# 基于 Web Services 应用集成技术的研究及实现*)

毛春丽[1] 贾 焰[2] 周 斌[2]

(湖南商学院艺术设计系 长沙 410205)[1] (国防科学技术大学计算机学院 长沙 410073)[2]

**摘 要** 将现存的中间件技术如 CORBA 与新兴的 Web 服务实现互操作,可实现新的、增值的服务。文中阐述了如何将 CORBA 对象发布为 Web 服务以及 Web 客户如何能在不改变原有程序模式的情况下访问 CORBA 服务。

**关键词** Web Services,CORBA,CORBA-SOAP 网关

## Research and Implementation of Application and Integration Technology Based on Web Services

MAO Chun-Li[1] JIA Yan[2] ZHOU Bin[2]

(Department of Art Design, Hunan Business College, Changsha 410205)[1]

(School of Computer Science, National University of Defense Technology, Changsha 410073)[2]

**Abstract** The interoperation of existing middleware technologies such as CORBA and the emerging Web services is necessary to implement new, valued-added services. This paper illustrates how to expose existing CORBA-implemented systems as Web Services and how to enable Web client access to services built using CORBA without changing the original programming mode.

**Keywords** Web services, CORBA, CORBA to SOAP GateWay

中间件互操作就是在建立在不同中间件上的应用之间交换信息和使用信息。目前主流的中间件平台有 OMG 的 CORBA 平台、SUN 的 J2EE 平台和微软的 COM/DCOM 等平台。不同中间件技术采用了不同的手段,造成了在跨越不同中间件系统时出现了较大的鸿沟。尤其在当前经济全球化,各个企业的各种应用系统需要重建,集成的需求下,解决"中间件孤岛"的问题显得格外重要。

最新的 Web Services 技术[1]基于面向服务的体系结构 SOA(Service Oriented Architechture),采用 Internet 通信协议和 XML 编码传输消息,使用标准方法进行服务描述、服务发布和服务发现,提供应用级的软件重用和业务集成,提供了一个"中间件的中间件"层次的抽象,成为了解决中间件互操作问题的最佳解决方案。文章从将 CORBA 对象封装为 Web 服务[2],使得用户可以通过 Web 技术访问 CORBA 对象的研究出发,希望可以给其他中间件系统上的应用桥适配到 Web 服务提供一个良好的借鉴。

## 1 CORBA 与 Web 服务技术比较

我们从互操作协议、互操作接口定义语言、互操作查找方式几个方面来对 CORBA 和 Web 服务[3,4]作比较。

互操作协议。通信双方之间关于消息中数据的格式、消息的类型等约定构成了这里所说的互操作协议,它可以向不同的底层协议进行映射。Web Services 采用了基于 XML 的 SOAP[3](简单对象访问协议 Simple Object Access Protocol)作为其互操作协议,可以方便地扩充并且易于理解。CORBA 的消息通信机制 GIOP/IIOP,GIOP 协议是通用的 ORB 间的传输协议,GIOP 映射在 TCP/IP 上就是 IIOP(Internet Inter-ORB Protocol)协议。

互操作接口定义语言。是一组相关操作的集合,定义了由服务器所提供的服务的信息,例如:服务名字、服务参数、参数属性等。Web Services 采用 WSDL[4]作为其互操作接口定义语言。WSDL 也是基于 XML 的,在数据类型的扩充和表示上有着其天然的优势,WSDL 分离了服务的接口和服务的绑定,大大增强了对服务细节的描述能力。IDL 接口定义语言是 CORBA 的互操作接口定义语言。

互操作查找方式。这里互操作查找方式是指服务的定位,涉及到服务信息的发布,服务信息的管理等问题。Web Services 采用 UDDI(Universal Discovery Description and Integration)作为其服务查找的方法。UDDI 注册中心是一个公有的机构,UDDI 是一个开放化的技术。名字服务是 CORBA 系统常用的定位机制,CORBA 服务器在启动时将 CORBA 对象的逻辑名注册在名字服务中,CORBA 客户通过逻辑名获取对象引用,从而使得对象引用对客户透明。

作为新一代的面向 Internet 的应用集成框架,Web 服务代表了一种更松散的分布应用结构,能够彻底屏蔽平台差异,在保护现有投资的基础上以最小成本获取企业间最大程度的应用集成和数据交换。不管在什么中间件平台上实现的应用逻辑都可以发布成 Web 服务,并将 Web 服务注册到 UDDI 注册中心,基于其他中间件系统的应用就可以从 UDDI 注册中心查找获取服务描述,通过 SOAP 消息来访问该服务。

## 2 系统设计与实现

我们设计一 CORBA-SOAP 网关来桥接 Web 服务与 CORBA 技术,该网关支持将 CORBA 对象部署为 Web 服务,

*)本课题得到 863 高科技项目基金资助:"基于 Web Services 的应用集成关键技术研究"(No. 2002AA116040)和"网络环境的新一代中间件核心技术与运行平台"(No. 2001AA113020)。毛春丽 副教授,硕士,主要研究方向:分布式计算、网络安全;贾 焰 教授,博士生导师,主要研究方向:数据库、分布计算;周 斌 副教授,博士,主要研究方向:分布计算、数据挖掘技术。

从而为使用CORBA技术开发的应用系统提供Web服务访问接口。当SOAP请求到达服务器的时候，CORBA-SOAP网关根据服务部署信息将请求适配给后端的CORBA对象，并将处理结果返回给SOAP客户端。CORBA-SOAP网关能够完成这两个中间件系统在前面所阐述的互操作体系几个层次的相互转换，也就是互操作查找方式、互操作接口定义和互操作协议间的转换，它屏蔽了后台服务的具体实现及请求的分发和激活，提供了对象定位的透明性、编程语言的透明性、平台的独立性、激活机制与散转机制之间的透明性，使得SOAP客户在调用Web服务时不感知提供服务的后端CORBA对象。

### 2.1 系统结构

我们设计的CORBA-SOAP网关[2]主要包含如下模块：服务管理模块、接口描述转换模块、目标端点映射模块及SOAP/IIOP协议转换模块。

(1)服务管理模块：主要实现服务部署信息(维护服务描述信息)及元信息(存储部署CORBA对象的操作参数元信息，确保协议转换为正确编码)的管理，具体包括部署服务、去服务、列表服务、查询服务等；

(2)接口描述转换模块：主要实现CORBA的IDL语言到Web服务的WSDL语言的映射，以便将CORBA对象部署为Web服务；

(3)目标端点映射模块：主要实现将SOAP请求目标URI转化为CORBA对象IOR及通过调用SOAP/IIOP协议转换模块的功能将SOAP请求翻译成CORBA调用(SOAP编码转换为相应的IIOP编码)；

(4)SOAP/IIOP协议转换模块：主要实现类型的序列化及反序列化；

在实现技术上，我们充分利用了动态机制实现请求的适配与转发，主要采用了名字服务、动态激活接口等技术提供了一个灵活、高效的、可维护的与CORBA服务器的桥接插件。

### 2.2 服务管理

服务管理模块负责管理各个服务的部署描述符(DeploymentDescriptor)和元信息，部署服务时，元信息被加入到元信息库中，处理请求时，SOAP服务器向元信息库查询来获得构造参数的必要信息。这些信息在部署一个新服务时被加入到系统中，并在该服务被去部署时从系统中删除；在运行刻，对某服务的请求将依赖于这些部署信息和元信息进行参数编解码，并最终投递到正确的后端系统(此指CORBA)。服务管理器模块还负责管理整个服务器各个部件的配置信息。系统初启时，系统按照配置信息设置属性、加载模块；系统运行期间，也可根据用户请求动态修改系统配置。系统配置信息由一个XML格式的配置文件描述，从而使得SOAP服务器成为一个可配置、可插拔的灵活的体系结构。

元信息库与部署描述符都用于在SOAP服务器端保存信息，它们的分工和区别是：部署描述符用于保存定义服务、配置服务及服务与后端实现对象之间关系的有关信息，例如服务ID、服务支持的方法、服务使用的特定类型映射和错误侦听器、将请求目标URI映射为具体后端对象实例所需要的信息等；元信息库则用于保存后端实现特定的与数据类型相关的信息，如操作(方法)参数及返回值相关的数据类型信息，它不具有任何“服务”的概念。

### 2.3 接口描述转换

IDL与WSDL分别是CORBA和Web服务的描述语言，为了将CORBA对象发布为Web服务，必须实现它们之间的转换。IDL和WSDL虽然在基本结构和功能上存在诸多相似，但是它们之间仍存在重要区别：一是IDL只描述接口，而WSDL还描述实现，这带来的问题是映射时必须为编译器提供有关Web服务实现的信息，如：调用Web服务的消息格式、访问服务操作的协议以及目的地的网络地址等。二是IDL数据类型丰富，但WSDL只有简单的数据类型。例如IDL中的复杂数据类型Array，Sequence，Struct等在WSDL中都没有，因此带来的问题是需要为IDL复杂数据类型制定标准的Schema。

CORBA-SOAP网关中，Web服务与CORBA对象的映射关系在部署时刻由服务的开发部署人员提供，并记录在部署描述符中。运行时刻，当客户请求服务的WSDL描述时，服务容器取读取服务部署信息，获得实现服务的CORBA对象的接口描述IDL，结合服务容器的运行地址信息，生成Web服务描述WSDL。基于上述服务模型，根据OMG所颁布CORBA to WSDL/SOAP Interworking Specification，CORBA对象接口中的主要描述项转换为Web服务WSDL文档中相应的描述的规则如下[5]。

接口：与CORBA对象接口对等的概念是服务的端口类型，实现服务的每一个接口映射为服务的一个端口类型，它表征服务的一个功能刻面。

操作：CORBA对象的接口中需发布的操作映射为服务相应的端口类型中的操作，每一个服务操作包含一对SOAP消息，分别是调用该操作的请求消息和应答消息；

属性：属性也是接口中可访问元素，每个属性根据其访问权限映射一个服务相应的端口类型中的get(只读属性)或get/set操作(读写属性)；

接口继承关系：WSDL不支持继承概念，因此，接口的继承关系将被平板化，基接口中的操作直接映射为该服务相应端口类型中的操作。

需要说明的是，并不是每一个接口元素都需要进行映射，只有那些对高层抽象意义的接口操作或属性才映射为服务描述。例如，某些接口中的方法仅用于内部对象之间的交互，不需暴露给Web服务的客户。

### 2.4 目标端点映射

目标端点映射是指运行时刻CORBA-SOAP网关根据SOAP请求确定后端CORBA对象的引用。为了实现目标端点的映射，一方面要以恰当的形式在WSDL中描述Web服务的访问端点，另一方面需要实现适配器对Web服务访问端点进行正确的理解并进行相应的转换。

WSDL除了描述服务的接口规范之外，还需描述服务的访问端点的信息。服务的访问端点信息由两部分构成：服务name属性与端口的location信息。生成WSDL时，WSDL中服务的name属性来自于服务id，服务访问端点的location则是指服务容器的位置信息。SOAP客户端构造请求消息时，服务id与所访问的端口标识共同定义了消息体中操作的名字空间，其形如“服务id#端口标识”，服务容器的位置信息则是作为承载SOAP消息的底层传输协议的消息传输端点，从而使得SOAP消息能够传递到正确的服务容器中。因此，CORBA-SOAP网关得到SOAP请求后，由操作的名字空间可以得到当前所请求的目标服务和端口类型。

由目标服务与端口标识得到目标对象引用时需要访问服务部署时形成的服务元信息。服务元信息中记录了实现该服

务的 CORBA 对象接口以及如何得到相应的 CORBA 对象引用等详细的信息。有两种方式提供 CORBA 对象引用信息：一种是直接在服务元信息中记录某种形式的对象引用，如 CORBA 对象的串化 IOR，这时可以直接获得后端 CORBA 对象的引用。另一种方式是将对象引用保存在一个公知的注册器(如 CORBA 的名字服务)中，服务元信息中记录该对象在注册器中的逻辑名，通过名字解析得到后端对象的引用。通常后一种方式更为灵活，能够透明地适应 CORBA 对象引用的变化，被我们所采用。

### 2.5 SOAP/IIOP 协议转换

将对服务的请求适配到具体的后端对象的过程实际上是利用 SOAP 消息中的目标对象信息和参数信息，构造对后端 CORBA 对象的请求消息的过程。因此，适配部件需要实现 SOAP 协议数据类型与后端 CORBA 对象系统协议数据类型之间的转换。

在服务端支持 CORBA 服务的开发、部署、配置过程中的类型映射，包括：CORBA 服务配置时类型声明；CORBA 服务的定位、激活中的类型查询；CORBA 服务的编解码(参数、返回值、异常)。在客户端，支持客户向 CORBA 服务所发送请求及应答的编解码。

SOAP[3]是一种以 XML 为基础的消息协议，其数据类型以 XML Schema 为基础，十分简单。而 CORBA 对象系统通常提供了比较丰富的数据类型，例如 CORBA IDL 中不仅简单类型分类详细(如表达整型的数据类型就包括 6 种)，而且还定义了特有的复杂类型，如序列、值类型、对象引用等。不过，由于 XML 本身具有良好的可扩展性，因此通过扩展 XML Schema 和 SOAP 数据类型，使得每个 IDL 类型唯一地映射为一种 XML 数据类型。产生 WSDL 文档时，CORBA 对象接口中的 IDL 类型声明映射为 WSDL 文档中相应的 XML 类型 Schema，使得 SOAP 客户能够以 XML 数据类型构造 SOAP 请求。在实现转换时遵循了 OMG 的 CORBA to WSDL/SOAP interworking 规范。

### 2.6 工作流程

我们通过图 1[1]来描述如何将 CORBA 对象发布为 Web 服务以及如何为使用 CORBA 技术开发的应用系统提供 Web 服务访问接口。

(1)调用接口描述转换模块和 SOAP/IIOP 协议转换模块，将 CORBA 的 IDL 描述转换为 Web 服务的 WSDL 描述，以便将 CORBA 对象部署为 Web 服务。

(2)服务提供者定义 Web 服务的服务描述(利用 WSDL 语言)并按照 UDDI 协议把它发布到服务注册中心。

(3)利用服务管理模块，将 IDL 描述中的服务 ID、服务支持的方法、服务使用的特定类型映射和错误侦听器、请求目标 URI 映射为具体后端对象实例所需要的信息等记录在服务的部署描述符中。

(4)WSDL 描述 WSDL 文档导入到 SOAP to CORBA Gateway，Gateway 将生成一个可以与其它 Web 客户端共享的新的 WSDL 文件。这两个文件之间的主要区别是 Gateway 所生成的文件将把 Gateway 作为服务端点，并且将 binding 和 portType 分离成一个接口 WSDL 文件。

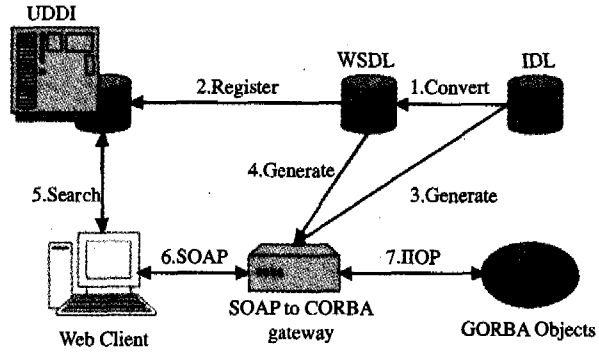


图 1 SOAP 和 CORBA 间的交互操作过程

(5)服务请求者使用 UDDI 协议执行查找操作来从服务注册中心检索用 WSDL(Web Services Description Language)定义的服务描述。

(6)然后使用服务描述按照 SOAP[2]协议与服务提供者进行绑定并调用 Web 服务实现或同它交互，服务请求者向 Gateway 发送 SOAP 请求。

(7)利用目标端点映射模块和 SOAP/IIOP 协议转换模块将 SOAP 请求信息转换为 IIOP 请求信息去定位并调用后端的 CORBA 对象，然后将所得 IIOP 应答信息转换为 SOAP 应答信息返回给 Web 客户端。

**结束语** 我们所设计的 CORBA-SOAP 网关还可实现如下功能：提供统一的客户访问策略；支持 RPC 发送阻塞、异步请求；实现脏数据检查；保留原设计风格；具有良好的灵活性、高效性和可维护性。

目前，许多企业的关键业务系统是基于 CORBA 等传统分布对象技术构建的。在 Web 服务成为实现组织间应用集成的主流技术框架的发展趋势下，将 CORBA 对象部署为 Web 服务，让 Web 客户端通过发送 SOAP 请求调用 CORBA 服务，实现基于 Web 服务的应用集成可以快速地建立或延伸现有应用程序的规模，节省系统开发所需的时间从而加快市场开发速度，具有极大的商业实用价值。

## 参 考 文 献

1 IBM Developer Works. [EB/OL]. http://www-900. ibm. com/developerworks/cn

2 国防科大计算机学院 613 教研室编. StarWebService2. 0 概要设计，2004-08

3 Simple Object Access Protocol(SOAP)1. 1,[EB/OL],W3C Note 08 May 2000. http://www. w3. org/TR/SOAP/,2001. 06

4 Web Services Description Language (WSDL)1. 1,[EB/OL],W3C Note 15 March 2001. http://www. w3. org/TR/wsdl,2001. 09

5 Object Mnagement Group. CORBA to WSDL/SOAP Interworking Specification. 2003. 11. 2