基于WSDL文件生成测试用例的研究

李 玲,袁兆山,张 敏
(合肥工业大学 计算机与信息学院,安徽 合肥 230009)

摘 要:解析Web服务的WSDL文件,提取出相互独立的功能场景,以及各场景的条件-事件组合。对该条件-事件组合首先进行一致性分析,如果分析得到错误则返回错误信息,如果分析正确则继续对条件-事件组合进行完整性分析。根据完整性分析补充缺少的条件-事件组合,利用覆盖场景算法对补充的条件-事件组合进行合并,最终形成所有有效的场景。根据该场景的各个条件-事件组合生成测试用例。

关键词:Web服务;WSDL;一致性分析;完整性分析;场景;条件-事件组合
中图分类号:TP311.5 **文献标识码**:A **文章编号**:1673-629X(2007)10-0054-04

Research on Generation of Test Cases by Completeness and Consistency Analysis Based on WSDL

LI Ling, YUAN Zhao-shan, ZHANG Min
(School of Computer and Information, Hefei University of Technology, Hefei 230009, China)

Abstract: In this paper, analyze the WSDL file of the Web service at first. Then extract functional scenarios which are dependent with each other and condition-events of that scenario. Do consistency analysis on those condition-events. If the process fails, return error information. If the process successes, do completeness analysis on them. Based on the result of completeness analysis, makeup the missing condition-events. And use covered scenario algorithm to combine those missing condition-events and obtain the final effective scenarios. Based on those condition-events of that scenario, generate test cases.

Key words: Web services; WSDL; completeness analysis; consistency analysis; scenario; condition-event

0 引 言

Web服务(以下简称WS)是由URI标识的软件应用程序,其接口和绑定可以通过XML构件进行定义、描述和发现,WS使用基于XML的消息与其他软件应用程序直接交互。WSDL是WS描述语言,以XML的方式对Web服务的调用/通信加以描述。WS体现了新的开发思想和广泛的开放性和应用性。越来越多的开发者使用WS来提供实际的应用程序,因此对WS的质量也提出了更高的要求[1]。

1 概 述

WS由客户端、服务端和代理商三个部分组成,因此WS的测试也应该包括这三个方面,即WS客户端、WS服务端以及二者的集成[2]。此外WS可能是由多个WS提供者提供的WS组合而成,因此测试WS必须从两方面入手,用合法输入的测试用例验证其功能的正确性,用非法输入的测试用例验证其鲁棒性,从而确保WS的质量[3]。笔者提出一种方法,首先解析WSDL文件,提取条件-事件组合(在某些条件下,服务做出的行为,简称CE组合),然后分析CE组合的一致性和完整性,如果不符合一致性分析则返回错误信息,让服务提供者进行修改。如果不符合完整性分析,则补充丢失的CE组合并对需要补充的CE组合进行合并生成最终的有效场景。最后根据这些有效场景生成测试用例。该种方法主要从丢失的CE组合着手,它补充了当条件不成立或不完全成立的情况下,服务应该做出的正确反应,因此大大增强了服务的健壮性从而保证了服务的质量。

2 WSDL文件的解析

WSDL文件由两个部分组成:顶部分由抽象定义组成;底部分则由具体描述组成。抽象定义包括Types, Messages, PortTypes;具体定义包括Bindings, Services[4]。

收稿日期:2006-12-29
作者简介:李 玲(1980-),女,安徽淮南人,硕士研究生,研究方向为软件工程;袁兆山,教授,研究方向为软件工程、计算机网络。

* Types:独立于机器和语言的类型定义。Types元素包含了Types栏,声明Web服务需要的数据类型。

* Message:包括函数参数(输入与输出分开)或文档描述。Message元素包含了Messages栏。Message元素定义了Web服务中的操作的参数。Message元素分为输入参数、输出参数,以及兼作输入输出的参数。Message元素中的每个part子元素都和某个参数相符。每个part元素都有名字和类型属性。

* PortTypes:引用消息部分中消息定义来描述函数签名(操作名、输入参数、输出参数)。PortTypes元素可以有零个、单个或多个。PortTypes元素可在operation元素中定义一个或是多个操作。operation元素可以有一个、两个、三个子元素: input, output 和 fault元素。每个input和output元素中的消息都引用Message栏中的相关的Message元素。

* Bindings:PortTypes部分的每一操作在此绑定实现。Binding元素可以有零个、单个或多个。Binding元素制定每个operation的网络调用和回应。

* Services:确定每一绑定的端口地址。Services元素可以有零个、一个、多个。它还包含了port元素,每个port元素引用一个Bindings栏里的Binding元素。Bindings和Services栏都包含WSDL文档。

3　用一致性和完整性分析WSDL文件生成测试用例

3.1　概念介绍

(1) 场景:场景代表了一个或者几个CE组合。场景分为三类:功能场景,异常+异常处理,忽略场景。

(2)CE组合: 是指在某些条件(既可以是单个条件也可以是多个条件的组合)发生的情况下,服务会做某件事情。

(3) 完整性分析:通过分析所有的条件组合以及内部或者外部的事件来确认系统规格说明是否完整[5]。

(4)一致性分析:监测需求规格说明书是否明确指出条件-事件组合,以及条件是否会导致系统发生意外[6]。

3.2　分析WSDL文件生成测试用例

(1) 首先从WS的WSDL文件里提取相互独立的场景。相互独立的场景是指场景之间没有任何方式联系或相互影响。可以通过分析WSDL文件里描述WS的功能来获得初始的功能场景。只要两个功能之间没有任何相互关系,它们即可分别成为一个场景。

(2) 对每个场景分别从Message栏、PortTypes栏、Binding栏里提取CE组合,并且对每个CE组合首先进行一致性检测,即监测CE组合里的条件组合是否存在矛盾(某些条件不可能同时发生)。然后对其每一个条件考虑"非"的情况,得到新的条件组合(即完整性分析)。对这些新条件组合先进行一致性分析,如果符合一致性,则补充其应该发生的事件。如果不符合一致性,则将其归入忽略集合。

(3) 随着WS的条件的增多,需要补充的CE组合数量大大增加。因此提出覆盖场景算法来合并CE组合(算法见下文)。

(4)根据最终形成的场景生成测试用例,并将测试用例用XML表示。

覆盖场景算法:

1) 合并策略。

两组CE组合可以组合成一个覆盖场景的条件:

* 它们处理相同的事件。

* 它们有相同数量的条件,除了某个位置 p 不同外,其他位置的取值都一样(即除了某一个条件的取值不一样,其他条件相同)。

* 一组CE组合在位置 p 上取0,另一组取1。

2) 算法概述。

对于一个功能场景,K 为条件的数量,则缺少的CE组合可以被分为 $K \times K$ 个组,每个组记为 $G(r,t)$ 其中 $0 \leqslant r,t \leqslant k$。$G(r,t)$ 是指该组合里的CE组合有 r 个条件取值为1,t 个忽略条件。首先搜索有相同数量的忽略条件的组合(数量从0开始递增)然后在这些组合里找出只有1位不同的,根据合并策略将其组合。每当形成一个新的组合则删除原来2个CE组合 $(G(r_1,t_1),G(r_1-1,t_1))$,添加一个新的 $G(r_1-1,t_1+1)$。

```
For(n = 0;n <= K;n ++) //K 为条件的个数,n 表示忽略条件的个数
{
  For(i = 0;i <= K - n;i ++) //i 表示条件中"1"的个数,即按照条件中"1" 的个数的顺序进行场景合并
  {
    For(j = 0;j < Sizeof_ Group(i,n);j ++) //Sizeof_ Group(i,n) 是 Group(i,n) 里 CE 组合的个数,即对场景组合 Group(i,n) 里的每个 CE 组合进行合并
    {
      Pick up Group(i,n).Get(j)//选取当前 Group(i,n) 的第 j 个 CE 组合
      If(an appropriate covering scenario is found)
      //在 Group(i + 1,n) 里找到与 Group(i,n).Get(j) 匹配的 CE 组合
      {
```

```
    Combine it with current picked covering scenario, and
    //合并该两个CE组合
    Insert the covering scenario after combination into appropriate scenario Group(I, n + 1);
    //将新产生的CE组合放入Group(I,n + 1)
    Delete those two combined covering scenarios from their groups;
    //从Group(i,n)和Group(i + 1,n)里删除初始2个CE组合
   }
   Else//找不到匹配的CE组合
   {
    Store Group(i,n).Get(j) into final covering scenario set
    //将该CE组合作为最终的覆盖场景保存在结果集里
    Delete Group(i,n).Get(j)from Group(I,n);
    //从Group(i,n)里删除该CE组合
   }
  }
 }
}
```

4 试 验

4.1 试验一

淮北矿业集团煤矿环境监控与管理信息系统是为淮北矿业集团公司设计的一套以环境在线监控、环境信息管理和辅助决策为主要功能的系统软件。在该系统中利用一个WS来提供天气预报服务。由于篇幅限制,将该WS的WSDL文件内容按照WS客户端、WS服务端以及二者的集成三个部分归纳如下表:

系统	描述
WS服务器	当WS服务器接到WS客户端的查询某个城市的天气预报的请求后,如果可以获得该WS客户端的IP地址,并且服务器的数据库非空,WS服务器则发送该城市的天气预报给该客户端
WS客户端	在WS客户端向WS服务器发送查询某个城市的天气预报的请求后,客户端等待服务器的返回的该城市的天气预报,并且客户端能够正确显示该信息
WS服务器和客户端集成	在网络上有多个WS客户端。同一时间WS服务器只能接受一个WS客户端查询某个城市天气预报的请求。如果WS服务器能够获得该客户端的IP地址,并且数据库非空,则WS服务器向该客户端返回查询到的城市的天气预报信息。客户端能够正确显示该信息

4.1.1 分析该描述,提取所有独立的功能场景

(1)WS服务器:根据客户端发送的查询请求,可以提供相应的信息返回给客户端。

(2)WS客户端:客户端发送查询请求,在接到服务器端返回的信息后能正确显示。

(3)WS服务器和客户端集成:客户端发送查询请求,服务器收到请求,如果在同一时间只收到一个客户端的请求,则返回客户端需要的查询信息,客户端接到该信息后正确显示。

4.1.2 为每个功能场景提取条件

(1) WS服务器(三个条件):

a. 收到客户端发来的查询某个城市的天气预报的请求。

b. 可以获得该客户端的网址。

c. 服务器的数据库非空。

(2) WS客户端(三个条件):

a. 向WS服务器发送查询某个城市的天气预报的请求。

b. 收到WS服务器返回的该城市的天气预报的请求。

c. 能够正确显示该信息。

(3) WS客户端和服务器端集成(八个条件):

a. WS客户端向WS服务器发送查询某个城市的天气预报的请求。

b. WS服务器收到客户端的查询某个城市的天气预报的请求。

c. WS服务器同一时间只能接受一个客户端的请求。

d. WS服务器可以获得该客户端的IP地址。

e. WS服务器的数据库非空。

f. WS服务器向客户端发送该城市的天气预报的信息。

g. WS客户端收到WS服务器返回的该城市的天气预报的信息。

h. WS客户端能够正确显示该信息。

4.1.3 完成CE组合,并进行一致性和完整性分析

以WS服务器系统为例:

WS服务器系统可能发生的事件如下:

e1:WS服务器向客户端返回客户端请求的城市的天气预报信息。

e2:WS服务器空闲。

e3:WS丢弃客户端的请求。

e4:WS返回数据库为空信息。

WSDL文件中可以获得:abc->e1.

因此丢失的条件组合为:! a! b! c, ! a! bc, ! ab! c, ! abc, a! b! c, a! bc, ab! c。

其中! ab! c, ! abc不符合一致性分析。

补充其余缺少的CE组合:

! a! b! c->e2 ! a! bc->e2

a! b! c->e3 a! bc->e3

ab! c->e4

执行覆盖场景算法:

初始时:条件个数 $K=3$;

G(0,0)={! a! b! c->e2}G(1,0)={! a! bc->e2,a! b! c->e3}

G(2,0)={a! bc->e3,ab! c->e4}G(3,0)={abc->e1}

G(0,1)={}G(1,1)={}G(2,1)={}

G(0,2)={}G(1,2)={}

G(0,3)={}

经过该算法后生产的覆盖场景为(4个):

! a! bx->e2,a! bx->e3,ab! c->e4,abc->e1。(x表示不关心该条件的取值)

其中! a! bx->e2场景由于其现实意义为可忽略场景。

所以只要针对a! bx->e3,ab! c->e4,abc->e1这三个场景生成测试用例即可。

然后将测试用例用XML语言表示。

同理对客户端系统、WS服务器和客户端集成系统的CE组合进行完整性分析并执行覆盖场景算法,结果如下表:

描述	功能场景数	条件数	补充CE组合数	有效场景数
WS服务器	1	3	5	4
WS客户端	1	3	5	4
集成系统	1	8	127	12

4.2 试验二

股票查询WS:WS客户端可以向服务器发送查询所有股票价格的请求或者发生查询某一股票价格的请求。WS服务器根据WS客户端的请求或者发送所有股票价格或者发送一只股票一段时间内的股票价格,同一事件WS服务器只能接受一个客户端发送的请求。执行结果如下表:

描述	功能场景数	条件数	补充CE组合数	有效场景数
WS服务器	2	3	12	4
WS客户端	2	3	12	4
集成系统	2	7	52	10

比较两个实验的集成系统数据,结果见图1。

根据上述试验可以看出随着条件个数的增多,需要补充的CE组合数量大量增加(以2的n次方上升),而覆盖场景算法根据正则表达式的计算方法,合并CE组合但不减少其覆盖的效果,因此执行覆盖场景算法后,生成的有效场景数量仅以几何级数增长。从而在保证条件覆盖的情况下,大大减少了测试用例的数量。另外该方法从WS服务器、WS客户端,以及二者的集成三个方面考虑(包括网络应答方面的情况),对每个条件都考虑其不满足的情况,对每种条件组合都做了相应处理,或者补充,或者归入忽略集合,所以测试的覆盖范围非常全面。

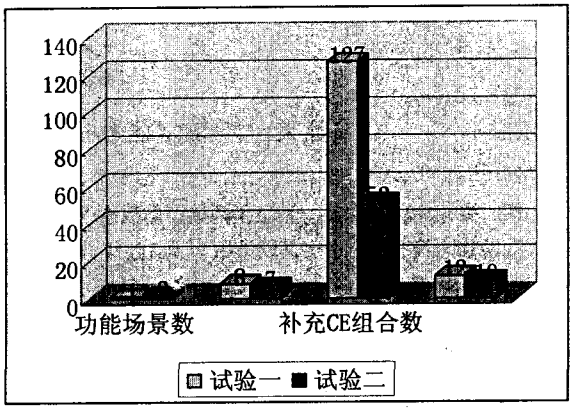

图1　两个实验中的集成系统数据比较

5 结束语

介绍了一种从WSDL文件析取CE组合,并对CE组合先进行一致性分析,然后进行完整性补充的方法。对缺少的CE组合进行覆盖场景算法,生成最终的有效场景。最后根据该场景生成测试用例。该方法从正反两个方面对WS进行了测试,从而验证了WS的功能的正确性以及鲁棒性。另外在该方法里,虽然完整性分析里确认缺少的CE组合很多,但是最终生成的有效覆盖场景的数量却有限。这在保证测试的完整性基础上极大减少了生成的测试用例的数量。但是完整性补充的工作随着条件的增多将呈指数级别增长,因此对于大型系统将会耗费很多时间。

参考文献:

[1] 杨芙清,梅　宏,吕　建,等.浅论软件技术发展[J].电子学报,2002,30(12A):1901-1906.

[2] Clarke E,Grumberg O,Peled D. Model Checking[M].[s.l.]:MIT Press,2002.

[3] Xu Wuzhi,Offutt J,Luo Juan. Testing Web Services by XML Perturbation[M].[s.l.]:ISSRE,2005:257-266.

[4] Bai Xiaoying,Dong Wenli. WSDL-Based Automatic Test Case Generation for Web Services Testing[M]. Washington,DC,USA:IEEE Computer Society,2005:215-220.

[5] Tsai W T, Wei X, Chen Y, et al. Developing and Assuring Trustworthy Web Services[M].[s.l.]:ISADS,2005:43-50.

[6] Heitmeyer C L, Jeffords R D, Labaw D G. Automated Consistency Checking of Requirements Specifications[J]. ACM Transactions on Software Engineering and Methodology,1996,5(3):231-261.