面向服务的体系结构主要实现技术比较研究

尹红丽[1] 王永明[2]
(云南师范大学计算机科学与信息技术学院 昆明 650092)[1]
(华南理工大学计算机科学与工程学院 广州 510641)[2]

摘 要 面向服务的体系结构(SOA)是一种网络环境下分布式应用系统的概念模型。在这个模型中,松耦合的系统组件在网络上被描述、发布和调用。实现这样体系结构的网络应用系统有多种方法。对CORBA,DCOM,RMI以及Web Service,Jini等几种主要实现方法进行了详细的分析并对这些方法作了对比,指出这些实现方法各自的优势和不足。分析对比的结果将为今后根据具体应用选用正确、有效的实现方式提供指导作用。

关键词 SOA,比较研究,分布式系统,软件体系结构
中图法分类号 TP242.6 **文献标识码** A

Comparative Research on Several Main Implementation Technique of SOA

YIN Hong-li[1] WANG Yong-ming[2]
(School of Computer Science and Information, Yunnan Normal University, Kunming 650092, China)[1]
(School of Computer Sci. & Eng., South China University of Technology, Guangzhou 510641, China)[2]

Abstract SOA is a conceptual model of distributed application system in the network environment, in this model, the loose coupled system components were described, released and called. There are several methods to implement the application system with SOA model. We compared and analyzed these methods, such as CORBA, DCOM, RMI, Web Service and Jini, pointed out their advantages and shortcomings. The results can be used as reference to chose implementation method.

Keywords SOA, Comparative research, Distributed system, Software architecture

1 引言

随着企业计算的发展,企业级应用需求要求新的软件系统不再是从底层做起,而是依据企业逻辑需求重新组织已有的数据存储,将现有的数据和事务通过新的渠道,比如Internet浏览器或者手持设备呈现给用户。另外,为了提高企业计算的高效性、可用性、规模性,现有许多的操作系统都是分布式操作系统,运行在许多机器之上。这样的企业级解决方案就必须协调运行在群组硬件之上,实现这种系统的一种方法就是将该系统组织成群组服务的模式,每一个服务都提供一组定义良好的功能集合。整个系统其实就被设计和实现为一组相互交互的服务,而将功能以服务的形式展现出来是该系统灵活性的关键。它使得系统中的某些服务能够充分利用其它的服务而无需考虑其物理位置。系统通过添加新的服务来不断升级,这样就应运而生了面向服务的体系结构(SOA)。W3C(The World Wide Web Consortium,万维网联盟)将SOA定义为[1]:"一种应用程序体系结构,在这种体系结构中,所有功能都定义为独立的服务,这些服务带有定义明确的可调用接口,可以以定义好的顺序来调用这些服务形成业务流程"。Gartner则将SOA描述为:"客户端/服务器的软件设计方法,一项应用由软件服务和软件服务使用者组成,SOA与大多数通用的客户端/服务器模型的不同之处,在于它着重强调软件组件的松散耦合,并使用独立的标准接口。"

传统的分布式计算模型是从本地计算模型的基础上发展而来的,为了能够最大限度减低复杂度,重用本地计算模型去解决分布式计算的问题,传统的分布式计算模型尽量向编程人员屏蔽网络的存在,所以在编程人员看来,网络编程与本地编程并没有根本的差别,编程人员不用担心网络的存在。编程人员无须区分远程地址空间和本地地址空间的区别,所有的组件均利用一种通用语言——接口语言,一般称为接口定义语言(Interface Definition Language)通过它来做到与具体编程语言无关。通过这些简化,编程人员采用本地计算模型来实现一个分布式的计算应用。人们称这种方法为"本地化设计、分布式工作"模型。CORBA,DCOM,RMI等都属于这种解决方式。Web Service和Jini是新型的动态分布式实现方案。本文将对这些实现方法的原理进行分析和对比研究。

2 CORBA(通用对象请求代理体系结构)组件实现方法

CORBA[2]遵循了通用的分布式系统解决方案模式。服务提供者通过在可用的命名目录(CORBA命名)中发布其服务对象,对此分布式对象感兴趣的客户则通过该命名服务找到服务器对象,然后就可以调用服务器对象上的合适的服务了。为了做到这一点,客户必须提前与命名服务以及服务器

到稿日期:2008-04-29 本文受云南省教育厅科学研究基金项目(08Y10326)资助。
尹红丽(1976-),女,讲师,主要研究方向为智能处理、多智能体系统;王永明(1976-),男,博士,主要研究方向为网络集成、网络机器人。

提供的各种服务绑定以得到有关的信息(比如服务器对象提供的属性、方法、接口等)。这是通过 CORBA 接口定义语言(Interface Definition Language,IDL)完成的。IDL 是中立于任何语言的,CORBA 通过 IDL 来实现与编程语言的无关性。IDL 有助于描述服务而无需深入到具体实现细节。用这种方法可以将实现(Implementation)与接口(Interface)相分离。CORBA 服务器必须实现 IDL 接口,而 CORBA 客户使用此接口。CORBA 组件之间的交互见图 1。

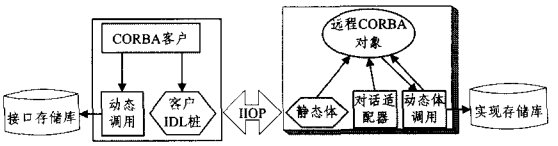

图 1 CORBA 组件

所有本地调用的方法均通过一个称为桩的客户代理完成,它将方法参数和数据进行串行化,并发送到称为体的服务器端对象,而体则需要重构此方法及其详细内容。体要负责代表客户来调用这个方法,并正确地返回结果。桩和体的交互是通过 ORB 进行的。ORB 也称为对象 BUS,它负责找到合适的服务器对象(使用服务器对象引用),并向服务器传输方法参数、数据,此外还要从服务器传回方法结果。客户与 ORB 通过一个桩实现交互,而服务器对象则使用一个对象适配器(Object Adapter,OA)或者 ORB 接口与 ORB 实现交互。客户的 ORB 与服务器的 ORB 之间通过通用跨 ORB 协议(General Inter ORB Protocol,GIOP)或者 Internet 跨 ORB 协议(Internet Inter ORB Protocol,IIOP)来通信。因此,为了使一个客户对象参与到 CORBA 环境中,它必须知道服务器的位置、服务描述以及服务调用机制。

3 DCOM(分布式组件对象模型)组件实现方法

DCOM[3] 也采用通用的分布式系统解决模式,与 CORBA 十分相似。通过对象提供服务的服务器在一个可用的命名目录(活动目录服务)中发布其对象。对此分布式对象感兴趣的客户必须知道此服务器提供的多种服务。这通过 MS-IDL 实现。服务对象实现了 IDL 接口。

所有本地调用的方法均通过一个称为代理的客户代理来完成,它将完成方法参数和数据的串行化并发送到一个称为桩的服务器对象上,在此要重构此方法及其详细内容。这个桩负责代表客户调用方法并正确地返回结果。由于 DCOM 规范是二进制级的,它可以将不同编程语言编写的二进制组件加以集成,如 C++,Java 和 Visual Basic 等。与 CORBA 不同,DCOM 只能基于 Microsoft Windows 平台,是 Windows-Windows 的分布式解决方案。DCOM 组件之间的交互见图 2。

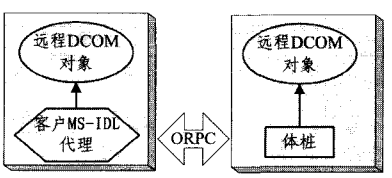

图 2 DCOM 组件

4 远程方法调用(RMI)实现方法

从 Javal.1 开始,远程方法调用(RMI)作为 Java 分布式对象技术的核心,成为 Java API 之一。它使得 Java 程序之间能够实现灵活的、可扩展的分布式通信。RMI 允许对象存在于多个指定地址空间,分布在各种 Java 虚拟机上。每个指定地址空间可以在同一计算机上或网上的不同计算机上。由于远程方法调用跨越 Java 虚拟机边界到不同的指定地址空间,因此没有对象共享的全局变量。

RMI 具有以下特点:

1. 面向对象的设计模式:RMI 能够传递完整的对象类型,包括其实现,因此可以在分布式计算解决方案中,而不仅仅在本地计算中,使用面向对象的编程;

2. 易于连接到已有的系统中:RMI 是通过 Java 的本地方法接口(JNI:Java Native Interface)与已有的系统交互的。使用 RMI 和 JNI 可以用 Java 编写新的客户端而服务端使用已有的实现,并可以根据需要对 RMI 服务端已有实现的任何部分重新编写,大大提高了软件的复用率;

4. 多线程并行计算:RMI 是多线程的,允许服务器使用 Java 线程更好地并发处理客户请求;

5. 易写易用:RMI 简化了远程 Java 服务器和访问那些服务器的 Java 客户端的编写。这种简化使得编写分布式对象系统变得容易,并且加快了系统雏形的产生和软件早期版本的测试和评估。此外由于 RMI 程序易于编写,它们同样也易于维护;

6. 分布式无用单元收集:RMI 使用它的分布式无用单元收集算法来收集远程服务器上不再被网络上任何客户引用的对象。分布式的无用单元收集可以根据需要定义服务器对象,当它们不再被客户端访问时,将被从系统中删除,从而大大提高了系统的效率。

5 Web Service 组件实现方法

Web Service[4] 作为一种分布式的服务提供方式,可以认为是 SOA 体系结构的一种实现方式,其目的是使分布在网络上不同地理位置和不同平台的客户可以获得服务。Web Service 框架体系由 3 个基本元素组成——服务提供者、客户(服务接收者)和服务注册。服务提供者将自己的服务在服务注册处进行注册,客户通过访问服务注册来获得服务的信息,然后获得服务。

与当前传统的组件技术不同,Web 服务不使用特定于对象模型的协议(比如 DCOM,RMI 或 IIOP,这些协议要求在客户机和服务器上同时具有特定的基础结构,既需要在可控制的环境中使用)。在 Web 环境中,客户机和服务器千变万化,它们使用不同的操作系统和不同的技术,因此把组件的实现紧密地绑定在特定的组件技术上是不切实际的。Web 服务是使用 HTTP,SOAP 和 XML 等普遍存在的标准协议和数据格式来通信的,因此所有支持这些协议和数据格式的系统都能支持 Web 服务。

Web 服务以消息的形式来提供服务,它使用基于 XML 的消息来作为数据通信的基本方法。这使 Web 服务完全与语言、平台和对象模型无关。Web 服务可以使用任何语言和对象模型在任意的平台上实现,任意的应用程序都可以使用

Web 服务。只要描述 Web 服务功能的接口和消息序列以及协议保持不变，Web 服务和客户应用程序可以任意改变而不会相互影响。运行时，客户应用程序发出服务调用请求，客户端代理程序把该请求转化为符合 Web Service 调用所要求的格式，通过 SOAP 协议向服务发出调用，服务接到请求后进行处理，把结果通过 SOAP 协议返回给客户。从而客户完成一次服务调用。因为 Web 服务要在异构的 Web 环境中使用，在这个环境中存在着多种不同的操作系统、对象模型和编程语言，所以 Web 服务必须具有以下特点：

1. 松散耦合：客户程序仅使用自描述的、基于文本的消息与 Web 服务进行通信；

2. 方便的通信：连接到 Internet 上的任意系统或设备都可以与 Web 服务进行通信；

3. 通用的数据格式：Web 服务使用广泛支持的 XML 标准来描述其数据格式，所有支持这个标准的系统都可以识别 Web 服务的消息。

6 Jini 组件实现方法及几种方法比较

Jini[5,6] 组件包括服务、客户和查找服务（Lookup services）。在客户开始接受服务之前，必须要经过 3 个过程：发现（Discovery）、加入（Join）和查找（Lookup），这 3 个过程同时也是通过 3 个同名的协议完成的：发现协议（Discovery）、加入（Join）协议和查找（Lookup）协议。客户和服务都可以通过发现过程来定位查找服务的位置，加入过程使得服务可以把自己注册到查找服务上去。而查找过程帮助客户查找到感兴趣的服务。

发现（Discovery）、加入（Join）和查找（Lookup）的过程简述如下：当一个 Jini 服务连入网络时，它就采用组播（multicast）的方式向网络中发出消息，表明自己的存在。网络中的查找服务则保持监听某个特定的端口，当它监听到这个消息后，就会接收消息数据包，并进行分析，以便决定是否与发送者相联系。如果它决定和发送者联系，就会通过一个 TCP 与发送者建立连接，并通过 RMI（Remote Method Invocation）向发送者回传一个服务登记对象（service register），发送者通过调用服务登记对象中的 register() 方法，作为参数上传一个 service item 对象，这就完成了发现和加入。Service item 对象是一个对象容器，其中包括一个 service object 对象，作为 Jini 服务代理（Proxy），它负责和服务器交互，它实现了一个或者若干个客户与服务交互的接口。客户同样通过发现协议（Discovery）获得一个服务登记对象（service register），然后调用服务登记对象的 Lookup() 方法（不要把它和 Lookup 服务混淆，前者是服务登记对象的一个方法，后者是整个查找服务），作为参数上传一个 service template 对象，用来作为查询准则。这个对象含有一个引用指向一组 Class objects 对象，这些对象指示查找服务，客户所需要的 Java 类型的 service object 对象。lookup() 方法返回符合客户需要的 service object 对象。每个 service object 对象都会实现（implement）一个或者若干个客户与服务交互的接口（interface）。客户通过引用（reference）来使用 service object 对象，通过这个对象来获得服务。

与一般的分布式系统努力掩盖网络存在的做法不同，Jini 努力去适应网络存在这个事实。比如网络可靠性的问题，Jini 是自愈合的，这一点是通过 Jini 的租用机制实现的。租用机制使得服务经过一定的时间，自动地从联邦中移除，而无需管理员维护。服务提供者要和查找服务协商对查找服务的租用期，并且维护一个租用使它保持有效，否则查找服务就会取消服务注册。而客户也要和服务提供者协商服务租用期。当网络发生变化时，Jini 使得管理员无需做改变软件中的 URL、更新资源文件、为客户机安装服务代理等工作，而这些在传统的分布式计算体系中是很正常的。比如 CORBA，.NET，J2EE 等都需要静态的安装（需要人工安装且不能动态的更新）客户桩（stub）和服务器端骨架（skeleton），而 Jini 通过代码移动解决了这个问题，这也是 Jini 最强有力的一个特点。Jini 采用如下方法解决分布式计算中存在的常见问题：1）利用 Java 来克服编程模型和编程语言的问题；2）提供了一个新的基础设施。

Jini 还提供了一种分布式事件机制，这是对 JavaBeans 事件模型的扩展，由此服务请求者可以在服务提供者处注册其所需服务。服务一旦可用，服务请求者还能获得服务通知，因此不再需要任何轮询。部分失败是通过租用机制加以处理的。租用为服务的使用引入了一个时间的概念。在 Jini 中，允许服务请求根据实际需要的时间维持与服务提供者的连接。在 Jini 环境中，租用可以在某个特定时间段将服务者列在查找服务中，如果超出此时间段，则需要重新协商一个新的租用，或者干脆将它从查找服务中移除。这样就可以支持建立一个动态的网络，服务提供者能够随时离开或重新加入而不会影响到其它系统。此外，由服务请求者与服务提供者协商得到的租用，还可以控制某个服务提供者的服务请求者。下面表 1 是 Jini 对相关问题的处理方法，表 2 是几种分布式计算体系结构比较研究。

表 1 Jini 的问题处理方法

问题	Jini
	编程模型
网络延迟	通过 Jini 远程接口区别本地对象/组件和远程对象/组件
部分失败	使用租用机制和发现协议加以处理
并发性	通过事务管理器接口加以处理
服务提供者和服务请求者之间的耦合	由于代理对象将按需下载到客户端，因此没有更多的紧耦合。服务提供者与服务请求者之间的关系是动态的，服务期的长短可以依据服务请求者和服务提供者的协商而定。
	编程语言
地址空间	采用 Java 语言，不支持指针，区别本地对象和远程对象
	基础设施相关
客房获得所需服务的通知机制	提供一个分布式事件机制，服务请求者可以在某个服务提供者处注册其所需的服务，并在服务可用时得到通知。
服务请求协议	与协议无关，可以包容任何协议：JRMP，IIOP，ORPC，SOAP 或某个专用的协议。

表 2 几种分布式计算体系结构的问题处理方法比较

问题	CORBA	DCOM	RMI
		编程模型	
网络延迟	没有显式区别本地对象和远程对象	没有显式区别本地对象和远程对象	显式区别本地对象和远程对象
部分失败	无法处理	无法处理	无法处理
并发性	并发是一个可选服务	可以得到并发服务	没有处理并发

服务提供者与客户的耦合	存在紧耦合，客户桩(stub)需要由客户应用进行编译	存在紧耦合，客户桩(stub)需要由客户应用进行编译或者导入	无紧耦合，客户桩(stub)可以在运行时下载到客户
通知客户获得所需服务的机制	没有可用的分布式事务通知机制	没有可用的分布式事务通知机制	使用 Java 事件通知模型
区别本地和远程地址空间	没有显式区别本地对象和远程对象，需要在本地建立远程对象的对象引用	没有显式区别本地对象和远程对象，需要在本地建立远程对象接口的专用指针	区别本地和远程对象。使用3种技术来处理远程交互：1)建立对远程对象的引用；2)非远程对象可以串行化并以传值的方式传递；3)简单数据类型也可以利用传值的方式传递
编程语言及基础设施相关			
何种编程语言	多种编程语言	多种编程语言	Java
应用组件之间的交互	所有远程对象使用 CORBA 接口定义语言(CORBA-IDL)进行发布	所有远程对象使用 COM 接口定义语言(COM-IDL)进行发布	所有远程对象使用 Java 远程接口发布，并实现 Java 远程接口
请求协议	依赖于通用 ORB 协议(GIOP)通用 Internet 跨 ORB 协议(IIOP)	依赖于对象远程过程通信协议(ORPC)	依赖于 Java 远程方法协议(Java Remote method Protocol，JRMP)
操作系统	具有平台无关性	Microsoft Windows 系列	具有平台无关性，但需要安装 Java 虚拟机(JVM)
应用的网络方面	ORB 负责处理数据的调用和格式转换	ORPC 负责处理数据和调用格式的转换	不需要特别数据/调用格式的转换，因为是在 JVM-JVM 之间进行，利用 Java 串行化技术通信

结束语 SOA 的设计目标是以服务为基础，通过服务的交互来实现系统动态、松耦合集成，极大地降低了复杂性与成本。但目前为止没有文献对实现它的方法进行明确的论述，造成人们在实现它时总是无从下手。本文在深入分析了 SOA 实现方法机理的基础上，提出各自的优点和不足之处；并将主要方法作横向比较研究。Jini 是一种基于 Java 的分布式系统，具有许多其它分布式模型没有的优点，是如今使用最多的分布式计算编程模型。但在如今多种模型并存的现实情况下，Jini 不可能完全代替 CORBA，DCOM，RMI，基于 UDDI 的 Web Service 等分布式计算体系，所以应该努力解决这些分布式计算体系之间共存并且相互操作的问题，这样不仅能够充分利用不同的分布式计算体系的优点，还可以巧妙地避免彼此的缺点。而且从应用的层次上来看，用户并不希望被绑定在一家公司的产品上，而希望有更多的选择余地，所以，下一步的研究应着重于解决这些分布式系统之间互操作的方法，它不仅有理论上的意义，还具有现实的意义。

参 考 文 献

[1] Wu Jie. Distributed System Design. 高传善，等译. 北京机械工业出版社，2001：5-15

[2] 李兵，陈鸣. CORBA 对分布式管理体系结构的支持. 解放军理工大学学报：自然科学版，2001(4)：39-43

[3] 徐泽华，王耀南，罗瑞琼. 基于 DCOM 的分布式计算机控制系统的设计. 工业控制计算机，2001(6)：7-10

[4] 程炜，杨宗凯，乐春晖. 基于 Web Service 的一种分布式体系结构. 计算机应用研究，2002(6)：105-107

[5] Kumaran S. Jini 技术指南. 林琪，欧阳宇，等译[M]. 北京机械工业出版社，2003：10-20

[6] Eckel B. Thinking in Java [M]. 2nd edition，Revision 12，2000：50-65

(上接第 266 页)

信念网络[12]等来学习测试用例。因此，未来工作主要是多种分类学习模型的应用以及进一步的实例研究。

参 考 文 献

[1] Hamlet R. Random testing. Encyclopedia of Software Engineering. John Wiley and Sons，1994

[2] Rumbaugh J，Jacobson I，Booch G. The Unified Modeling Language User Guide. Boston：Addison-Wesley，2001

[3] Wang L，Yuan J，Yu X，et al. Generating Test Cases from UML Activity Diagram Based on Gray-Box Method // Proc. of APSEC'04. IEEE Computer Society，New Jersey，2004：284-291

[4] Chen T，Poon P，Tang S，et al. Identification of Categories and Choices in Activity Diagrams // Proc. of the 5th International Conference on Quality Software (QSIC'05). IEEE Computer Society，2005：55-63

[5] Chen M，Qiu X，Xu W，et al. UML Activity Diagram-based Automatic Test Case Generation for Java Programs. The Computer Journal，2007，doi：10.1093/comjnl/bxm057

[6] Chen T Y，Huang D H，Zhou Z Q. Adaptive random testing through iterative partitioning // Proc. of the 11th International Conference on Reliable Software Technologies，LNCS 4006. Berlin Heidelberg：Springer-Verlag，2006：155-166

[7] Ciupa L，Leitner A，Oriol M，et al. Object distance and its application to adaptive random testing of object-oriented programs // Proc. of the 1st International workshop on Random Testing (RT'06). ACM Press，2006：55-63

[8] Grochtmann M，Wegener J，Grimm K. Test case design using classification trees and the classification-tree editor CTE // Proc. of the 8th International Software Quality Week (QW'95). Software Research Institute，San Francisco，CA，1995

[9] Singh H，Conrad M，Sadeghipour S. Test Case Design Based on Z and the Classification-Tree Method // Proc. of the 1st International Conference on Formal Engineering Methods (ICFEM'97). IEEE Computer Society，1997：81-90

[10] Krupp A，Mueller W. Classification Trees for Random Tests and Functional Coverage // Proc. of the Conference on Design，Automation and Test in Europe，EDAA. 2006：1031-1032

[11] McMinn P. Search-based Software Test Data Generation：A Survey. Software Testing，Verification and Reliability，2004，14(2)：105-156

[12] Han J，Kamber M. Data Mining：Concepts and Techniques. Morgan Kaufmann Publishers，2001：279-296

[13] Kolman B，Busby R，Ross S. Discrete Mathematical Structures. Fourth Edition. Prentice Hall，Inc.，2001

[14] Blaha M，Rumbaugh J. Object-oriented Modeling and Design with UML. Second Edition. Pearson Education，Inc.，2005