

基于 Ajax 和 CORBA 中间件的分布式订单管理系统

郑 勇¹, 卢捍华², 孙雁飞², 闵丽娟², 王亚石²

(1. 南京邮电大学 计算机学院, 江苏 南京 210003;

2. 南京邮电大学 信息网络研究所, 江苏 南京 210003)

摘 要: 为了提高订单管理系统开发的灵活性和开发效率, 结合 Ajax 和 CORBA 中间件技术设计和实现了基于 J2EE 的分布式订单管理系统。该系统采用多层 Web 架构, 前端采用 Ajax 技术交互, 后端利用 RMI - IIOP 整合 EJB 和 CORBA 通信, 底层数据库操作采用基于 C++ 的 CORBA 实现, 整个系统充分利用了 Ajax 和 CORBA 中间件的优势, 把表示逻辑和业务逻辑分离, 并利用 CORBA 语言无关性, 采用 IDL 映射实现了 Java 开发和后端 C++ 开发的技术透明性, 提高了基于 J2EE 的分布式系统的开发效率。

关键词: 订单管理系统; 中间件; 分布式; 业务逻辑

中图分类号: TP311.52

文献标识码: A

文章编号: 1673-629X(2009)08-0201-04

A Distributed Order Management System Based on Ajax and CORBA Middleware

ZHENG Yong¹, LU Han-hua², SUN Yan-fei², MIN Li-juan², WANG Ya-shi²

(1. Computer Institute, Nanjing University of Posts & Communications, Nanjing 210003, China;

2. Information Network Institute, Nanjing University of
Posts & Communications, Nanjing 210003, China)

Abstract: An order management system based on Ajax and CORBA is proposed by this paper, in order to improve the efficiency and flexibility of distributed system developing. The system adopts multi-layer Web architecture and technology of middleware. Ajax is used at foreground to finish Web interactions, while RMI - IIOP is used to integrate EJB and CORBA. The bottom database operations are implemented by CORBA which is realized by C++ programming language. The whole system takes advantages of Ajax and CORBA. Presentation logic and business logic is clearly separated, and technique transparency between Java developing and C++ developing is maintained by IDL mapping. As a result, efficiency of distributed system developing would be improved obviously and flexibly.

Key words: order management system; middleware; distributed; business logic

0 引言

订单管理系统是企业信息化发展的重要组成部分, 随着面向服务的架构 (Service Oriented Architecture, SOA)^[1~3] 和企业应用集成 (Enterprise Application Integration, EAI)^[4,5] 的发展, 订单管理系统作为 EAI 系统的一部分, 对交互能力和通信功能的需求大大增

强, 同时分布式系统的开发越来越复杂, 对服务器和数据库的压力也越来越大。在这种形势下, 如何进一步改进和提高系统的灵活性和开发效率成为一个焦点问题。

Ajax 是 Asynchronous JavaScript and XML 的简称, 是最新的网络客户端综合技术, 可以在 B/S 构架下实现富客户端, 可用于构建面向异步消息的无刷新的网络应用^[6,7]。而 CORBA (Common Object Request Broker Architecture) 是一种标准的面向对象应用程序体系规范, 它为可移植的、面向对象的分布式计算应用程序提供了不依赖于平台的编程接口和模型, 具备不依赖于编程语言、计算平台、网络协议的特点, 提供了面向对象应用的互操作标准, 是一种标准的面向对象应用程序体系规范^[8]。

文中利用 Ajax 和 CORBA 技术的特点和优势, 设

收稿日期: 2008-11-25; 修回日期: 2009-02-16

基金项目: “十一五”国家科技支撑计划 (2007BAH17B04); 国家高技术研究发展计划 “863” 基金资助项目 (2009AA01Z212, 2009AA01Z202); 江苏省自然科学基金 (2007BK603); 江苏省高技术研究计划项目 (BG2007045); 南京邮电大学攀登计划项目 (NY2007044); 南京邮电大学人才引进项目 (NY2007044); 福建富士通公司资助项目
作者简介: 郑 勇 (1985-), 男, 山东莱芜人, 硕士研究生, CCF 会员, 研究方向为计算机在通信中的应用; 卢捍华, 高级工程师, 硕士生导师, 研究方向为电信 BSS/OSS, CRM, SOA 等。

计和实现了基于 J2EE 的订单管理系统,为了提高系统的处理能力和系统性能,后端数据库操作采用较为高效的基于 C++ 的 CORBA 实现,并且通过 IDL 映射关系和 RMI-IIOP^[9,10]通信机制与 J2EE 的 EJB 进行交互和处理,充分发挥了 CORBA 语言无关性的优势,并且通过 IDL 的接口定义保证了前台开发和后台开发的透明性,清晰彻底地把表示逻辑和业务逻辑相隔离。

1 订单管理系统需求

一个完整的订单管理系统可能包括销售与采购系统、物流系统、订单管理系统、财务系统、账务统计系统等。一个完整的订单流程可以用图 1 来表示。

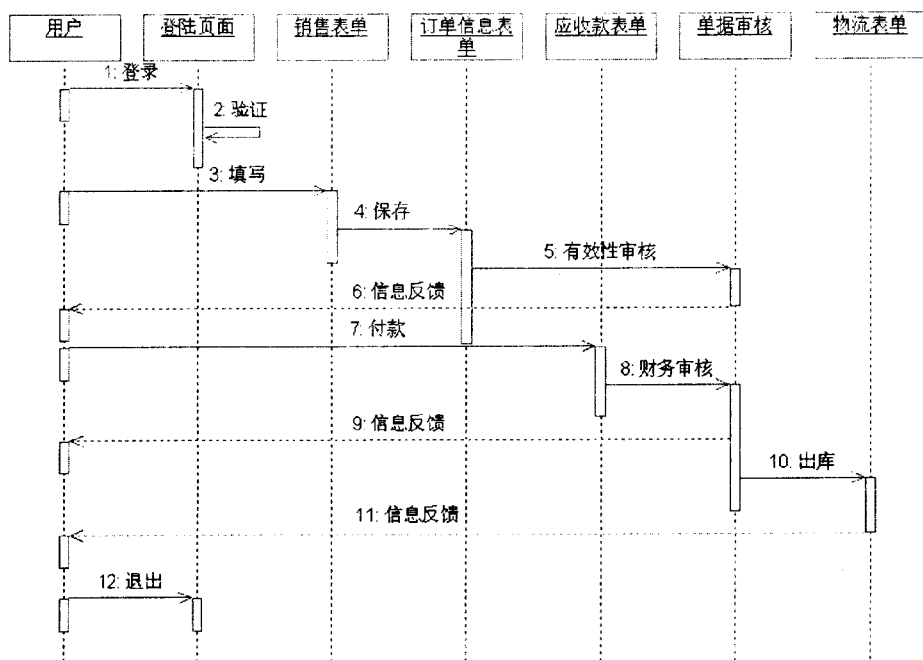


图 1 订单流程时序图

2 订单管理系统分析与设计

订单管理系统整体上是一个基于 J2EE 的 B/S 多层 Web 分布式系统,如图 2 所示。本节先讲述系统总体设计,然后以客户信息查询为例给出各个模块的设计方法。

2.1 系统总体设计

浏览器作为客户请求端,通过 DHTML 和 Ajax 进行 Web 交互。客户端对服务端的请求以及服务端的响应接收都是通过 XMLHttpRequest 操作和交互来实现的。

服务器端包括 Weblogic 应用服务器、CORBA 服务器和数据库三大部分。Weblogic 服务器用于承载和部署 Web 应用,包括 JSP, Servlet, JavaBean 以及 EJB。

在这里 EJB 是作为调用请求和 CORBA 服务的桥梁,通过 RMI-IIOP 方式与后台 CORBA 的 C++ 实现通信并传递往返数据的。

CORBA 服务器这里选用 Tuxedo 中间件。考虑到 Java 运行时(Java Runtime)本身的效率问题,特别是当订单管理系统处于 EAI 系统中,需要与诸多系统,如 CRM 系统、集团账务系统等进行交互,对性能的需求也越来越高,所以采用效率比较高的 C++ 语言来编码和具体实现 CORBA。Java 端和 C++ 端都基于统一的 IDL 定义来设计和实现,然后通过 WTC (Weblogic Tuxedo Connector)和 IDL 映射关系来统一 CORBA 服务,进而可以使得 EJB 桥接 RMI 和 IIOP,通过向 CORBA 的 CosNaming 服务进行查询和映射完成对

CORBA 服务端的调用^[11]。事实上这里的 EJB 是作为 CORBA 服务的客户端,而真正的后台实际操作是由基于 C++ 语言实现的 CORBA 服务来完成的。

CORBA 的实现部分为了提高效率,被设计成了一个多层次的可扩展的模型。首先通过调用通过 IDL 定义的接口找到 CORBA C++ 实现部分的接口类中对应的函数,函数根据业务逻辑需求访问后端的操作类,操作类里面封装了对数据库的基本操作,如查询、插入、删除、更新等,操作类访问数据

持久层的表类,而这个表类是与后台数据库匹配和关

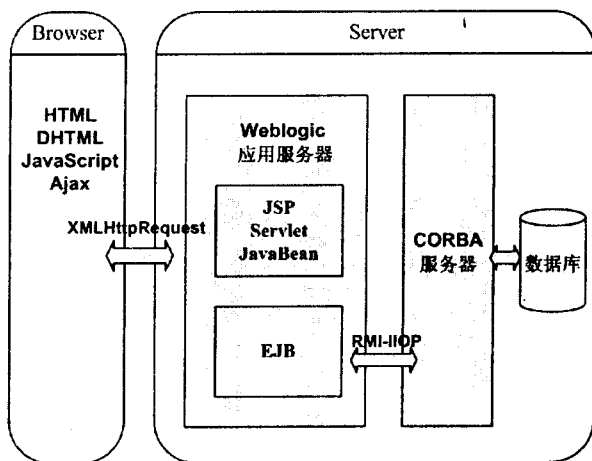


图 2 系统架构图

联的,实际上就是数据库实体表的对象持久化。

下面将以客户信息查询为例简述各个模块的设计原则和设计方法。

2.2 CORBA IDL 的设计

IDL 是一种说明性的接口定义语言,允许对象接口以任何具体编程语言无关的形式来定义和实现接口,是 CORBA 支持异构系统和独立开发的应用程序集成的关键^[10]。在本系统中,IDL 为前台开发和后台开发定义了映射关系,前台只需知道接口而无须明晰接口实现即可实现远程调用。

客户信息查询的 IDL 可以按如下方式定义:

```
module Customer {
interface ICustomer {
    short searchCustomerInfo
        (in string str-input,
         out string str-output);
}
```

2.3 客户端 Ajax 调用

客户端发起查询客户信息的请求主要是利用 Ajax 技术通过编写 JavaScript 远程脚本^[12]来实现的。远程脚本实质上就是一种 RPC 模式,它可以用图 3 来描述。

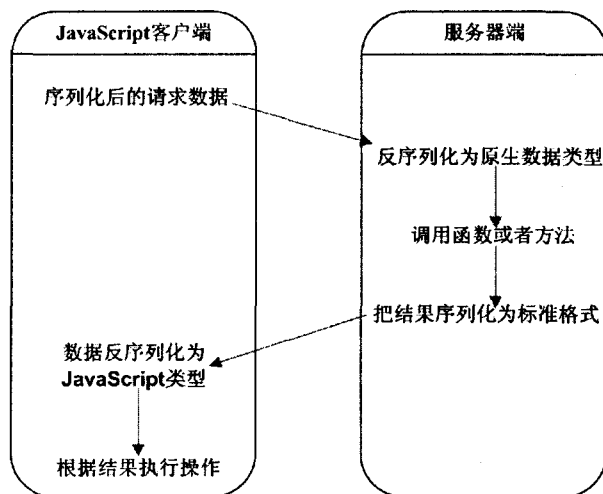


图 3 JavaScript 远程脚本原理图

以客户信息查询为例的远程脚本如下:

```
remoteCall ("customerService. searchCustomerInfo", [ condition ], function(reply) { });
```

这里的调用名称采用 searchCustomerInfo 与 IDL 接口中定义的方法名称是保持一致的。

2.4 服务器端处理

根据远程脚本的原理,服务器端需要先把请求传递过来的数据信息进行反序列化,然后调用 CORBA 的 C++ 实现,最后再把结果序列化并且反馈给客户端。

服务器端的 Java 和 EJB 就是负责数据的反序列化,以及与 CORBA 通信的。

数据的反序列化实质上就是 XML 数据到 Java Object 数据的转换。而服务器端的核心在于 EJB 通过 RMI-IIOP 与后端 CORBA 通信。文献[11]提供了两种 EJB 与 CORBA 通过 RMI-IIOP 通信的方式,一种是 CORBA 作为 EJB 的服务端,另一种是 EJB 作为 CORBA 的服务端。在本课题中显然是 EJB 作为 CORBA 客户端调用 CORBA 服务的,其原理如图 4 所示。

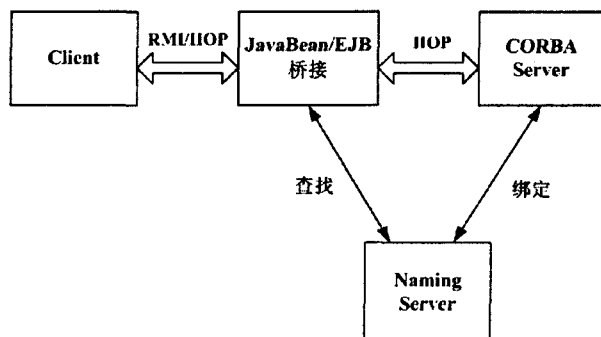


图 4 CORBA 为服务端的 EJB/CORBA 集成^[11]

通过 IDL 统一接口定义,EJB 客户端向 Naming Server 查找相应的服务,即可调用 CORBA 服务实现,正因为这种调用机制和 CORBA 的语言无关性,才能使得 EJB 可以调用后端 C++ 的 CORBA 实现:

```
// Java 端数据反序列化
org.exolab.castor.xml.Marshaller.marshal(condition,w);
//通过 EJB 调用 CORBA
result=customerClient.callCorba("searchCustomerInfo",w.toString());
// EJB 调用 CORBA 过程
orb=(ORB)new InitialContext().lookup("java:comp/ORB");
org.omg.CORBA.Object finder_oref=orb.string_to_object("corbaloc:tgio: test/FactoryFinder");
FactoryFinder finder_ref=FactoryFinderHelper.narrow(finder_oref);
org.omg.CORBA.Object customer_oref=finder_ref.find_one_factory_by_id(idl.Customer.CustomerFactoryHelper.id());
CustomerFactory customerFactory_ref=CustomerFactoryHelper.narrow(customer_oref);
//返回查找的客户信息
Customer=customerFactory_ref.findICustomer();
```

CORBA 服务的实现是基于 C++ 编程语言的实现,整个过程是先由接口类根据方法名定位实现函数,函数调用操作类里操作数据库的方法,数据库操作函数操作持久化的数据库表类并对后台数据库进行查询

和更新操作。

3 结束语

Ajax 和 CORBA 技术都是当前流行的设计技术。Ajax 作为一种基于 JavaScript 和 XML 的客户端技术,不仅增进了客户端体验,同时也改进了软件体系设计特别是对服务器的调用设计。CORBA 作为开放、标准的规范体系使得各种中间件技术可以相互支持,特别是 CORBA 的语言无关性使得本系统中 EJB 和 C++ 实现的调用成为可能。

文中结合 Ajax 和 CORBA 技术设计和实现了基于 J2EE 的分布式订单管理系统。该系统利用 Ajax 和 CORBA 的优势把表示逻辑和业务逻辑向分离,同时利用 CORBA 的语言无关性和 RMI-IIOP 通信机制,实现了 Java 调用和 C++ 实现的编码透明性,有利于开发者分工和协同开发,提高了基于 J2EE 的分布式系统的开发效率。

参考文献:

- [1] OASIS SOA Reference Model TC. Reference model for Service Oriented Architecture 1.0[EB/OL]. 2006. <http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>.
- [2] Stal M. Using architectural patterns and blueprints for service

-oriented architecture[J]. Software, IEEE, 2006, 23(2): 54-61.

- [3] 童鑫,李军义. 面向 SOA 的企业服务总线研究与实现[J]. 计算机应用, 2008, 28(3): 819-822.
- [4] 彭武良,周丽,王雷. 企业应用集成技术综述[J]. 计算机应用研究, 2007, 24(9): 12-15.
- [5] 蔡萍,华庆一. 基于 SOA/EDA 电信企业应用集成技术研究[J]. 计算机技术与发展, 2007, 17(11): 560-561.
- [6] 赵永屹,宿红毅,胡韶辉. 基于 AJAX 与 J2EE 的新型 Web 应用的设计与实现[J]. 计算机工程与设计, 2007, 28(1): 189-192.
- [7] 田福生,张燕平. 用 Ajax 技术实现 B/S 模式下客户端间信息交互[J]. 计算机技术与发展, 2007, 17(10): 38-39.
- [8] 代霞,黄劲松. 基于 CORBA 综合网络配置管理的设计与实现[J]. 计算机技术与发展, 2008, 18(2): 91-93.
- [9] Oh Joo-Yong, Park Jun-Ho, Jung Gi-Hoon, et al. CORBA based core middleware architecture supporting seamless interoperability between standard home network middlewares[J]. IEEE Transactions on Consumer Electronics, 2003, 49(3): 581-586.
- [10] 韦兆文,刘波. 利用 RMI-IIOP 实现 EJB 容器之间的通信[J]. 计算机应用, 2003, 23(11): 128-133.
- [11] 夏丹. EJB/CORBA 集成应用研究与实现[D]. 武汉: 武汉理工大学, 2006.
- [12] Eichorn J. 深入理解 Ajax: 基于 JavaScript 的 RIA 开发[M]. 北京: 人民邮电出版社, 2007.

(上接第 200 页)

主要是因为对路由器的要求太高。面对这种情况近几年出现了应用层组播策略,这种策略将组播策略的实现转移到了各个终端,从而避免了改变网络配置,同时也减轻了路由器的负担。而怎样使应用层组播策略、IP 层组播策略和原始 TCP 流量控制合作更密切,是目前非常值得研究的一个问题。

参考文献:

- [1] Forouzan B A. TCP/IP 协议族[M]. 北京: 清华大学出版社, 2006: 372-375.
- [2] Bhattacharyya S, Towsleu D, Kurose J. The loss path multiplicity problem in multicast congestion control[C]//INFOCOM apos 99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. New York, NY, USA: [s. n.], 1999: 856-863.
- [3] 苏晓丽,郑明春,孟强. TCP 友好多播拥塞控制算法研究[J]. 计算机工程与科学, 2005, 25(4): 26-29.
- [4] 潘国庆,李陶深. 一种基于策略函数的应用层组播路由算

法[J]. 计算机技术与发展, 2008, 18(5): 138-140.

- [5] Ballardie A. Core Based Trees(CBT) Multicast Routing Architecture[S]. RFC 2201, 1997.
- [6] 苏晓丽,郑明春,孟强. 多播拥塞控制研究进展[J]. 通信学报, 2003, 24(5): 94-104.
- [7] 程传庆. IP 组播组管理协议及其在二层的实现[J]. 信息技术, 2003, 27(7): 50-52.
- [8] Waitzman D, Partridge C, Deering S. Distance vector multicast routing protocol[S]. RFC 1075, 1988.
- [9] Estrin D, Farinacci D, Helmy A, et al. Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification[S]. RFC 2362, 1998.
- [10] Ballardie A, Cain B, Zhang Z. Core Based Trees (CBT version 2) Multicast Routing: Protocol Specification[S]. RFC 2189, 1997.
- [11] Moy J. Multicast Extensions to OSPF[S]. RFC 1584, 1994.
- [12] Nagle J. Congestion Control in TCP/IP Internetworks[J]. Communications Review, 1984, 14(4): 11-17.