

基于 AOP 和 Web 服务的多层分布式系统

孟凡新¹, 张京军², 刘光远¹

(1. 河北工程大学信息与电气工程学院, 邯郸 056038; 2. 河北工程大学科研处, 邯郸 056038)

摘要: 把面向方面编程的思想引入到 Web 服务中, 将其横切关注点(日志、事务、权限认证等)实现为方面服务, 在系统执行时用编织器在 SOAP 消息层编织 Web 服务和方面服务来构建最终系统, 并结合面向服务软件体系结构和 J2EE 的多层架构提出一个基于方面服务和 Web 服务的软件架构。给出一个应用该软件架构的实例。

关键词: Web 服务; 横切关注点; 面向服务体系结构; 多层架构; 面向方面编程

Multi-tier Distributed System Based on AOP and Web Services

MENG Fan-xin¹, ZHANG Jing-jun², LIU Guang-yuan¹

(1. College of Info & Electric Engineering, Hebei University of Engineering, Handan 056038;

2. Scientific Research Office, Hebei University of Engineering, Handan 056038)

【Abstract】 Aspect-Oriented Programming(AOP) is introduced to Web services, whose crosscutting concerns such as logging, transaction, authorization, etc. They are separated and realized individually as aspectual services. When the system runs, a weaver weaves them at SOAP message level to construct the final system. Combining Service-Oriented Architecture(SOA) and J2EE multi-tier architecture, a multi-tier system architecture based on aspectual services and Web services is proposed. And a case is designed according to the system architecture.

【Key words】 Web services; crosscutting concerns; Service-Oriented Architecture(SOA); multi-tier architecture; Aspect-Oriented Programming (AOP)

1 概述

软件应用结构经历了从单层应用到两层结构以及现在倡导的多层结构的演变。分布式多层结构模型因为有着可伸缩性、可配置性、灵活可靠性以及跨平台和体现软件集成的思想而成为广泛使用的软件应用模型^[1]。随着网络的发展和应用程序的扩大, 使用 XML 作为数据格式和 SOAP 协议实现远程对象访问的 Web 服务^[2]成为面向服务体系结构(Service-Oriented Architecture, SOA)^[3]的最好实现技术, 它可以消除使用不同组件模型、操作系统和编程语言的系统之间存在的差异, 真正以松耦合的方式实现分布式应用之间的互操作性^[4]。Web 服务是一种部署在网络上的对象构件, 用于构建互联网上大型的、复杂的分布式应用系统。越来越多的分布式应用通过集成第三方的 Web 服务以提高系统的开发效率和可扩展性。目前两大主流分布式开发架构 J2EE 和 .NET 都支持 Web 服务技术^[5]。Web 服务中除了核心业务关注点外, 还有一些横切关注点, 如日志、事务、权限认证等, 其相关的实现代码分散在各个核心业务方法中, 导致代码混乱和代码分散。另外, 一个 Web 服务就像一个黑匣子, 可以被任何应用系统、在任何地方被动态访问, 而不必考虑服务的具体实现, 而当客户端对服务需求发生改变时, 服务提供者需要修改原来的 Web 服务并重新编写、编译、测试, 最后在 Internet 上部署。由于它可能被多个应用系统使用, 对它的修改将影响多个系统, 导致 Web 服务的适应性和重用性降低, 同时给应用系统的维护和扩展带来困难。

面向方面编程(Aspect-Oriented Programming, AOP)^[6]提供了一种明确捕获和模块化横切关注点(日志、事务、权限认证等)的机制, 有效解决了由于横切属性引起的代码分散和代

码混乱等问题, 提高了系统的扩展性和维护性^[7]。文献[8]提出一个基于 Web 服务的软件架构, 能很好地指导软件开发, 但是它没有考虑 Web 服务中的横切关注点。文献[9]等用 AspectJ 实现了一个 SOA 应用中的横切关注点, 使系统能满足客户端的需求变化。本文把面向方面编程的思想引入到应用的组成部分——Web 服务中, 将横切关注点实现为方面服务, 并结合 SOA 和 J2EE 多层架构提出一个基于方面服务和 Web 服务的多层软件架构, 提高了 Web 服务的适应性和重用性, 使由其构建的系统结构更易理解、维护和扩展, 最后用一个实例来说明该架构的可行性和实用性。

2 基于方面服务和 Web 服务的多层软件架构

2.1 服务器端的组成元素

把面向方面编程的思想引入到 Web 服务中, 将服务中的横切关注点(日志、事务、权限认证等)实现为方面服务, 服务提供者服务器端通过 4 个实体: 方面服务, Web 服务, 方面服务描述文档和编织器来提供服务。

(1)方面服务: 服务提供者将横切关注点如日志、授权、验证等从 Web 服务中分离出来单独实现为方面服务, 它可以像 Web 服务一样开发, 开发者可以使用自己熟悉的编程语言和开发平台来进行开发和部署, 并留有 Web 服务接口供外界使用, 是方面中 Advices 的具体实现。

(2)Web 服务: 不含横切关注点的 Web 服务, 提供者使用

基金项目: 河北省自然科学基金资助项目“面向侧面的分布式软件体系结构研究”(F2006000647)

作者简介: 孟凡新(1983—), 女, 硕士研究生, 主研方向: 软件工程, Web 服务; 张京军, 教授; 刘光远, 讲师

收稿日期: 2009-08-09 **E-mail:** fanxinmeng83@163.com

自己的编程语言、构件和开发平台开发和部署它们，并用 WSDL 描述其接口和调用规范，然后在 UDDI 中心注册和发布以供 SOA 应用开发者使用。

(3)方面服务描述文档：它是一个基于 XML 的文档，包括 AOP 中的 Pointcut 和 Advice，规范描述了触发方面服务的 Pointcut，Advice 类型即 Web 服务和方面服务的执行顺序类型，如 before, after, around(replace)和 Advice 内容，即方面服务的地址信息 URI。

(4)编织器：在系统执行时通过它截获并解析 SOAP 请求信息，根据方面服务描述文档调用 Web 服务和方面服务构建最终系统。

方面服务与 Web 服务的开发和部署方式一样，服务提供者可以使用自己的编程语言和开发平台开发和部署它们，不同之处在于 Web 服务用 WSDL 描述其接口和调用规范并在 UDDI 中心注册和发布，而方面服务留有 Web 服务接口供外界使用，但未在 UDDI 中心注册和发布，与之相对应的方面服务描述文档描述了 Web 服务和方面服务的编织方式，它类似于 AOP 中的“Aspect”，而编织器在应用运行时编织 Web 服务和方面服务构建最终的实际系统。在 Web 服务的实际调用中，Web 客户端和 Web 服务器端都应该包含一个 SOAP 消息监听器，它专门负责 SOAP 消息的接收与发送。所谓编织器，它是传统 SOAP 监听器的扩展，既能处理 SOAP 消息，也能处理 AOP 中的概念(Pointcut 和 Advice)。

2.2 基于方面服务和Web服务的多层软件架构

引入方面服务后，结合 SOA 和 J2EE 的多层软件架构，本文提出的软件架构如图 1 所示。

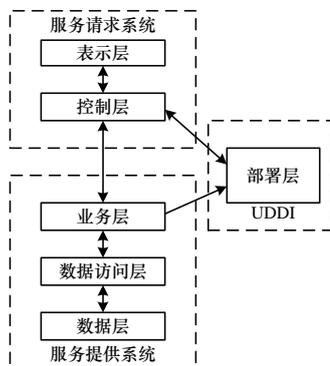


图 1 基于方面服务和 Web 服务的多层软件架构

该架构由 6 层组成，分别是表示层、控制层、业务层、数据访问层、数据层和部署层。

(1)表示层：客户端的用户界面，负责从用户方接收命令、请求、数据，然后传递给控制层，最后将处理结果呈现出来，如收集信息和检查信息合法性的 Web Form 或者 Windows 窗体。

(2)控制层：对请求信息进行处理，负责转发请求到注册中心或远程调用 Web 服务。控制层主要采用 JSP 和 Servlet 技术。

(3)业务层：提供系统的业务逻辑和控制，调用各种已定义的服务如 Web 服务、方面服务来组合构造各种应用，或将应用转换成 Web 服务的形式在部署层注册和发布供系统开发者使用，开发者无须下载服务即可使用且不用知道实现细节。

(4)数据访问层：一个针对具体应用系统的专属层，它为业务层提供与数据源交互的最小操作方式，是业务层需要的数据访问接口，使得业务层只处理业务逻辑而不必关心底层

的数据模式。

(5)数据层：包含 Web 客户端应用程序需要访问的数据，提供了数据信息和数据逻辑以及所有与数据有关的安全性、完整性、数据的一致性、并发操作等。数据层一般采用关系型数据库存储应用数据，如 SQL Server, Oracle, DB2 等。

(6)部署层：用 XML 与 WSDL 为 Web 服务指定标准的接口、行为和交互，在 WSDL 文档中定义了编写使用服务的客户端程序所需的全部内容，如服务地址、输入参数和输出参数等，它包括 Web 服务的注册、发布和发现。

表现层和控制层位于服务请求系统内，业务层、数据访问层、数据层位于服务提供系统内，部署层为服务注册中心。与通常的多层架构不同，本架构多了一个部署层，同时业务层除了包含 Web 服务外，还包含方面服务。与一般的服务调用过程不同，引入方面服务后的整个服务处理过程如图 2 所示。

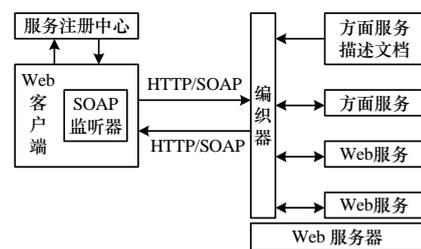


图 2 引入方面服务后的 Web 服务调用过程

在分布式系统开发中，系统开发者首先利用 UDDI 浏览器浏览 UDDI 注册服务器，查询到满足需求的服务后，将相应的服务描述文件(WSDL 文件)下载到本地服务器上，利用 Web 服务 Builder 组件生成客户端代理程序(Service Proxy)，通过这个代理程序在运行时就可以直接绑定到服务的具体实现上。

运行时，首先由请求方的客户端应用程序发出服务调用请求，由客户端代理程序将该请求转化成符合 Web 服务调用所要求的格式。然后由 SOAP 消息监听器将其以 SOAP 请求的形式通过 HTTP 传给服务提供者。服务提供者的编织器接收并解析收到的 SOAP 请求信息，根据方面服务描述文档检查它是否存在触发方面服务的 Joinpoint，一旦满足条件，将根据文档中规定的 Advice 类型和方面服务的地址信息 URI 调用 Web 服务和方面服务，并生成相应的 SOAP 响应消息，通过 HTTP 返回给客户端。服务请求方的 SOAP 监听器解析响应消息并由代理程序解析出处理结果返回给请求方的客户端应用程序。

3 实例分析

用一个实例验证上文提出的软件架构的可行性和实用性。用户管理功能在很多应用系统中都需要用到，如 BBS、图书管理系统等，开发者可以通过集成第三方的用户管理服务来实现系统。用户管理服务包括管理员登录、添加用户、删除用户、查询用户模块，另外需要在各个操作后都有日志记录。这里将日志记录功能从用户管理服务中分离出来单独实现为方面服务，并留有 Web 服务接口供外部调用，日志描述文档包括触发日志服务的 Pointcut(Login(), Add(), Delete(), Query()), Advice 类型 after 和日志服务的 URI，在系统运行时通过编织器根据日志描述文档组合这 2 个服务构成满足需求的应用。基于日志服务和用户管理服务的系统架构如图 3 所示。

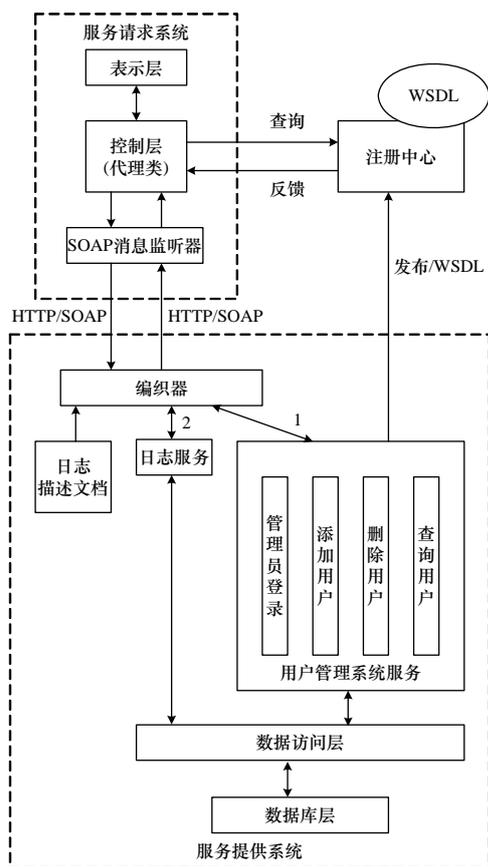


图3 基于日志服务和用户管理服务的系统架构

服务提供者把所提供的用户管理服务部署到Web服务器上,并将它的接口和需要的数据类型及结构用WSDL进行描述,生成相应的WSDL文件并在UDDI注册中心进行发布和注册。应用开发者(BBS开发者,图书馆管理系统开发者)根据需求查询注册表,注册中心搜索到满足请求者要求的Web服务后,将基于SOAP将该服务的WSDL反馈给请求者。请求者利用Web服务Builder组件生成客户端代理类并在程序中添加Web引用将其集成到自己的应用中。

在系统运行时,用户通过表示层提交数据,如管理员想添加一个用户,服务请求系统的表示层接收用户的数据并在控制层通过Servlet等方式过滤数据,然后创建用户管理服务的代理类的新实例并调用代理类上的方法Add(),将该请求转化成符合Web服务调用所要求的格式后,SOAP消息监听器将该方法参数化为SOAP消息并以HTTP传输协议通过网络发送给服务提供系统。服务提供系统的Web服务器处理完HTTP消息,把得到的SOAP消息转发给编译器,然后编译器对SOAP消息进行解析,根据日志描述文档检查到满足触发日志服务的Joinpoint-Add(),将根据日志描述文档中的Advice类型-after和日志服务的URI。先调用Web服务器上的用户管理服务:通过方法Add()、数据访问层和数据库实现对用户信息的添加,然后调用Web服务器上的日志服务:通过方法log()、数据访问层和数据库实现日志功能,最后生成SOAP响应消息并经Web服务器通过HTTP返回给服务请求系统。服务请求系统的SOAP监听器,代理程序解析出处理结果并返回给表示层显示给管理员。

这样将用户管理服务中的日志功能和用户管理功能分开

并单独实现,当系统运行时在SOAP消息层通过编织器组合它们实现系统功能,并无真正的代码编织。某个应用对日志功能的改变,如在操作之前进行日志记录,服务提供者只需重新编写和部署日志功能并相应地修改日志描述文档即可,而不会对用户管理服务产生影响,这样无需修改用户管理服务便可满足应用的需求改变,提高了服务的适应性和重用性。同时系统结构采用多层架构和面向服务软件体系结构,提高了软件的复用程度,便于实现软件的并行开发,减少了开发的时间和复杂度,使应用系统结构更易理解,具有良好的可扩展性和可维护性。

4 结束语

分布式多层结构模型因为有着可伸缩性、可配置性、灵活可靠性以及跨平台和体现软件集成的思想而成为广泛使用的应用模型,SOA及Web服务也是当前研究的热点。本文将面向方面编程的思想引入到Web服务中,将其横切关注点分离出来单独实现为方面服务,并结合SOA和J2EE多层架构提出一个基于方面服务和Web服务的多层软件架构,在系统运行时通过一个编织器实现Web服务和方面服务的组合来实现系统功能。由于在SOAP消息层实现编织,Web服务和方面服务并无真正编织,它们的开发和部署相互独立,在提高Web服务适应性和重用性的同时,也提高了由其构建系统的可扩展性和可维护性。最后用一个实例来说明该架构的可行性和实用性。

下一步的工作包括:(1)编织器的实现。在SOAP消息监听器的基础上,扩展其功能,使其能处理AOP概念和SOAP消息以实现编织功能。(2)利用该架构开发一些大型系统,更好地体现该架构的优越性。

参考文献

- [1] 左银龙, 黄志球, 高鹏, 等. 分布式多层应用系统的设计与实现[J]. 计算机工程, 2001, 27(3): 176-178.
- [2] Kreger H. Web Services Conceptual Architecture[EB/OL]. [2009-03-01]. <http://www-128.ibm.com/developerworks/cn/webservices/ws-wsca/part1>.
- [3] Pallos M S. Service-oriented Architecture: A Prime[EB/OL]. (2001-12-05). <http://www.eaijournal.com/PDF/SOAPallos.pdf>.
- [4] 郑晓东, 王志坚, 周晓峰, 等. 一种基于Web Services的分布式计算模型研究及其实现[J]. 计算机工程与应用, 2004, 40(1): 144-148.
- [5] 董文炜. 基于SOA的分布式Web应用系统的研究[D]. 西安: 西北工业大学计算机学院, 2007.
- [6] Kiczales G, Lamping J, Mendhekar A, et al. Aspect-oriented Programming[C]//Proc. of the 11th European Conference on Object-oriented Programming. Jyväskylä, Finland: [s. n.], 1997: 220-242.
- [7] 朱俭. 面向方面编程(AOP)介绍[J]. 计算机工程, 2004, 30(12): 170-172.
- [8] 肖刚, 陆佳炜, 张元鸣, 等. 基于Web Service的网络构件研究及应用[J]. 计算机工程, 2007, 33(23): 102-104.
- [9] 刘岩, 毛迪林, 杨疆, 等. 基于AOP的Web Services管理架构研究[J]. 计算机应用与软件, 2006, 23(11): 70-72.

编辑 顾逸斐

