

Web 服务测试的一种实现

秦 锋, 李 乔, 郑 啸

(安徽工业大学 计算机学院, 安徽 马鞍山 243002)

摘 要:随着 Web 服务技术的不断发展和广泛应用,需要运用测试技术来保障 Web 服务的正确有效运行。基于 Apache Axis, 解析 WSDL, 用 CTM 产生有效测试例, 实现了 Web 服务的功能性测试和压力测试, 并就安全性测试、授权测试、多个 Web 服务测试方面还需完善的工作做了进一步探讨和展望。

关键词:Web 服务; Axis; 分类树方法; 功能性测试; 压力测试

中图分类号:TP311.5

文献标识码:A

文章编号:1673-629X(2007)08-0239-04

A Kind of Implementation of Web Services Testing

QIN Feng, LI Qiao, ZHENG Xiao

(School of Computer Science, Anhui Univ. of Tech., Ma'anshan 243002, China)

Abstract: With the development and extensive application of WS technology, testing technology is needed to make WS highly reliable and run effectively. WSDL is analysed based on Apache Axis, yield test cases by means of classification tree method (CTM) and implement functional testing and stress testing of Web Services. Finally, further problems such as security testing, authorization testing and testing of several Web Services in the fields and the development trends are discussed.

Key words: Web services; Axis; CTM; functional testing; stress testing

0 引 言

随着网络技术和面向对象技术、分布式计算的进一步的融合发展,从 COM, COM + /DCOM, CORBA 到 Web 服务,网络应用正朝着 SOA(Service Objected Architecture)^[1]软件体系结构方向转变。Web 服务实现了“基于 Web 无缝集成”的目标,它是一种可透过网络存取、由多个应用程序组件组合所构筑出来,相互为用的环境。Web 服务通过定义好的接口使用标准的 Internet 协议进行交互,具有封装性好、耦合度低、可集成能力强等优秀的特性。以 XML 形式存在的 SOAP 协议被封装在 HTTP 等 TCP/IP 应用层协议中,它是 Web 服务组件间的标准通信协议。新应用技术出现的同时,也需要新的测试技术。测试技术能为服务软件的性能、质量、可靠性等方面提供有力的支持。由于 Web 服务是一种包含大量运行行为的分布式应用,因而不可能完全沿用传统软件测试技术对其进行测试。在目前 Web 服务测试的研究领域中,主要开展的研究工作有易测试性研究、测试数据的生成研究和测试过

程管理研究。文献[2]从输入和输出的依赖关系、调用顺序、层次功能描述和并发顺序规范四个方面对 WSDL 文件进行了扩展,以利于测试数据的生成。文献[3]实现了使用 TTCN-3 描述的测试数据的自动生成,文献[4]则是基于合约变异的技术,实现一种有效测试数据的自动生成方法。文献[5]对 Web 服务测试报告的管理方法进行了研究,提出了一种实现方法和框架。通常,Axis 能够开发调用 Web 服务的客户端和部署 Web 服务,一般的客户端调用 Web 服务程序,都是基于特定的 Web 服务^[6]来开发的,即使测试,也是针对某个特定对象的 Web 服务,因此,不具有通用性。笔者研究 Web 服务测试实现技术,通过对 WSDL 文件的解析,运用分类树方法(Classification Tree Method, CTM)获得有效测试数据,然后基于 Axis 来实现 Web 服务的功能性和压力测试。

1 Axis 及框架

Axis(Apache Extensible Interaction System)^[7]是 Apache 组织推出的 SOAP 引擎,非常适合于请求/响应的消息模式。

Axis v1.1 软件包目前版本支持的标准是:W3C SOAP 1.1 和 1.2;WSDL 1.1;SAAJ 1.1(SUN SOAP

收稿日期:2006-11-07

基金项目:安徽省教育厅自然科学研究项目(2006KJ064b)

作者简介:秦 锋(1962-),男,安徽和县人,教授,研究方向为数据挖掘、计算机软件理论和网络应用技术。

with Attachments API for Java); JAX-RPC(SUN Java API for XML-Based RPC)1.0。Axis 给出了一个很好的 SOAP 实现平台,通过它开发 Web 服务应用省掉了许多底层技术细节,从而使 Web 服务的开发变得轻松高效。有很多流行的开发工具都使用 Axis 作为其实现支持 Web 服务的工具,如著名 JBuilder 以及 Eclipse J2EE 插件 Lomboz。图 1 是 Axis 核心引擎的体系结构图。

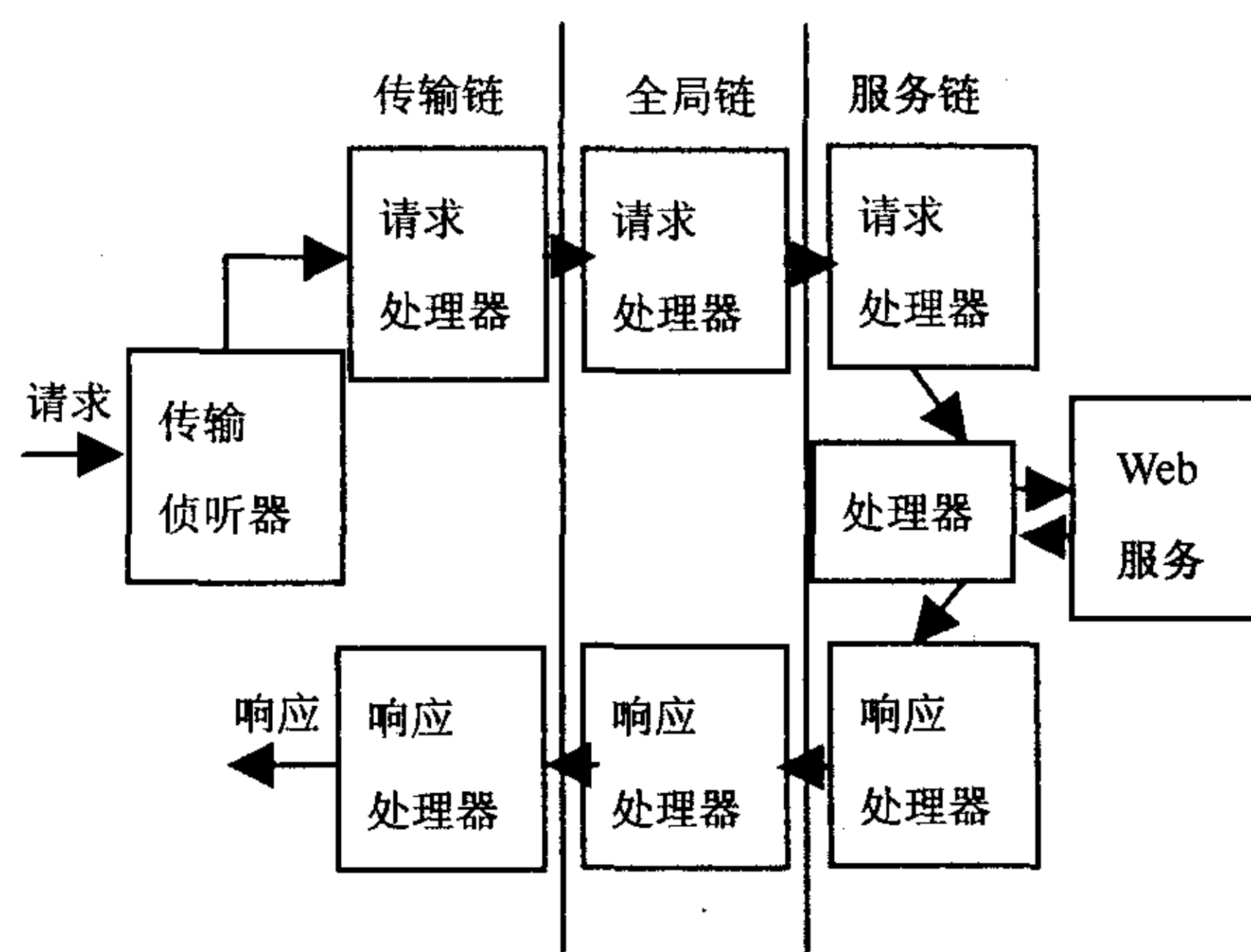


图 1 Axis 引擎

Axis 项目包括以下几个部分：

(1) 消息流子系统。

消息流子系统提供了灵活的消息传递框架,这个消息传递框架包括处理程序、链、序列化程序和反序列化程序。链表示的是有序的处理器的集合。Axis 中定义了三种类型的链:特定传输链、全局链、服务链。特定传输链只是在 SOAP 消息在使用某个特定的传输协议时才被调用,而全局链是所有的 SOAP 消息流都必须经过的链,它是所有 Web 服务都需要相同的处理时调用的链。服务链只是在调用一个特定的服务时才调用链中的处理器。

处理程序可以被组合在一起成为链,并且能够使用一个灵活的部署描述符来配置这些处理程序的顺序。如图 1 中所示,请求和响应分别构成了两条链路。

(2) 传输框架子系统。

提供了一个传输框架,这个传输框架可以帮助你创建自己的可插式传输发送器和传输侦听器。

(3) 数据编码子系统。

Axis 完全按照 XML Schema 规范提供各种数据类型的自动序列化,并且提供功能扩展接口来使用你自己定制的序列化器和反序列化器。

(4) 其他。

Axis 完全支持 WSDL 以及日志记录、出错以及故障处理机制,而且它提供一些工具用来解析 WSDL 文档,并将 WSDL 文档转换成客户端的调用框架以及根

据类来产生 WSDL 定义文档。

2 测试模型和实现

2.1 Web 服务的测试执行框架

图 2 是 Web 服务测试执行的框架,它由四部分组成:测试主管(Test Master)、测试代理(Test Agents)、测试例(Test Case)和测试监视器(Test Monitor)。这四部分可以同处一地,但是经常是分布式地位于不同的地方,并且它们彼此之间使用 SOAP 进行通讯。

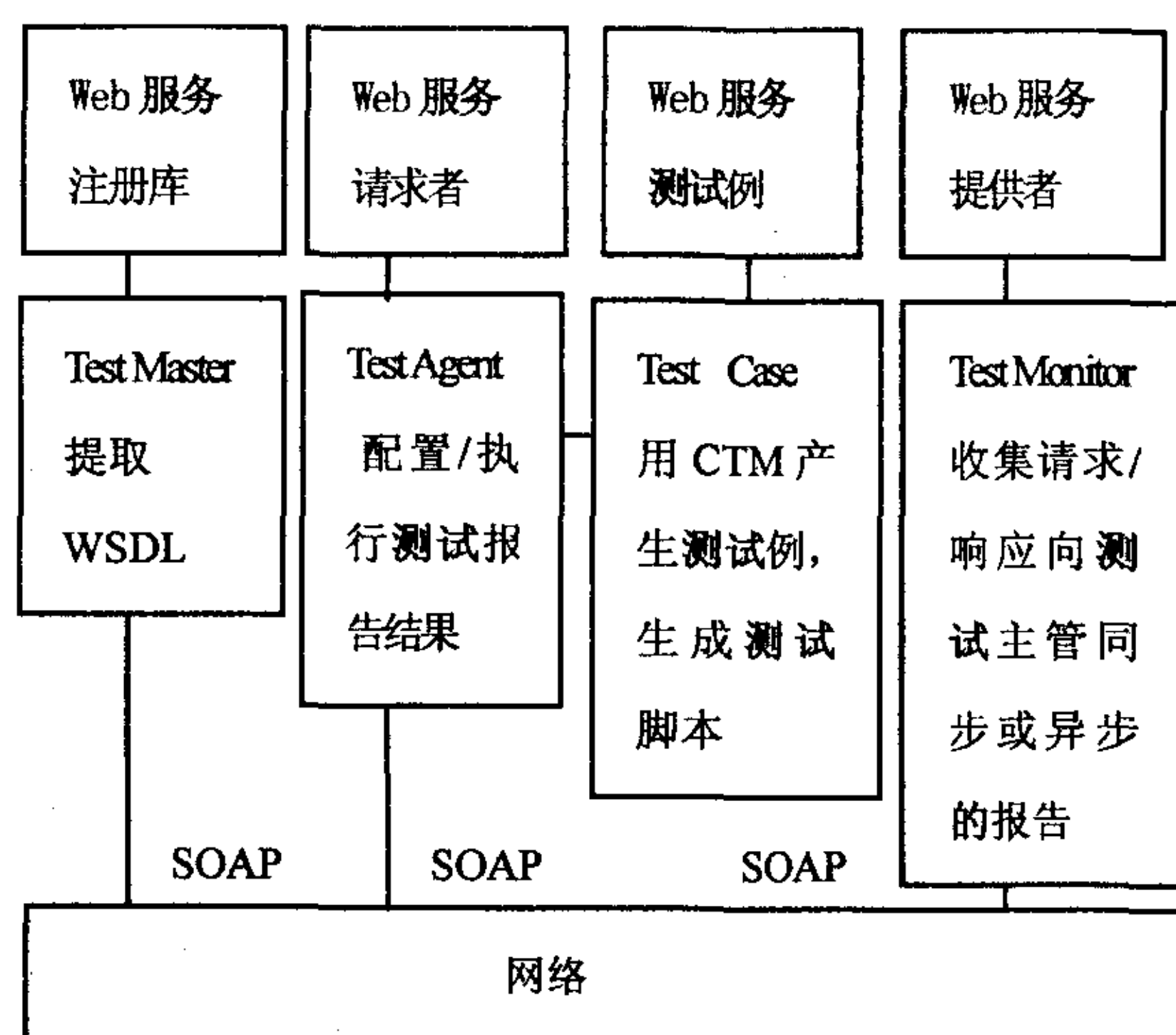


图 2 Web 服务测试执行框架

(1) 测试监视器经常驻留在与被测的 Web 服务同样的地方。测试监视器捕获在 Web 服务和测试代理客户端的信息,并且记录状态的变化。

(2) 测试主管通过从 Web 服务注册库中得到 WSDL 文档,并且通过使用 SOAP 并行的发送命令到一个或更多测试代理那里运行。它也同步或异步地接受来自于一个或更多测试监视器的数据。

(3) 测试代理担当在 Web 服务方面测试主管的代理,它执行测试,验证结果,并且报告响应结果给测试主管。测试代理会先在注册库中搜寻需要的 Web 服务,如果一个测试代理也和被测的 Web 服务在同一位置的时候,它也能担当一个测试监视器的角色。

(4) Web 服务测试例的产生,它通过测试代理解析出的 WSDL 文档,得到调用 Web 服务需要输入的参数,用 CTM 产生有效测试例,然后再由测试代理配置执行测试。

2.2 测试例的生成和选择

WSDL^[8]是描述 Web 服务的标准 XML 格式,它用一种和具体语言无关的抽象方式定义了给定 Web 服务收发的有关操作和消息。WSDL 信息模型分离了服务接口定义(抽象接口)与服务实现定义(具体端点)。抽象接口规范描述了终端的处理能力,它在 WSDL 中表示为 portType。绑定机制用特定的通信协议、数据编码模型和底层通信协议,将 Web 服务的抽象定

义映射至特定实现。如果绑定结合了实现的访问地址,抽象端点也就成为可供服务请求者调用的具体端点,WSDL的 port 元素表示了这一结合。

抽象接口可支持任何数量的操作。操作是由一组消息(messages)定义,消息定义了操作的交互定式。与抽象的消息、操作概念相对应的具体实现是由 binding 元素指定。在 WSDL 中,Web 服务描述中的主要元素如下:

(1)Types。定义了 Web 服务使用的所有数据类型集合,可被元素的各消息部件所引用。

(2)Message。通信消息数据结构的抽象类型化定义。使用 Types 所定义的类型来定义整个消息的数据结构。

(3)Operation。对服务中所支持操作的抽象描述。

(4)portType。对某个访问入口点类型所支持操作的抽象集合。

(5)Binding。包含了如何将抽象接口的元素转变为具体表示的细节,具体表示是指特定的数据格式和协议的结合。

(6)Port。定义为协议/数据格式绑定与具体 Web 访问地址组合的单个服务访问点。

(7)Service。这是一个粗糙命名的元素,代表端口的集合,它是相关服务访问点的集合。

因此,portType 描述了 Web 服务是什么,binding 元素描述了如何使用 Web 服务,port 及 service 元素描述了 Web 服务的位置。WSDL 文档的 URL 就是 Web 服务本身的 URL 在末尾附加“? wsdl”。在客户端代码中请求 Axis 生成 WSDL 文档,制定想要的具体服务、端口以及操作,而 Axis 将能够从 WSDL 文档中抽取其他附属信息,不用自己定义。

由 WSDL 中得到能够使用的具体 Web 服务,以及调用每个 Web 服务需要输入的具体参数后,就能够通过 CTM 方法产生测试例,该方法是针对功能性测试中面向黑盒的,产生测试例的一种具体方法。通过 CTM,一个 SUT(System Under Test)的输入和输出域的各个方面被划分成多个有效输入类,因此各个类之间的每种组合就是一个测试例,这其中每个类之间的某种耦合或关联会减少有效测试例的数量,这样会很大程度上提高产生测试例的效率。目前,软件 CTE (Classification - Tree Editor)^[9]支持 CTM,通过该 CTE 可以实现 CTE,自动产生测试例。

2.3 用 Axis 实现测试过程

由于 Web 服务的通信协议多采用标准的 SOAP 协议,因此 Axis^[10]中提供了一套 API 来实现 SOAP 的封装、传输、解析。Axis 作为 SOAP 处理机,它的处理

对象就是 SOAP 消息。Axis 中对 SOAP 消息的处理是通过在处理器间传送一个 MessageContext 对象来完成的。MessageContext 含有处理请求消息或响应消息所需的所有数据,其中关键数据有:一个请求消息、一个响应消息以及当前消息处理的所有元数据。每个处理器都可以访问 MessageContext 的所有数据,因此,每一处理器都可以使用、修改请求消息及任何可能存在的响应消息。

(1)对于 SOAP 的封装器,首先可以调用 org. apache. axis. wsdl. gen. Parser 和 org. apache. axis. wsdl. symbolTable. *,对不同 Web 服务中提供的 WSDL 文件进行解析,并通过调用自定义的 Invoker 类中 listServiceNames()、listOperationNames()、listParameters() 方法分别存储 Web Services 的名称、操作名、入口参数。org. apache. axis. client. Call 和 org. apache. axis. client. Service 是两个比较常用的类,一般的客户端程序要访问一个 Web 服务时,都要生成一个客户端的 Service 对象和 Call 对象,在访问服务之前,首先要对 Call 对象设置相应的参数,包括服务的位置、操作名、入口参数、返回值类型等,可以用刚才方法解析出来并存储的 Web 服务的名称、操作名、入口参数放到 Vector 对象中,接着调用 Call 对象的 invoke 方法访问服务,而 invoke 方法的初始化参数就是刚才 Vector 对象经过排序化的数组。

通过以上步骤,Web 服务应用程序通过指出目标 URL、服务的名称、方法的名称、输入参数和返回值等使用 Call 对象建立一个请求并发出调用。传输监听程序监听到调用信息后,将上述中指定的数据打包到 Message 对象当中,然后将该 Message 放入到 MessageContext 对象(org. apache. axis. Message)之后,传输监听程序还要加载各种属性值,此时 SOAP 的封装也就完成。

(2)接着调用 Web 服务的请求传送给引擎全局链,在此进行有关全局的 Web 服务输入等配置后将其传送至服务处理程序。服务处理程序通过设置 MessageContext 中的 ServiceHandler 字段来选择相应的程序来处理指定的服务。Web 服务处理完毕后返回至服务客户端处理程序,然后按照原路逆行方向将最终结果返还给应用程序,这就是 SOAP 的传输过程。

(3)当成功发送出去 SOAP 数据后,就需要对调用 Web 服务的返回数据结果加以接收,这就要对 SOAP 封装的数据加以解析并实现,可以通过调用 Call 对象的 getOutputParams 方法得到一个 Map 对象,然后用迭代器 Iterator 分别解析出返回的参数名和参数值,并利用返回参数名作为 HashMap 对象键值,用 HashMap 对

象的 put 方法把返回参数名和参数值放入数据集合 HashMap 对象中,一旦 Call 对象的 invoke 方法调用成功以后,返回的这些参数名和参数值则放入 HashMap 中,再用迭代器 Iterator 并用数据集合 HashMap 对象的参数名作为键值就能取出返回的参数值。基于以上的实现步骤,就能够实现 Web 服务的动态调用,也就完成了 Web 服务中功能性测试的要求,你可以检查服务是否能够正确执行它的各个功能,如果这些测试失败,通常就意味着检测到了产品的一个基本问题。

压力测试可以基于以上的功能性测试,它不但测试服务对并发请求的响应,而且可以确认测试压力是否导致产生功能上的问题。对 Web 服务进行压力测试时,目的是要弄清楚被测试的 Web 服务在被施加了某些高强度压力的情况下仍然继续正常运行。通过时间触发类 Timer,来控制每秒调用 Web 服务的频率次数,Timer 时间设置的越短,也就意味着调用 Web 服务的频率就越高。通过 Thread 类,模拟多用户调用 Web 服务,来查看每个用户(Thread)调用 Web 服务的平均响应时间。

3 结束语

Web 服务是未来网络应用的发展方向,SOAP 和 WSDL 是 Web 服务的核心协议,Axis 给出了一个很好的 SOAP 实现,它使得开发和测试 Web 服务应用变得有保障和有效。文中实现了 Web 服务的功能测试和压力/负载测试的原型系统,在对单个服务 Web 服务进行测试的时候取得了很好的效果,在以后的研究工作中还将进行复杂的包含多个 Web 服务的系统的测试。同时,由于 Web 服务在使用中传送的消息可能会丢失、被窃取或是修改,因此面临着安全风险的问题。实际上这涉及到机密数据信息的输入、保存、处理和传递的一系列问题。对于此问题的自动测试是非常困难的,但是测试者应该根据真正的环境和协议来仔细设计测试例。WS-Security1.0^[11]规范提供了 Web 服务安全的实现方式,如 XML DIGSIG、XML 加密。信息

的保护只是安全性方面的一个内容。通过合适的实体(用户类别),来对 Web 服务中重要信息进行有效存取也是尤为重要的。因此,Web 服务的授权测试也是非常必要的。授权测试应该测试每一个使用 Web 服务的用户类别来确定是否对每类用户进行了正确的授权。以上都是以后测试工作中需要进一步完善的内容。

参考文献:

- [1] Sprott D, Wilkes L. Understanding Service - Oriented Architecture[EB/OL]. 2004 - 02. <http://msdn.microsoft.com/architecture/soa/default.aspx>.
- [2] Tsai W T, Paul R, Wang Y, et al. Extending WSDL to Facilitate Web Services Testing[C]//Proceedings of the 7th IEEE International Symposium on High Assurance Systems Engineering. Tokyo:[s. n.], 2002:171 - 172.
- [3] Schieferdecker I, Stepien B. Automated Testing of XML/ SOAP based Web Services[C]//Proceedings of the GIFAchtung. Kommunikation in Verteilten Systemen. KIVS. Leipzig, Germany:[s. n.], 2003:43 - 54.
- [4] 姜 瑛, 辛国茂, 单锦辉, 等. 一种 Web 服务的测试数据的自动生成方法[J]. 计算机学报, 2005, 28(4): 568 - 577.
- [5] Luo ling. BAI Xiaoying. Web services - Based Test Report Generation[J]. Tsinghua science and technology, 2005, 10(3):282 - 287.
- [6] 冉玉春, 赵凌燕, 郝 锐, 等. 用 Apache Axis 开发 Web 服务[J]. 计算机应用, 2004, 24(5):140 - 143.
- [7] Apache Software Foundation. Axis User's Guide[EB/OL]. 2004 - 02 - 08. <http://ws.apache.org/axis/java/user2guide.html/>.
- [8] W3C note, WSDL(Web Service Description language)[EB/OL]. 2001 - 03. <http://www.w3.org/TR/wsdl.html>.
- [9] Classification Tree Editor[EB/OL]. 2005 - 08 - 09. <http://www.systematic-testing.com/>.
- [10] Graham S. 用 Java 构建 Web 服务[M]. 北京:机械工业出版社, 2003.
- [11] Galbraith B, Hankison W. Web 服务安全性高级编程[M]. 北京:清华大学出版社, 2003.

(上接第 238 页)

C# 在 Windows 窗口的编程及 C# 在 ASP.NET 中的模块设计、交叉应用,相信具有一定的借鉴性。

参考文献:

- [1] 马风格, 梁 夏. 基于 B/S 模式的电力通信网监控系统的开发[J]. 计算机技术与发展, 2006, 16(10):177 - 178.
- [2] 徐保民, 杨铨玖, 胥爱军. 数据库系统原理与应用[M]. 北京:清华大学出版社, 北京交通大学出版社, 2005.

- [3] 微软公司. 面向 .NET 的 Web 应用程序设计[M]. 北京:高等教育出版社, 2004.
- [4] 王华杰, 黄 山. C# 数据库编程[M]. 北京:科学出版社, 2003.
- [5] Price J. C# 数据库编程从入门到精通[M]. 北京:电子工业出版社, 2003.
- [6] McManus J P, Kinsman C. C# 开发人员指南:ASP.NET、XML、Web 服务与 ADO.NET[M]. 常晓波, 朱剑平译. 北京:中国电力出版社, 2003.