

一个 Web 服务管理框架设计方案

徐少平¹, 徐少文², 黄美玲¹

(1. 南昌大学 信息工程学院 计算机科学与技术系, 江西 南昌 330000;
2. 九江学院 计算机系, 江西 九江 332000)

摘要: 简单地说, Web 服务(Web Services)就是自描述模块化的业务应用程序, 它可以通过可编程接口经由 Internet 将业务逻辑发布为服务, 并通过 XML, SOAP, WSDL, UDDI 和 ebXML 等标准协议来查找、订阅和调用这些服务。文中针对日益增长的 Web 服务管理的需求以及存在的问题, 对 Web 服务管理(Web Service Management)、Web 服务管理原则和管理模式等做了深入的研究探讨, 提出了一个 Web 服务管理框架设计方案, 同时阐明了框架的内部组成模块以及各部分之间的相互关系。

关键词: Web 服务; 面向服务的体系架构; Web 服务管理框架

中图分类号: TP311

文献标识码: A

文章编号: 1005-3751(2006)04-0190-03

A Web Services Management Framework Design Scheme

XU Shao-ping¹, XU Shao-wen², HUANG Mei-ling¹

(1. Dept. of Computer Sci. and Tech., Faculty of Info. Eng., Nanchang University, Nanchang 330000, China;
2. Dept. of Computer, Jiujiang Coll., Jiujiang 332000, China)

Abstract: Service-oriented architecture model can utilize standard technology (such as XML) promote and serve to transmit through Internet. To put it briefly, Web service is the module business application program that is described by oneself. They may release business logic as serving via Internet through the programmable interface, and discover, subscribe and invoke these services through Internet agreement. According to requirements of the analysis of managing Web services, this paper presents the framework of Web services management, and expounds the function of the module and relation each other at the same time.

Key words: Web services; SOA; Web services management

0 引言

传统电子商务技术的运作方式在数据格式、可扩展性、事务处理、安全性等方面存在缺陷, 不能很好适应现代商务的需要。随着计算机网络、数据库、软件开发等技术日趋成熟, 这为构建新型的电子商务创造了条件。Web 服务^[1]正是在这种背景下提出的一项新技术, 其基本目标是使软件开发者能跨平台无缝集成某类应用或服务。在 Web 服务技术蓬勃发展的同时, 对于它的管理的需求也日益增长, 由于 Web 服务本身的松散性, 管理上的松懈往往使得提供的服务的质量无法得到保证, 同时对于服务的安全性也是一个大问题。根据对 Web 服务管理需求的分析, 文中提出了一个 Web 服务管理框架设计方案, 同时阐明了框架的内部组成模块以及各部分的相互关系。

1 Web 服务管理

Web 服务采用基于开放的 Internet 标准和技术, 通过

使用 XML, SOAP, WSDL, UDDI 和 ebXML 等标准协议^[1]封装应用程序并将其发布为服务。Web 服务建立于面向服务的体系结构(Service-Oriented Architecture, SOA)^[2]基础之上。SOA 是最新的分布式计算技术, 可以将来自不同系统的应用程序软件组件发布为服务。

Web 服务管理涉及到 Web 服务的各个方面, 包括网络、系统、性能、安全、配置和资源管理, 它必须确保被管理对象的实时信息与先进的服务工具的结合, 有效地防止或快速地修复系统出现的各种问题。结构化信息标准促进组织(OASIS)于 2005 年 3 月 9 日批准了 Web 服务分布式管理(Web Service Distributed Management, WSDM)为 OASIS 标准, 这是 OASIS 最高级别的标准。WSDM 允许开发者利用 Web 服务构建管理应用, 能使多个管理者通过单一界面访问资源。目前已有美国 BEA 系统、美国甲骨文、美国 Sun Microsystems、美国 WebMethods 等多家企业表示支持该标准。

实际上, WSDM 是一个介于管理者和用于访问资源的不同协议之间的整合层。它由两个规范组成: Web 服务管理再发布和 Web 资源服务管理。Web 服务的管理定义如何把 Web 服务作为资源管理; 如何描述和访问这个

收稿日期: 2005-07-23

作者简介: 徐少平(1976-), 男, 江西九江人, 助教, 硕士, 研究方向为计算机图形学、计算机网络。

可管理性。Web 资源服务的管理还提供了能使可管理 Web 服务应用跨不同企业或机构互操作的机制和方法; Web 服务管理再发布定义如何把资源的可管理性接口作为 Web 服务表示和访问,还定义资源的身份、量度、配置和关系等核心功能集。

WSDM 从管理上可以分为二种管理方式:

(1)基本服务管理。从本质上讲是一种反应式的服务,以不同专业领域划分的帮助台^[2,3]响应服务是此类服务的典型代表。这种在服务问题出现后才来解决的被动响应方式缺乏对不同领域相互依赖性的洞察能力,所以它的服务水平一般化且运作效率低下,这种低效性还反映在基本服务指标的按需收集信息方面,致使按照实际业务价值划分优先级的设想无法有效实现;

(2)集成服务管理。反映了帮助台与策略服务台的更多集成,并通过与运行中心的配合使工作流程更加有效。在这一阶段,与业务相关的 SLA^[3]的定义更加明确,服务目标也更加清晰。这种模式提供了基于角色的服务和信息的访问,并且它们所覆盖的专业领域范围也更加广泛。

2 Web 服务管理原则

基于服务的 Web 应用在当前具体应用过程中还存在以下问题:

(1)Web 服务用 XML^[4]来描述,并且使用标准的互操作协议和传输方式来进行访问。一般来说,一个基于 Web 服务的应用必须能够在任何不同的环境下(系统、语言、平台以及企业)使用服务,然而对所有的 Web 服务都规定使用特定的管理技术的做法是不实际的。例如 JMX 在用 Java 语言来实现的 Web 服务中运行的很好,但是在 VC++ 或者微软的 .NET 平台中却无法运行。

(2)基于 Web 服务的应用比传统的应用需要穿越更多的网络边界,所以要求服务的管理也要穿越网络组织边界,而目前的大多数系统都是局限于使用某一公司的同一技术。

(3)Web 服务拥有自然的松散耦合,那意味着应当可以动态地发现服务,在运行的时候使用 Web 服务参数。目前的 Web 服务管理仍然存在在开发的时候静态地绑定服务,影响了系统的可移植性。

解决上述问题的方法是以“管理-代理模型”为基础,在 WSDL^[5,6]中定义管理接口,将管理应用作为一个 Web 服务,用 WSDL 描述可管理 Web 服务并且使用注册中心和 WSDL 来发现 Web 服务。因此管理 Web 服务的时候,必须遵守以下一些特定的原则:

①分离管理接口。Web 服务是一种通过开放的标准的发现和调用机制,它基于一个可访问的接口。Web 服务的发现和绑定过程是由接口来驱动的。在 WSDL^[7]文件中接口被描述为端口类型,当一个组织在 UDDI 注册中心查询一个 Web 服务的时候,它的目标就是找到在 WS-

DL 文件中描述的接口,所以管理操作应当通过分离的接口描述和发布 Web 服务接口来暴露给用户。

②通过运行时框架收集数据。Web 服务通过 SOAP^[8]协议来调用和发送数据,SOAP 协议现在已经成为事实上的标准传输协议。SOAP 消息处理器和相关的 Web 服务基础结构形成了一个控制端,它允许自动地采集一定级别的管理信息。

图 1 说明了 Web 服务基础结构(Web Services Infrastructure)如何实现自动地采集管理信息,在 SOAP 处理器(SOAP PROCESSOR)初始化的时候发现并实例化一个 MBeanServer,它可以为自己创建或者搜索一个已经存在的 Web 服务。MBeanServer 在面向代理的管理模型中扮演着一个重要的角色。

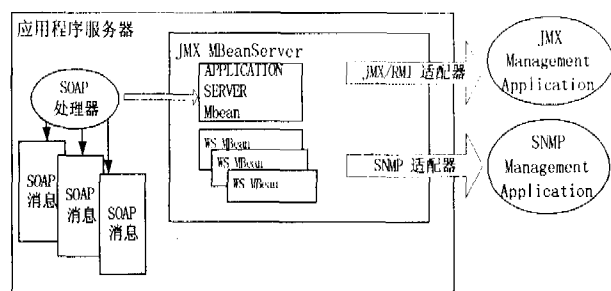


图 1 Web 服务的管理基础结构

③使用事件收集器。对一个管理系统而言,Web 服务需要为发生的一些重要事件发送信号:这包括灾难事件、配置数据的改变、操作的调用以及一个商务的特殊生命周期的事件。一个方便的实现方法就是创建一个 Web 服务的事件收集器。图 2 说明了事件收集器的使用。

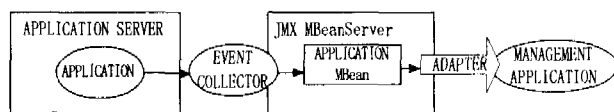


图 2 管理的中介-事件收集器

3 Web 服务管理框架设计方案

Web 服务实际上是一些软件组件,用来解决专门的业务问题。这些服务可以驻留在不同的系统中,并且可以使用不同的技术来实现,而它们却使用标准的 Web 协议(如 XML 和 HTTP)来打包和传输数据。如图 3 所示,根据 Web 服务管理的分析,文中提出了一种 Web 服务管理框架设计方案^[9,10],下面具体阐述设计方案中的各个组成模块。

3.1 Web 服务管理角色

Web 服务管理框架必需管理以下 4 个角色或者说系统中有 4 个重要的参与者需要被管理^[11,12]。

(1)Web 服务提供者(WSP)。这是一个设计和开发 Web 服务的实体。

(2)Web 服务基础设施提供者(WSIP)。该实体提供服务的基础以部署 Web 服务。

(3)Web 服务代理(WSB)。该实体存储在 Web 服务

注册中心注册的 WSIP 或者 WSP 的相关信息。

(4) Web 服务客户 (WSC)。实际的 Web 服务使用者, 比如终端用户或者一个应用程序。

3.2 Web 服务支持模块

Web 服务支持模块^[12]负责根据技术和业务功能来定义系统的行为, 主要包括订阅引擎、契约管理、度量和收费、生命周期管理、审核器、事件管理, 它们的各自模块功能如下:

(1) 订阅引擎。订阅引擎完成服务的订阅, 以及为制定服务的订阅策略。

(2) 契约管理。契约是用来连接 Web 服务客户, 它通过一些条款 (比如收费和服务级别) 来规范 Web 服务客户和服务的提供者。当一个 Web 服务被调用、认证和授权时, 它首先要匹配对应的契约。对于那些非法的契约, Web 服务将拒绝提供服务。

(3) 度量和收费。客户在订阅了一个 Web 服务后根据双方达成的契约来访问 Web 服务。契约管理和度量的应用程序可以看作是一个计费应用的输入数据。

(4) 生命周期管理。管理某个 Web 服务生命周期的各个阶段。这些阶段包括: 开发、部署、注册、测试等。

(5) 审核器。审核器是用来监视和记录其他组件的。它最基本的功能就是根据 Web 服务的上下文来审核 Web 服务事件的顺序, 通过可靠性检查和维护和安全相关的记录来完成这个功能。

(6) 事件管理。在 Web 服务生命周期中定义了许多事件。这些事件可以从技术角度来分组, 也可以从业务领域来分。事件管理器负责维护每一个 Web 服务基本的事件信息, 并且可以配置不同的规则。

3.3 安全通信模块

安全管理器主要提供安全认证、隐私、信任、完整性、机密性、安全通信管道, 联合、委托及跨 Web 服务的审核。

安全机制实现以下的目标:

(1) 可以通过一组声明信息来校验 Web 服务, 如果不能拥有所需的校验信息 (name, key, permission, capability and so on), 该 Web 服务将会被忽略或拒绝服务。这些相关的校验称之为策略 (Policy)。

(2) 服务请求者可以发送消息来确认需求声明以一种和安全相关联的安全令牌信息。该信息包括了特定的行为和用来确认行为的信息声明。

(3) 当一个请求者没有需求声明时, 请求者可以试图和其他的 Web 服务订立契约来获得必要的声明。这些其他的 Web 服务称之为安全令牌服务。

3.4 其他辅助模块

其分辅助模块包括以下几种:

(1) 监测。Web 服务监测是一个十分重要的管理功能。Web 服务的管理应当能够监视所部署的所有 Web 服务的事件。它将帮助管理人员分析业务趋势以及管理 Web 服务。

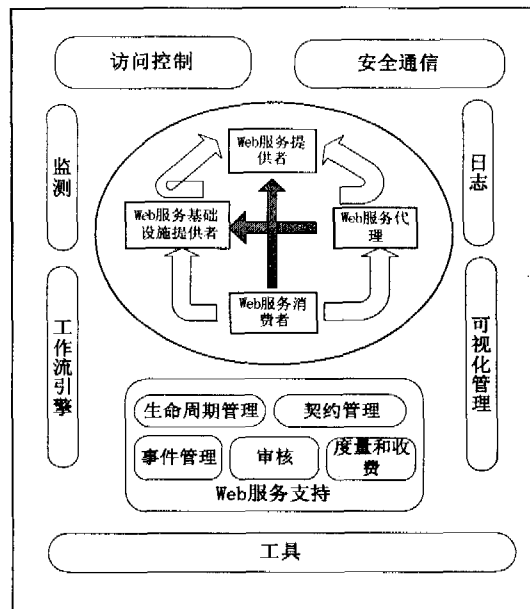


图 3 Web 服务管理框架

(2) 工具。主要用来支持 Web 服务提供者开发 Web 服务; Web 服务基础设施提供者部署 Web 服务; Web 服务代理提供 Web 服务信息; Web 服务客户调用 Web 服务。

(3) 日志。Web 服务的请求、响应、和会话相关的客户信息, 不同的事件等等都要被记录下来, 这些日志用于记录一些重要的操作, 比如服务支持、监测、性能分析等。

(4) 服务的可视化。可视化的应用组件, 能够创建和管理 Web 服务的虚终端, 这些虚终端动态地关联现实的 Web 服务, 以便提供负载平衡, 并管理多个版本的 Web 服务。

(5) 工作流引擎。工作流现今在商务领域是一个很热门的概念, 它同样应用于 Web 服务当中。Web 服务是软件组件, 它的开发是用来解决业务问题的。工作流的意义在于使得设计 Web 服务的业务功能变得更加形象化, 最终解决一个大型的商务问题。工作流引擎创建、测试、管理 Web 服务的逻辑流, 也包括动态的发现、安全性以及事务方面的工作流。

上述设计框架综合了 Web 服务管理的核心模块和辅助模块, 从逻辑上完整地解决了对 Web 服务进行管理各种问题。当然随着 Web 服务的发展, 管理的内容和形式也会发生变化, 该框架也可作相应的扩展以适应新的需求和变化。

4 结束语

文中介绍了基于 SOA 之上的 Web 服务概念, 引出了 Web 服务管理, 并对 Web 服务管理的方法、原则、模式等做了深入研究, 同时根据管理的实际需求提出了一个 Web 服务管理框架, 并对该框架所涉及到的组成模块做了详细介绍, 接下来需要做的工作将是利用具体的软件开发工具实现该 Web 管理架构。

(下转第 195 页)

束字符则继续读

其中 `clearcommerror()` 函数将返回接收缓冲区存放的实际的字符数,它是当前缓冲区内的字符的一个快照,并不一定是所需要接收的实际的完整的字符串,所以必须要根据接收的字符串的起始和结束字符来判断,如果字符串不完整,则需要继续读取,并根据情况来将传输中分离的字符串进行组合。还有一点要注意,由于串口是硬件设备,它的数据接收和发送速度相比之下较慢,所以在循环读取和写串口时要用 `sleep()` 函数来进行一下延迟,再进行串口的操作,避免造成数据丢失。

3.3 串口的监听

串口监听是串行系统实现通信的关键环节,其实现的方法主要有三种:一种是使用定时器控件,修改其时间间隔属性,使其在设定好的时间间隔内循环执行放在定时器中的代码,从而实现了对串口的监听;一种是建立通信事件,设置哪一个串行通信事件被检测,一旦该事件发生则开始相应的执行程序;另一种是建立一个多线程机制,通过主进程来创建一个串口监听的线程来实现实时的数据通信。这里采用了多线程机制,在 Delphi 里产生一个线程的方式是建立线程对象,在线程单元内有一个 `Excute` 子程序区,执行的代码都在其中,在编写代码的时候要在每次执行的时候检查线程的 `Terminate` 属性,当为 `false` 时执行线程代码。因为线程的建立和执行是由应用程序所产生,往往会使用到一些公共的变量,所以必须要采用同步化的机制,主要的同步方式有:

* **Critical Section**:指临界区,该方法只能用于单一的应用程序内,在临界区内的程序代码不能被两个线程同时访问。

* **Mutex**:互斥体,可以用来处理全局变量,后进入的线程必须等到前一个释放后才能执行。

* **Semaphore**:信号灯,它拥有计数功能,可以指定同时操作公共变量的人的个数。

(上接第 192 页)

参考文献:

- [1] 徐少平,孙 骏,徐少文,等. Web 服务核心协议及其实现[J]. 计算机与现代化,2004(5):32-35.
- [2] 胡智文,陈国龙. 新一代分布式计算模型——XML Web Services[J]. 计算机工程,2004,30(18):1-3.
- [3] 柴晓路,梁宇奇. Web Services 技术、架构和应用[M]. 北京:电子工业出版社,2003.
- [4] Holzner S. XML 完全探索[M],陶 阳等译. 北京:中国青年出版社,2001.
- [5] Ogbuji U. Simple Object Access Protocol[EB/OL]. <http://www.w3c.org/TR/SOAP>,2001-07-09.
- [6] Word Wide Web Consortium. Web Services Description Language (WSDL)[EB/OL]. <http://www.w3c.org/TR/wsdl>,2000-09-25.

其关键代码如下:

```
Procedure TReadThread. Execute;
var
  {变量定义}
Begin
  While not Terminated do
    Begin
      Synchronize (ReadPort);
      {线程执行代码区}
    end;
end;
```

其中将读串口的过程定义成一个函数,然后利用 `Synchronize()` 函数实现对公有变量的访问同步。

4 结束语

本系统充分利用了操作系统所提供的强大的 Win API,并且结合实际的工程项目,实现了串口数据的有效通信,同时提高了数据通信设计的灵活性,开发人员可以依据实际情况来实现系统的多方面要求,增强了系统的使用效率。本串口通信技术已经在大型铝厂的出铝过程自动控制系统中得到了很好的应用,也为该系统在今后的产业化过程中打下了良好的基础。

参考文献:

- [1] 李也白,全 臻,刘伟立. 基于 RS-485 的树状异步串行通信系统的研究[J]. 计算机应用研究,1999,16(4):21-22.
- [2] 范逸之,陈立元. Delphi 与 RS-232 串行通信控制[M]. 北京:清华大学出版社,2002.
- [3] 崔建华,郭瑞金. Delphi 串口通信工程开发实例导航[M]. 北京:人民邮电出版社,2003.
- [4] 龚建伟,熊光明. Visual C++ / Turbo C 串口通信编程实践[M]. 北京:电子工业出版社,2004.
- [5] 李现勇. Visual C++ 串口通信技术与工程实践[M]. 北京:人民邮电出版社,2002.
- [7] Brittenham P. Universal Description, Discovery, and Integration (UDDI)[EB/OL]. <http://www.uddi.org>,2001-09.
- [8] A brief introduction to the web services protocol stack: Soap, wsdl and uddi, HP Web Services Platform[EB/OL]. <http://www.hp.com/go/webservices>,2001-08-25.
- [9] Dale J. Advanced Web Services[J]. Open Agent Systems,2002(3):11-20.
- [10] Leymann F. Web Services Flow Language (WSFL 1.0)[EB/OL]. <http://www-4.ibm.com/software/solutions/web-services/pdf/WSFL.pdf>,2001-06.
- [11] W3C 的 Web Services 架构标准[EB/OL]. <http://www.w3.org/TR/2003/WD-ws-arch>,2003-08-08.
- [12] Gottschalk K, Graham S, Kreger H, et al. Introduction to Web Services Architecture[J]. IBM Systems Journal,2002,41(2):10-15.