基于 WebSphere 平台的 Web Service 概念实践

范玉珍　顾毓清
（中国科学院软件研究所　中科软件有限公司　北京100080）

The Experience for the Concept of Web Service on WebSphere

FAN Yu-Zhen　GU Yu-Qing
(Institute of software, Chinese Academy of Sciences, Zhongke Software Corporation, Beijing 100080)

Abstract　This paper investigates and summarizes the basic concept of Web Service, covers how to set up the fundamental Web Service structure on WebSphere platform with the provided practical example, the last section of this paper discusses the comparison between the Web Service component-based architecture and the traditional component-based architecture.

Keywords　Web service, Advanced component, AgentGroupManagement AC

1. 前言

经济与社会全球化要求企事业突破传统的单位限制，加强交流和资源共享，共组虚拟和动态的企业联盟，Internet 应用就成为企事业实现互联互通的桥梁。

为适应全球化的需求，各种应用系统需要进行整合，传统的应用程序组成方式开始发生变化，新的技术和理念支撑着应用系统由单一技术平台转向分布式技术平台。

Web Service 迎合了经济社会全球化与应用系统分布式这两大发展趋势，其将软件看成是一种 Internet 上的服务单元，借助 XML 技术，实现各种应用程序资源的互联互通、从而提供分布式的（全球性的）信息整合手段和应用系统解决方案，有利于促进软件资源的共享，推动经济与社会的全球化进程。

本文将主要探讨如何在 IBM WebSphere 平台上实现 Web Service 理念。

2. Web Service 概述

2.1　什么是 Web Service

关于 Web Service 的定义有很多种，如下的两种定义较为贴切地表达了其真实的含义：

定义1　Web Service 是基于网络的、分布式的模块化构件，它执行特定的任务，遵守具体的技术规范，这些规范使得 Web Service 能与其他兼容的构件进行互操作。

定义2　所谓 Web Service，它是指由企业发布的完成其特别商务需求的在线应用服务，其他公司或应用软件能够通过 Internet 来访问并使用这项应用服务（UDDI 规范2.0）。

定义1强调了 Web Service 的技术特性（是对象/构件技术在 Internet 中的延伸）；而定义2强调了 Web Service 的功能特性（是在 Internet 上由一个实体发布的可供别的实体使用的在线服务程序单元）。

2.2　Web Service 的主要特点

Web Service 是下一代分布式系统的核心，它具有如下主要特点[1]：

·完好的封装性，Web Service 既然是一种部署在 Web 上的对象，自然具备对象的良好封装性，对于使用者而言，他能且仅能看到该对象提供的功能列表。

·松散耦合与分布式特性，Web Service 的目标是为用户提供一套方法，来整合网上的可用资源（服务构件）成为一种具体的解决方案，各种资源可能分布在不同的地方、不同的平台上，处于一种无中心的分布环境中，整合采取松耦合方式。当然，当一个 Web Service 的实现发生变更的时候，调用者是感觉不到的，对于调用者来说，只要 Web Service 的调用界面不变，Web Service 的实现任何变更对他们来说都是透明的。

·使用协约的规范性，这一特征从对象而来，但相比一般对象其界面规范更加规范化和易于机器理解。首先，Web Service 使用标准的描述语言（基于 XML 的 WSDL）来描述和提供对象接口和功能；其次，由标准描述语言描述的服务接口是能够被发现的（通过 UDDI）；同时，这些服务接口可以用来实现聚合、Web Service 之间的事务和工作流等；此外，Web Service 还使用规范的方法来描述、传输和交换信息，从而保证松散耦合的对象环境下的安全性（比如授权认证、数据完整性（比如签名机制）、消息源认证以及事务的不可否认性等）；最后，Web Service 通过管理协约来保证在所有层次的处理都是可管理的。

·使用标准协议规范，作为 Web Service，其所有公共的协约完全需要使用开放的标准协议进行描述、传输和交换。这些标准协议具有的规范是完全免费的，因此可由任意方进行实现。一般而言，绝大多数规范将最终有 W3C 或 OASIS 作为最终版本的发布方和维护方。

· 高度可集成能力。由于 Web Service 采取简单的、易理解的标准 Web 协议作为构件界面描述和协同描述规范，完全屏蔽了不同软件平台的差异，无论是 CORBA、DCOM 还是 EJB 都可以通过这一种标准的协议进行互操作，实现了在当前环境下最高的可集成性。

2.3　Web Service 的基本构架

Web Service 是独立的、模块化的应用，能够通过网络，特别是 WWW 来描述、发布、定位以及调用。Web Service 的基本构架描述了三个角色（服务提供者、服务请求者、服务代理者）以及三个操作（发布、查找、绑定），具有简单、动态和开放的特征。其基本构架如图1所示[2]。

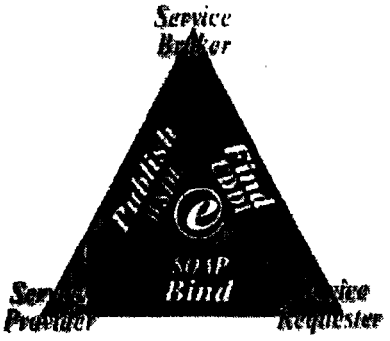

图1 Web Service 基本构架

从图1可以看出，Web Service 包含了如下几个关键的规范：

·Web Service 描述语言(WSDL) WSDL 是用来描述网络(network)服务或终端(endpoint)的一种 XML 语言，它用于定义 Web Services 以及如何调用它们(描述 Web Service 的属性，例如它做什么，它位于哪里和怎样调用它)。WSDL 文档可用于动态发布 Web Services，查找已发布的 Web Services 以及绑定 Web Services。

·通用描述、发现和集成(UDDI) UDDI 是用于服务描述和发现的一组标准规范，提供了在 Web 上描述并发现商业服务的框架。UDDI 通过一个分布式注册表来管理 Web Service 的发现，这个分布式注册表是业务和它们的以公共 XML 格式实现的服务描述的注册表。

·简单对象访问协议(SOAP) SOAP 是用于交换 XML 编码信息的轻量级协议。它有三个主要方面：XML-envelope 为描述信息内容和如何处理内容定义了框架；将程序对象编码成为 XML 对象的规则；执行远程过程调用(RPC)的约定。

3. WebSphere 平台上的 Web Service 构件构架与实践

WebSphere 是 IBM 公司对原有的网络应用中间件进行改进而推出的一种 Web 应用软件平台，WebSphere 对 J2EE、XML、LDAP 和 WAP 等重要的工业界标准均提供支持。利用其中的 WebSphere Business Component(WSBC) Studio 平台，可以完成 Web Service 的构造、部署和运行管理。

Web Service 的基本角色是 Internet 环境下用于创建开放分布式系统的软件构件。WebSphere(WSBC)将 Web Service 构件划分为基本构件和高级构件。

3.1 基本构件和高级构件

基本构件(Base Component，BC)是简单商业对象，使用熟知的 EJB 编程模式，提供可配置的商业功能。基本构件包含可重用和可定制的 EJB，并且支持 Java 类。高级构件(Advanced Component，AC)是由与技术无关的方式定义的黑盒，构件定义提供基于 XML 文档的构件接口，构件实现则提供上述接口的实现。高级构件可由基本构件、Java 编程技术或其他基础结构实现。

3.2 高级构件体系结构

AC 使用者仅通过 AC 接口与 AC 通信，AC 接口由 WSDL 文档描述的 AC 规格说明书来定义。AC 客户向 AC 接口发送 XML 消息请求，AC 接口将 XML 消息响应返回给 AC 客户。在 SOAP 通信协议中，上述 XML 消息由 SOAP 编码，并嵌入在 SOAP 信封中。其体系结构如图2所示[3]。

3.3 高级构件的接口类型

AC 接口由三种类型组成：分别是 AC 方法、AC 事件及 AC 系统管理。它们的特征如下：

AC 方法代表与逻辑一致的操作，例如由 AC 客户调用的商业操作。AC 方法可参与事务，很大程度上代表大粒度的商业操作。

AC 事件发送通告信息，通常 AC 事件较之 AC 方法传载信息量较小，因此用于轻量级的信息通知。AC 事件分为发布和定购两种类型，不可参与事务。

AC 系统管理专用于系统管理操作，例如启动/停止或配置操作。

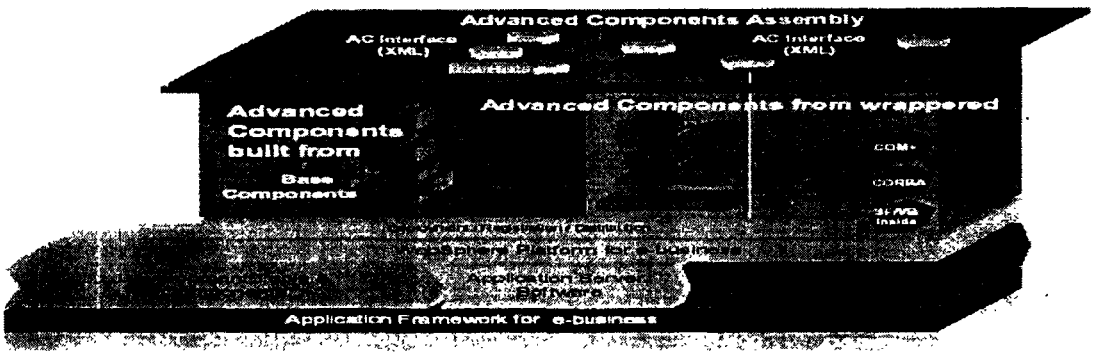

图2 AC 体系结构层次图

3.4 代理商及其成员组管理高级构件的设计、实现与使用

在这里，我们将以 AgentGroupManagement AC 为例，具体讨论如何在 WebSphere 平台上实现 Web Service 及其应用。

3.4.1 AgentGroupManagement AC 简介 企业组织借助 AgentGroupManagement AC，处理和维护该组织中代理商及其成员组的固有信息。代理商信息表现为代理商的基本(basic)和基础(fundamental)信息，例如：姓名、成员编号和电话号码等。成员组信息表现为若干具有相同特征的代理商的集合，例如：组名、成员组编号等。授权的组织管理员，即操作者，使用 AgentGroupManagement AC 管理代理商及其成员组，以及二者之间的所属关系。

3.4.2 AgentGroupManagement AC 适用的商业领域 AgentGroupManagement AC 适用于销售管理，客户关系管理，或者任何需要维护代理商及其成员组基本信息的领域。

3.4.3 AgentGroupManagement AC 功能及用例图 为处理和维护上述领域中代理商及其成员组信息，

AgentGroupManagement AC 使操作者执行以下功能：

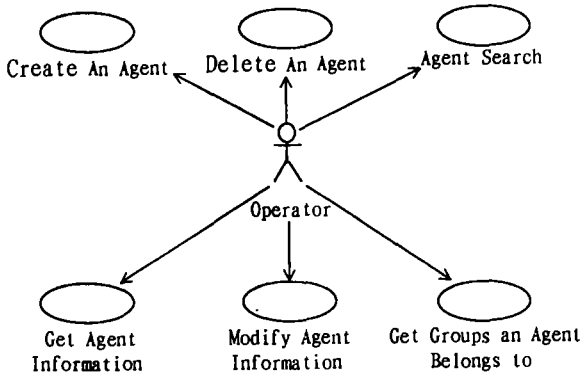

图3 代理商管理用例图

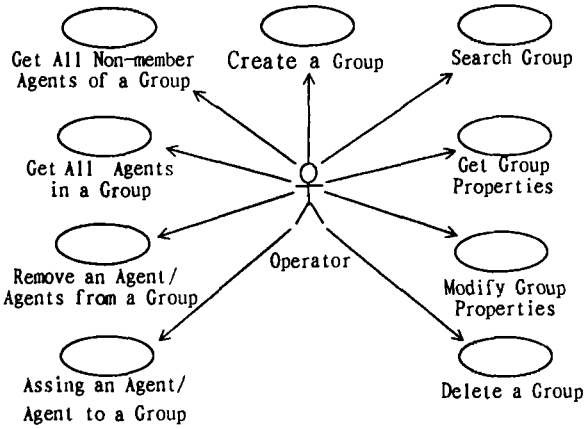

图4 成员组管理用例图

·创建成员或成员组，并加入相关信息；
·返回成员或成员组信息；
·更新成员或成员组信息；
·使用搜索条件查询成员或成员组；
·删除成员或成员组；
·分配某个成员到指定成员组；
·将某个成员移出指定组。

本节的用例图(图3、图4)体现了不同产业和交付渠道中代理商及其成员组管理应用的各项功能。尽管其中大部分用例图来源于呼叫中心，但适用于所有的相关领域。

3.4.4 AgentGroupManagement AC 建模及代码生成

利用 Rational Rose 对 AgentGroupManagement AC 设计建模，所建模型如图5所示。

利用 WSBC 工具集中提供的代码生成器自动生成相应源代码(Java code)和 XML 文档，这种从 UML 模型产生代码的过程称为前向代码生成(forward generation)。此后，利用 WSAD 或其他 IDE 增加代码以完成商业逻辑实现。在前向代码生成完成后，如果模型设计发生变化，修正 UML 模型后重新生成 Java 源代码，这种过程称为再次代码生成(regeneration)。

3.4.5 AgentGroupManagement AC 的描述文档 除 Java 代码外，WSBC 代码生成器为即将发布为 Web Service 的 AgentGroupManagement AC 产生如下 XML 文档：

-Web Service 接口定义，对应于接口 WSDL
-Web Service 服务定义，对应于服务 WSDL
-Web Service 构件实现，对应于 AC 实现 XML
-Web Service 部署描述，对应于 AC 部署描述 XML
-Web Service 管理配置，对应于 AC 系统管理配置 XML
-Web Service 接口语法，对应于 AC 接口模型 Schema

图5 AgentGroupManagement AC 模型结构图

3.4.6 AgentGroupManagement AC 的使用 每一种 AC 接口类型(见前面3.3节)对应一套网络协议，以促进 AC 同其客户端通信。AC 接口同协议的映射关系如表1所示。

表1 AgentGroupManagement AC 接口与协议映射图

接口类型	通信类型	传送机制
AC 方法	Business 功能处理	SOAP over HTTP or SOAP over EJB(IIOP)
AC 事件	Business 通知	SOAP over JMS(Java Message Service)
AC 系统管理	系统管理	SOAP over JMX (Java Management Extensions)

限于篇幅，下面以通过 HTTP 使用 SOAP 调用 AC 方法例示客户端对 AgentGroupManagement AC 实例的使用。

AC 供应者需要提供给 AC 客户端 WSDL 接口文档和 WSDL 服务文档，以使其获得必要的信息来访问 AgentGroupManagement AC。

WSDL 接口文档定义 AC，以下为更新代理商档案方法的消息表示：

```
<message name="UpdateAgentProfileRequest">
    <part name="anAgentBaseInfo" type="imns:AgentBaseInfo"/>
    <part name="aRoleID" type="xsd:string"/>
</message>
<message name="UpdateAgentProfileResponse">
    <part name="agentBaseInfo" type="imns:AgentBaseInfo"/>
</message>
<portType name="AgentGroupManagementInterface">
    <operation name="updateAgentProfile">
        <input message="tns:UpdateAgentProfileRequest"/>
        <output message="tns:UpdateAgentProfileResponse"/>
        <fault message="tns:FaultMessage" name="UpdateAgentProfile_"/>
    </operation>
    ……
</portType>
```

WSDL服务文档定义AC实例，以下定义了AgentGroup Management AC 的实例 AgentGroupManagementACInstance：

```
〈service documentation = "AgentGroupManagementAC" name = "AgentGroupManagementACInstance"〉
    〈port binding = "def: AgentGroupManagementInterfaceSOAPHTTPBinding" name = "AgentGroupManagementInterfaceSOAPHTTPPort"〉
        〈soap: address location = "http: // localhost: 9080/soap/servlet/rpcrouter/"/〉
    〈/port〉
    〈port binding = "def: AgentGroupManagementInterfaceSOAPEJBBinding" name = "AgentGroupManagementInterfaceSOAPEJBPort"〉
        〈soap:address
          location = "iiop: // localhost: 900? EJBHomeName = SOAPEJBRouter&amp; ContextFactory = com. ibm. webspher. enaming. WsnInitialContextFactory"/〉
    〈/port〉
〈/service〉
```

AC 客户端可以以两种方式访问 AC，分别是：使用 AC 服务和使用 SOAP 调用。以前者为例：使用 Java 客户机代理，该客户机代理为 WSDL 文档中描述的服务提供自然的 Java 接口，并将精密地镜像原始 bean 的接口。代理封装 SOAP 客户机编程 API 以及有关编写、发送、接收和分析 SOAP XML 文档的详细信息。

```
声明：AgentGroupManagementACServiceProxy AgentGroupManagementACSer-viceid = New AgentGroupManagementACService Proxy;
调用：AgentBaseInfo theAgentBaseInfo = AgentGroupManagementACServiceid. updateAgentProfile (anAgentBaseInfo,aRoleID);
```

以上调用范例使用Java客户机代理为：AgentGroupManagementACServiceProxy，适用于 Java Servlet 和 Applet 这两种编程模式。

4. Web Service 构件构架与传统构件构架的比较

与传统的构件构架（这里主要是指 COM/DCOM、JAVABEAN/EJB、CORBA 等组件构架）相比，Web Service 构件构架具有如下的优势：

1）Web Service 构件构架是一个更具分布式特性的构架，它不仅具有传统构件构架所具有的跨语言、跨平台等能力，而且具有跨传统构件标准的能力，能够有效地整合各种传统的构件模型，实现不同类型构件的相互调用（譬如，可以将 EJB 构件和 COM 构件包装成 Web Service 构件，从而实现两种异种构件的相互调用）。

2）Web Service 将构件看成是一种 Internet 上的服务单元，并把 Internet 当成是一个构件库，更利于构件的广泛重用；而传统的构件构架将构件看成仅是本地的一个可调用到的功能单元，需要专门的构件库。

3）Web Service 构件构架是对传统构件的改进，可以直接在 Internet 上作为服务单元发布，减少了传统软构件需要通过存储介质发送、安装、注册等环节，因而更加廉价、更加方便。

4）Web Service 构件构架更利于实现跨企业、跨区域的程序和数据交互，Web Service 构件通过将服务接口"裸露"的方式，使得外部的应用程序可以轻易地跨越企业、区域间的防火墙等安全屏障，实现商务逻辑的便利互访，而传统的构件架构要实现防火墙跨越，开发难度会非常大，且程序很难维护。

当然，我们可以看出，Web Service 在通过 Web 进行互操作或远程调用的时候是目前最有效的方式，而对于某些情况，譬如单机应用和局域网内的同构应用，Web Service 就没有什么优势了。

结束语 本文介绍了在 WebSphere 上的一个 Web Service 构件实现方案，该方案已在某地银行系统中得到了实施。此外，还需要说明的是：目前，Web Service 虽然浪潮迭起，各大软件开发商也争相提出自己的解决方案（微软提出的方案是.Net，并把 Web Service 称之为是软件开发技术的一次革命，Sun 拿出的方案叫作 Sun One），但是从总体上来讲，目前 Web Service 商业级系统所需的大多数技术和标准才刚开始出现，还没有统一的、最后的定论。不过，Web Service 作为 Internet 时代的一种计算模式，其美好的前景是不容质疑的。

参 考 文 献

1 柴晓路. 架构 Web services 系列. IBM developerWorks 中国网站，2001. 8
2 Li Jin. Web Services Overview. IBM developerWorks 中国网站，2002. 1
3 IBM 公司. IBM WebSphere Business Components Architecture Overview. IBM 公司白皮书，2000. 12
4 包路跃. Web Services 概述. 天极网（www. yesky. com）Visual Studio. net 专栏，2002. 4
5 蒋松，白利强. Web Service——下一代的 WWW. 计算机世界报，2001. 10
6 微软公司. Microsoft . NET 让新一代因特网变成现实. 微软公司白皮书，2000. 6

（上接第112页）

用 HTTP 作为请求/响应消息传输协议。SOAP 被设计为与 XML Schema 规范密切配合，并支持在 Internet 的任何地方运行的 COM、CORBA、Perl、Tcl、和 Java、C、Python 或 PHP 等程序间的互操作性。当然 SOAP 为了体现简单性和扩展性，也牺牲了一些技术，如它不支持对象引用等。

SOAP 的一个主要目标是使存在的应用能被更广泛的用户所使用，为了实现这个目的，没有任何 SOAP API，SOAP 是假设你将使用尽可能多的存在的技术。几个主要的 CORBA 厂商已经在他们的 ORB 产品中支持 SOAP 协议，微软也在 COM 中支持 SOAP，DevelopMentor 已经开发了参考实现。同时. NET 体系结构的客户端也是通过 SOAP 协议访问 asp 提供的 web service，从而执行应用程序的。相信在不久的将来 SOAP 必将成为互联网中一颗璀璨的明星。

参 考 文 献

1 Extensible Markup Language (XML). Available at http: // www. w3. org/XML, April 1997
2 XML Developer Center. Available at http: // msdn. microsoft. com/xml, November, 2000
3 The Apache XML Project. Available at http: // xml. apache. org/soap, 1999
4 Simple Object Access Protocol (SOAP) 1. 1. Available at http: // www. w3. org/TR/SOAP, May 2000
5 Object Management Group. CORBA/SOAP Interworking Request For Proposal. February 5, 2001
6 Microsoft . Net Enterprise Servers. Available at http: // www. microsoft. com/net/, 2000