both of which are UML2.0 extension

* Acknowledgements: This work is supported by the Natural Science Foundation of China (No. 60073020), the University Natural Science Foundation of Jiangsu Province ofChina (No. 05KJB520119) and the Natural Science FoundationProject of Chongqing (No.CSTC2006BB2259).
ZHANG Guang-quan (1965-), PhD, professor; main research fields: software testing and formal method.
RONG Mei (1966-), PhD, associate professor; main research fields: software architecture.

0-7695-3063-X/07 $25.00 © 2007 IEEE
DOI 10.1109/IITA.2007.83

mechanism, to promote the imprecision with formal specification. The stereotype can construct a new model based on the defined model. The tagged value can store the information of stereotype.

2.1 Mapping from BPEL primitive activity to UML2.0 activity diagram

BPEL primitive activities include <invoke>, <receive>, <reply>, <assign>, <wait>, etc. BPEL primitive activities can map to the action node of UML2.0 activity diagram with stereotype. The operation parameters and partner links can map to the information of the tagged value. <invoke> can map to CallOperationAction, which is used to do normal calling operation. <receive> can map to AcceptCallAction, which is used to wait for a call operation. <reply> can map to ReplayAction, which is used to recover after a call operation. <assign> can map to ReadVariableAction and WriteVariableAction, which are used to get or evaluate the value of variables. <wait> can map to AcceptEventAction, which is used to wait for event. With UML2.0 extension mechanism, BPEL primitive activities can map to the action node of UML2.0 activity diagram, which is illustrated in Figure 1. An example which describes the <invoke> element of BPEL with UML2.0 activity diagram is illustrated in Figure 2.

2.2 Mapping from BPEL structure activity to UML2.0 activity diagram

BPEL structure activities include <sequence>, <switch>, <pick>, <flow>, <while>, etc. With the definition of <sequence>, the executing sequence of primitive activity and structure activity can be defined. <sequence> can map to ControlFlow in UML2.0 activity diagram, which is used to describe the serial executing procedure of process. An example in which two activities execute orderly is illustrated in Figure 3. <switch> can map to DecisionNode, which is used to describe the branch executing procedure of process. An example of mapping from <switch> to UML2.0 activity diagram is illustrated in Figure 4.

<pick> can also map to DecisionNode. Different from <switch>, there are guard conditions in the branch path of activity diagram. An example of mapping from <pick> to UML2.0 activity diagram is illustrated in Figure 5.<flow> can map to ForkNode and JoinNode in UML2.0 activity diagram, which is used to describe the parallel executing procedure of process. An example of mapping from <flow> to UML2.0 activity diagram is illustrated in Figure 6. <while> can map to DecisionNode in UML2.0 activity diagram. It is similar to <pick>, the difference between them is that <while> is used to describe the circulation executing procedure of process. An example of mapping from <while> to UML2.0 activity diagram is illustrated in Figure 7.

3. The Business Process Testing method based on UML2.0 activity diagram

After modeling web services business process with UML2.0 activity diagram, generating test cases with the visual model is needed. As the imprecise specification of UML2.0, it needs to define the formal specification of UML2.0 activity diagram [6]. With the test coverage criteria and DFS (Deep First Search) method, test cases can generate

3.1 The formal specification of UML2.0 activity diagram

Definition 1: UML2.0 activity diagram is a quintuple as follows: AD=(S, T, G, C, F) where $S=\{ s_0, s_1 , s_2, \ldots s_n\}$ is nonempty set of actions with s0 is the initial action and s_n is the end action; $T=\{ t_0, t_1 , t_2, \ldots t_n\}$ is nonempty set of transitions; $G=\{ g_0, g_1 , g_2, \ldots g_n\}$ is set of guard conditions with g_i corresponding to t_i, the default true value of g_i is true; C is the set of current transition action, which is the subclass of S; $F=\{(s, g, t)| s\in S, g\in G, t\in T\} \cup \{(t, g, s)|t\in T, g\in G, s\in S\}$ is the subclass of $\{(S\times G\times T)\} \cup \{(T\times G\times S)\}$, F denotes the flow connection between actions and transitions.

Definition 2: Suppose that UML2.0 activity diagram is a quintuple as follows: AD=(S, T, G, C, F),

for random t∈T, 't={ s∈S | (s, g, t)∈F} is the preposition set of t, t'={ s∈S | (t, g, s)∈F} is the postposition set of t; e(C)={t | (s, g, t)∈F } is the set of transitions which is triggered by current action, f(C)={t | t∈e(C) and (C-'t) ∩t'=∅} is the set of transitions which is triggered only by current action, C'=(C-'t)∪t' is the new transition which is triggered by t, if the transitions satisfying the condition is non unique, it can choose a random transition which is not triggered.

Definition 3: Suppose that UML2.0 activity diagram is a quintuple as follows: AD=(S, T, G, C, F), TS is the test sequence of AD, ts∈TS. ts= $c_0 \xrightarrow{g_0 t_0} c_1 \xrightarrow{g_1 t_1} \ldots \xrightarrow{g_{n-1} t_{n-1}} c_n$, where $c_0 = s_0, c_n = s_n$, c_i is the current action, g_i is the guard condition, t_i =f(c_i), i ≥ 0 ; $c_i = \{(c_{i-1} - {}'t_{i-1}) \cup t'_{i-1}\}, i \geq 1$.

3.2 The test coverage criteria of UML2.0 activity diagram

In order to assure the adequacy of testing, the test coverage criteria of UML2.0 activity diagram are given as follows [7, 12]: (1) Action coverage criterion: Each action in each activity diagram must be tested at least once. (2) Transition coverage criterion: Each transition in each activity diagram must be tested at least once. (3) Basic path coverage criterion: Each basic path which starts from initial node in each activity diagram must be tested at least once. (4) Sequence coverage criterion: Each combination sequence of transition must be tested at least once.

3.3 The test cases generation method based on UML2.0 activity diagram

While obtaining the model of web services business process with UML2.0 activity diagram, the next procedure is searching activity diagram with DFS method. If there is a branch structure, the branch will decompose to different sequences. If there is a circulation structure, the circulation will execute only once. When all actions and transitions have been searched, it can obtain entire test sequences. The concrete procedure can be described as follows. First of all, it needs to begin with the initial node and search the activity diagram according to DFS method. If the current action is not empty, then the action needs to be pushed to the stack, the action counter adds one. If there is t_i in $e(c_i)$, then c_{i+1} equals c_i-'t+t', the search is going to the next step. If there is a circulation, the search is going to the next transition that can be triggered. At the same time, the actions counter equals to two. Then the current guard condition and transition are pushed to the stack, the transitions counter adds one. When there is no transition can be triggered with s_n, the test sequence can be obtained by analyzing the data in the stack. If the data in the stack are according with all guard conditions, then the data in the stack can constitute a test sequence, otherwise, the data in the stack will be cleared. If the test sequence violates the test coverage criteria, it needs to trace to the previous action and search the action that can be triggered. After getting the test sequences, the input data can be obtained with category partition method [9]. According to the method corresponding to actions and the guard conditions, the expectation output data can be computed, and then the test cases are generated.

4 Case Study

This section uses a book renewing business process to discuss the application of web services business process testing method based on UML2.0 activity diagram.

The process is part of online library management system. Figure 8 shows the testing model of the business process. In the testing model, the user renews books by interacting with the web services; the server invokes the corresponding web service with the command of users. The part of user login process BPEL code is showed as follows:

```
<sequence> < invoke partnerLink="UserLogin" portType="UserLoginPT"
    operation="sendinial" Variable="UserInfo"/>
< reply partnerLink="UserMain" portType="UserMainPT"
    operation="initiate" outputVariable="InUserInfo"/> </sequence>
```

A test case generation procedure according to section 2.3 is showed as follows. Detailed procedure of method executing is illustrated in table 1. While a whole search has been done, a test sequence is that ts_1 equals s_0 t_0 s_1 t_1 s_2 g_3 t_3 s_3 t_4 s_4 t_6 s_5 g_8 t_8 s_8 t_{14} s_{11} t_{18} s_n. By getting the method corresponding to the action in the activity diagram, the method sequences can be obtained. With category partition method, the input sequences can be obtained. With the guard conditions and the methods corresponding to the actions, the output sequences can be obtained. The input sequences, the method sequences and the output sequences compose a test case. Each test sequence generates a test case at least. A test case generated with ts1 is illustrated in table 2. The test case can test not only the control flow of the business process, but also the data flow based on the parameter variables and partner links stored in tagged value.

Tab.1 The procedure of test case generation

Tab.2 The test case of ts_1

i	c_i	$e(c_i)$	$f(c_i)$	c_{i+1}
0	s_0	t_0	t_0	s_1
1	s_1	t_1	t_1	s_2
2	s_2	t_2 ,t_3	t_3	s_3
3	s_3	t_4	t_4	s_4
4	s_4	t_6 ,t_7	t_6	s_5
5	s_5	t_8 ,t_{12}	t_8	s_8
6	s_8	t_{14}	t_{14}	s_{11}
7	s_{11}	t_{18}	t_{18}	s_n
8	s_n			

5 Related Work

The research of modeling BPEL received much attention from researchers. Fu, Bultan and Su [2] proposed a BPEL modeling method with finite state machine. Sebastian, Karsten and Christian [3] proposed a BPEL modeling method based on Petri Nets. But both models cannot be identified distinctly. Among all UML diagrams, numerous test methods have been proposed. Li, Wang and Qi [7] used WP method generating test cases based on UML statecharts. Wang,Yuan and Yu, et al[8] proposed an approach using Gray-Box method based on UML activity diagram. Li, Zhang and Dai [10] transformed UML sequence diagram to IRCFG and proposed test coverage criteria of IRCFG. Rountev, Kagan and Sawin [12] defined control flow test coverage criteria based on UML sequence diagram. This paper proposes a visual testing model through building the mapping from BPEL to UML2.0 activity diagram. With formal specification and test coverage criteria of UML2.0 activity diagram, the testing model can generate test cases efficiently based on restricted DFS method.

6 Conclusion

According to the characteristic of UML2.0 and BPEL, this paper proposes a web services business process testing model based on UML2.0 activity diagram. As combined with formal specification, the model promotes the semantic expression of UML2.0 activity diagram which is imprecise originally. Meanwhile, the executing procedure of DFS method is succinctly in the method. In order to make the test method more sufficiently, investigating test case generation based on the other UML2.0 diagrams is needed.

Input	Method Sequence	output
	Server.UserLogin()	welcome
1	User.InputAccount()	
	Server.YourBookList()	1.booklist
5	User.ChooseBook()	
	Server.CheckTime()	1.booklist.5
	Server.Prompt()	Time out
	Server.Prompt()	Thank you

Reference

[1] Bertolino A, Marchetti E, Muccini H . Introducing a reasonably complete and coherent approach for Model-based testing[J]. Theoretical Computer Science, 2005, 116(1): 85-97.

[2] Fu X, Bultan T, Su J . Analysis of interacting BPEL web services[C]//Stuart Feldman . Proceedings of the 13th

International Conference on World Wide Web .New York: ACM Press, 2004.621-630.

[3] Sebastian Hinz, Karsten Schmidt, Christian Stah . Transforming BPEL to Petri Nets[J]. Springer-Verlag, 2005, LNCS3649:220-235.

[4] Giuseppe A, Di.Lucca, Anna Rita Fasolino . Testing Web-based applications: The state of the art and future trends[J]. Information and Software Technology, 2006, 48(12): 1172-1186.

[5] Lipaev V V . A methodology of verification and testing of large software system[J]. Programming and Computer Software, 2003, 29(6): 298-309.

[6] Ghedira C, Mezni H . Through personalized web service composition specification: from BPEL to C-BPEL [J]. Electronic Notes in Theoretical Computer Science, 2006, 146(1):117-132.

[7] Li Liuying, Wang Ji, Qi Zhichang . A test case generation method for UML statecharts[J]. Journal of Computer Research and Development, 2001, 38 (6):691-697. (Ch)

[8] Wang Linzhang, Yuan Jiesong, Yu Xiaofeng, et al . Generating test cases from UML activity diagram based on Gray-Box method[C]//Alain Jean . Proceedings of the 11th Asia-Pacific Software engineering conference .Busan: IEEE Computer Society, 2004.284-291.

[9] Ostrand TJ , Balcer MJ . The Category-Partition method for specifying and generating functional tests[J]. Communication of ACM, 1988, 31(6):676-686.

[10] Zhang Guangquan, Li Lifeng, Rong Mei, Dai Ye. Testing of IRCFG coverage criteria for sequence diagrams[J]. Computer Science, 2006, 33(12):270-273. (Ch)

[11] Shin Nakajima . Model-Checking behavioral specification of BPEL applications[J]. Electronic Notes in Theoretical Computer Science, 2006, 151(2):89-105.

[12] Rountev A, Kagan S, Sawin J . Coverage criteria for testing of object interactions in sequence diagrams[C]// Mauca Cerioli . Proceedings of the 8th International Conference on Fundamental Approaches to Software Engineering .Edinburgh: Springer-Verlag, 2005.234-248.

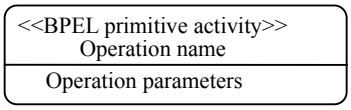

Fig.1 Mapping from BPEL primitive activity to UML2.0 activity diagram

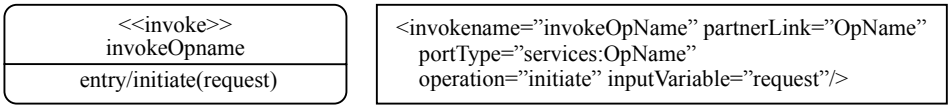

Fig.2 Mapping from <invoke> activity to UML2.0 activity diagram

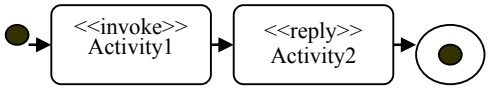

Fig.3 Mapping from <sequence> to UML2.0 activity diagram

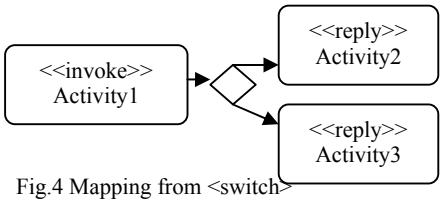

Fig.4 Mapping from <switch> to UML2.0 activity diagram

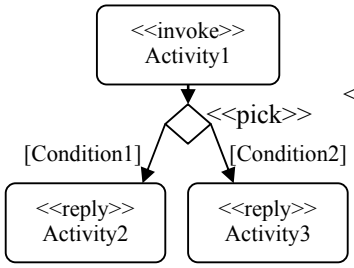

Fig.5 Mapping from <pick> to UML2.0 activity diagram

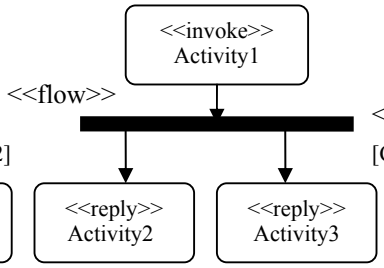

Fig.6 Mapping from <flow> to UML2.0 activity diagram

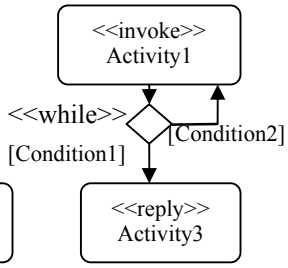

Fig.7 Mapping from <while> to UML2.0 activity diagram

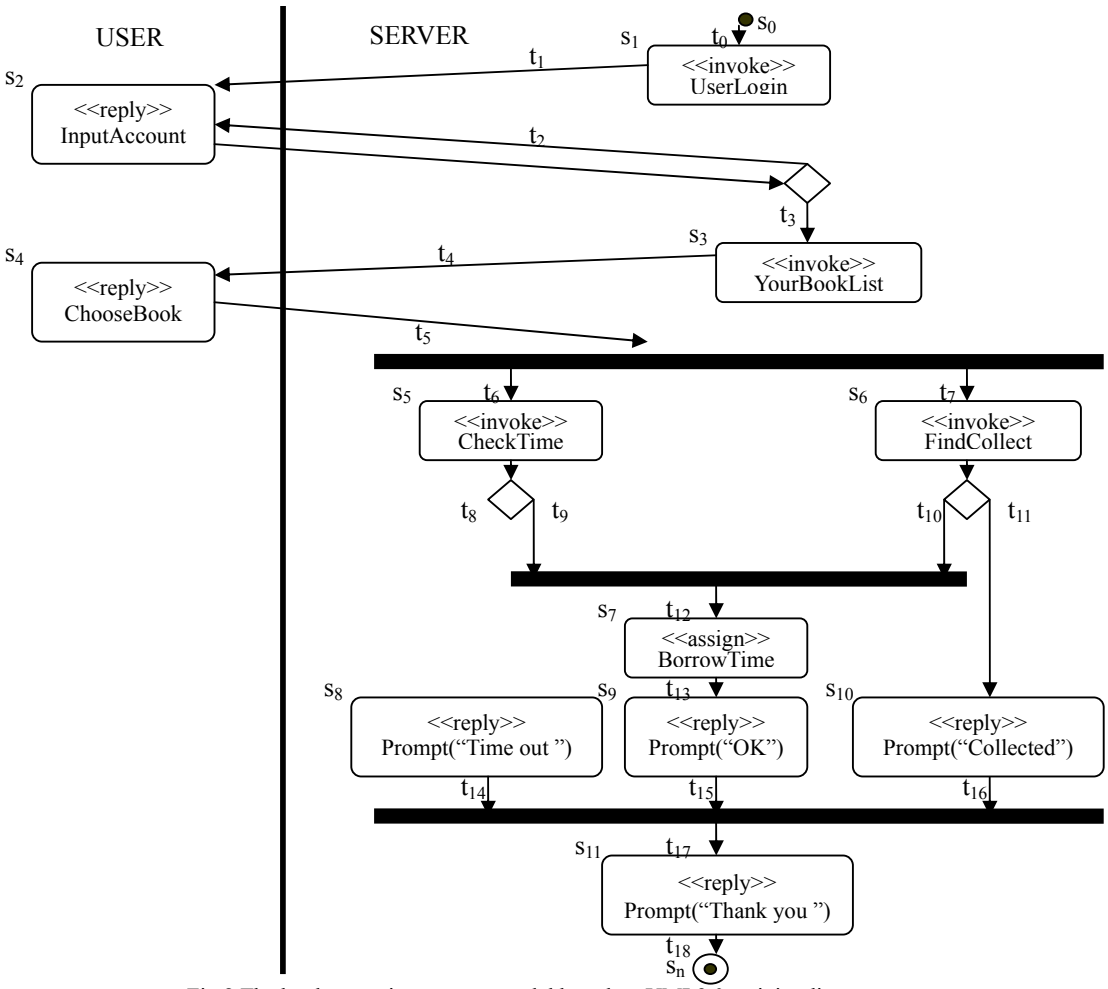

Fig.8 The book renewing process model based on UML2.0 activity diagram