基于 Axis 的 Web Services 在 SLA 服务管理系统中的应用①

Application of Web Services based on Axis in Service Management System of Service Level Agreements

王宇罡 张骏温 戴 钢 （北京交通大学计算机学院网管中心 100044）

摘要：本论文针对目前流行的基于 Axis 的 Web Services 技术进行了通用性的描述，包括 Axis 的架构、组件、部署描述符等；然后介绍了使用 Axis 的 Web Services 在国家 863 项目 SLA 服务管理系统中的应用，具体包括 Server 端、Client 端 Web Services 的导入、导出以及发布、部署等。

关键词：服务等级协议(SLA) Web Services Axis SOAP

1 引言

Web Services 技术是近年来逐渐兴起的技术。Web Services 的主要目标就是在现有的各种异构平台的基础上构筑一个通用的与平台无关、语言无关的技术层，各种不同平台之上的各应用依靠这个技术层来实施彼此的连接和集成。以本文中的 SLA 服务管理系统和 Jd863 服务管理系统为例，它们是两套架构及功能差别较大的系统，而我们要完成这两个系统间的定时自动传送数据的过程。应用 Web Services 技术，我们就可以很好地在两个系统之间进行通信，完成此功能。

2 Axis 及相关背景介绍

2.1 SOAP 及 Axis 简介

SOAP 是 Simple Object Access Protocol 的缩写，是一种基于 XML 的不依赖传输协议的表示层协议，用来在分散或分布式的应用程序之间方便地以对象的形式交换数据。SOAP 最初由微软和 Userland Software 提出，随着不断地完善和改进，SOAP 很快被业界广泛应用，目前完全发布版本是 1.1。在其发展过程中，W3C XML 标准工作小组积极促成 SOAP 成为一个真正的开放标准。在写作本文之时，SOAP1.2 草案已经发布，1.2 对 1.1 中相对混乱的部分做了改进[3]。SOAP 已广泛作为新一代跨平台、跨语言分布计算 Web Services 的重要部分。

Axis 是 Apache 组织推出的 SOAP 引擎，Axis 项目是 Apache 组织著名的 SOAP 项目的后继项目，目前最新版本是采用 Java 开发的 1.1 版本，C++ 的版本正在开发之中。Axis v1.1 软件包可以从 apche 的网站下载得到。

Axis 不仅仅是一个 SOAP 引擎，它还包括：一个独立运行的 SOAP 服务器；一个 servlet 引擎的插件，这个 servlet 引擎可以是 Tomcat 对 WSDL 的扩展支持；一个将 WSDL 的描述生成 JAVA 类的工具；一些示例代码以及一个监控 TCP/IP 包的工具。

Axis 提供两种将 Java 类发布成 Web Services 的途径——即时快速自动发布和通过配置文件进行发布。本例采用的是通过配置文件进行发布。

2.2 SLA 服务管理系统简介

SLA(Service Level Agreements，服务等级协议)是一个服务提供者与客户间的服务协定，它以量化的性能指标来规定服务需要满足的要求和费用，使用它能

① 项目基金：国家 863 计划课题 课题编号：2005AA147110

更好地确保服务的有效性，减少客户的抱怨，使服务供应商和客户达到双赢的目的。

3　基于 Axis 的 Web Services 在 SLA 服务管理系统中的实际应用

3.1　Axis 架构分析

结合实际应用，我们将对基于 Axis 的 Web Services 的架构进行分析，如图 1 所示。

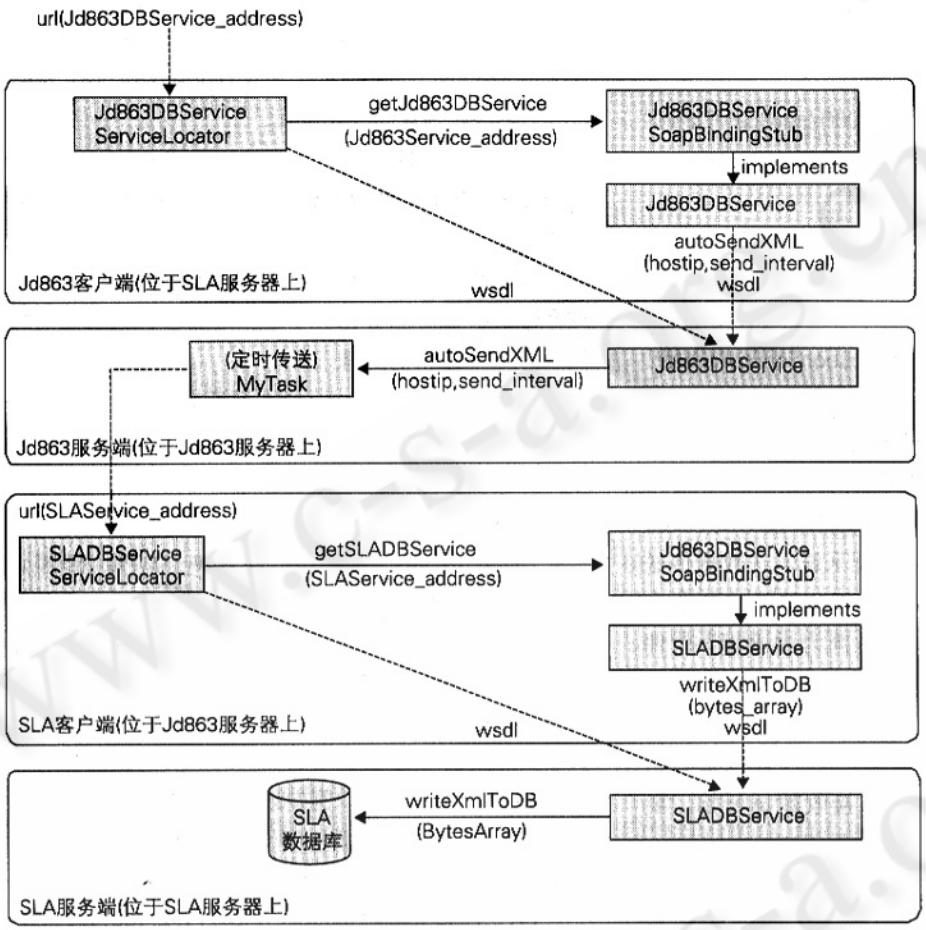

图 1　Axis 实际应用架构图

现将图中的一些术语介绍如下：

SLA 服务端：SLA Web Services 中作为 SLA 服务的发布者；

SLA 客户端：SLA Web Services 中作为 SLA 服务的接收者；

Jd863 服务端：Jd863 Web Services 中作为 Jd863 服务的发布者；

Jd863 客户端：Jd863 Web Services 中作为 Jd863 服务的接收者；

SLA 服务器：SLA 服务管理系统所在的机器；

Jd863 服务器：Jd863 系统所在的机器。

从图 1 中我们可以看出，Axis 客户端的核心类有 ServiceLocator，SoapBindingStub 以及 DBService。

ServiceLocator 负责查找网络上已发布的 Web Services，SoapBindingStub 是 Axis 客户端用来查找已发布的 WebSerive 在客户端的存根（即 DBService）的。

在两台服务器之间进行通信是靠 wsdl 来完成的。wsdl 的全称是 Web Services Definition Language，它描述了 Web Services 对外发布的信息，外界要访问它时所需的信息在 wsdl 文件中都有所说明。如本例中的 http://192. 168. 199. 126: 8080/services/DBService?wsdl 就是一个 wsdl 的链接地址。

除了以上组件，Axis 的服务端下面还有一个 server-config. wsdd，部署了服务端 Web Services 发布时有

SLA 服务管理系统是一个国家 863 软件项目，它的目的就是使 SLA 理念在实际应用中发挥作用。SLA 将服务对象和服务分别按层次结构分类，对不同层次的对象授予不同的服务级别。不同的服务也根据其服务内涵、服务环境以及服务策略的差异，选择提供服务的层次和方式，形成一个灵活、开放、可伸缩、可裁减、可配置的服务体系框架。强调服务等级的差别，可以保证所有对象获得其所要求的最合适的服务。

另外，SLA 服务管理系统是和 Jd863 系统配合使用的。Jd863 系统是一个国家专业软件孵化器，SLA 服务管理系统作为支撑软件为其提供各种服务、配置及管理。

关的信息。例如，wsdd 在传送与接收时的位置信息和相关的类信息，要发布的 Web Services 服务的版本，还有所需的 apache 提供的类信息等。

进行数据传送的核心代码如下：

```
String wsdlUrl =
    " http://192.168.199.126:8080/services/DBService?wsdl";
DBServiceServiceLocator
    webServiceBeanServiceLocator = new DBServiceServiceLocator();
DBService
    webServiceBean = webServiceBeanServiceLocator.
getDBService(new URL(wsdlUrl));
webServiceBean.writeXmlToDB(b);
```

3.2 基于 Axis 的 Web Services 的部署及发布

本实例有两个系统，SLA 服务管理系统以及 Jd863 软件孵化器系统。由 SLA 服务管理系统发送数据传送命令以及时间间隔，由 Jd863 软件孵化器系统接收命令，进行数据传送。数据的传送方向为由 Jd863 软件孵化器系统至 SLA 服务管理系统。

首先，两个系统各有一套数据库管理系统，本例中均为 mysql 数据库。在两个数据库中各有一张名为 sla_accesstrace 的表，用来传送与接收数据所用。要传送的数据已存在 Jd863 系统的数据库中。

接下来，为实现数据传送的功能，我们要在 SLA 服务管理系统和 Jd863 系统两端分别安装 Web Services 的服务端和客户端（也可以两端均安装服务端和客户端的集合体，本例中为分开实现）。

建立 SLA 服务管理系统服务器端 Web Services 步骤如下：

（1）先用 JBuilderX 建一个工程，本例中为 NewSlaService。

（2）添加要发布为 Web Services 的类，本例中为 sla.DBService，接着写好要发布的函数，本例中为 public void DBAccess(String sql,String op_type) 和 public void WriteXmlToDB(byte[]b)。

（3）新建 Web Services，名称为 SLA，导出 wsdl，Location URL 设置为 http://192.168.199.126:8080/services/DBService，Methods 里面选中 DBAccess 和 WriteXmlToDB。SOAP Version 选 1.2。其间要建立一个 Web 应用，本例中为 SLA。

（4）编译本工程，将生成一些类和部署描述符，具体为 DBService.class，sla.generated.*，SLA\WEB-INF\server-config.wsdd，SLA\SOAPMonitorApplet.jar，SLA\AdminInstruction.html，SLA\fingerprint.jsp，SLA\happaxis.jsp，SLA\index.html。

以上 1－4 都是建立服务端 Web Services，下面步骤将把建好的 Web Services 部署到实际的 Web 应用中去。

（5）设置 JAVA_HOME，TOMCAT_HOME，CLASSPATH，CATALINA_HOME，AXIS_HOME，AXIS_LIB，AXIS-CLASSPATH 等环境变量

（6）拷贝 DBService.class 至实际目录，本例中为 d:\SLA\WEB-INF\classes\sla。

（7）拷贝 sla.generated.* 至实际目录，本例中为 d:\SLA\WEB-INF\classes\sla\generated。

（8）拷贝 SLA\WEB-INF\server-config.wsdd 至实际目录，本例中为 d:\SLA\WEB-INF。

（9）拷贝 SLA\SOAPMonitorApplet.jar，SLA\AdminInstruction.html，SLA\fingerprint.jsp，SLA\happaxis.jsp，SLA\index.html。至实际目录，本例中为 d:\SLA。其中 index.html 可能要改名，避免和 Web 应用中的 index.html 文件重名。本例中改名为 Index33.html。

（10）修改 SLA\WEB-INF\web.xml，因为 Web Services 要生成一些 Servlet，web.xml 的内容会有所变化。

（11）在 ie 地址栏里输入 http://192.168.199.126:8080/index33.html，在出现的界面中点击“View”，则出现表示服务端 Web Services 已发布成功的画面。

4 结束语

基于 Axis 的 Web Services 在 SLA 服务管理系统中发挥了很大作用。用户可以通过它，方便地对系统参数进行设定，完成一定的功能。它的部署过程尽管稍有些繁琐，但是部署之后进行修改是很方便的。

参考文献

1 柴晓路、梁宇奇编著，《Web Services 技术、架构和应用》，电子工业出版社，2003。
2 第八届中国 Java 技术及应用交流大会文集，2005。
3 W3C Recommendation, 24 June 2003.