

“十五”国家重点电子出版物规划项目 计算机知识普及和软件开发系列

116

应用技术丛书

Java

分布式应用程序设计

北京希望电子出版社 总策划
苏 洋 编 写



北京希望电子出版社
Beijing Hope Electronic Press
www.bhp.com.cn

内 容 简 介

本书是基于 CORBA 规范的 Java 分布式程序设计指导书。

全书由 19 章组成, 主要内容包括: 分布式应用模式概览、CORBA 体系结构的基本内容、OMG IDL 语法结构、OMG IDL 与 Java 语言的映射关系、idl2java 映射文件分析、VisiBroker 的基本内容、基于 VisiBroker 开发分布式应用、可移植对象适配器、基于动态调用接口和动态框架接口创建分布式应用、定位服务的应用、命名服务的应用、事件服务的应用、接口仓库的应用、对象激活进程的应用、URL 命名服务的应用、基于 JBuilder5 和 VisiBroker 开发分布式应用、模拟人力资源管理分布式应用实例、模拟 ATM 机分布式应用实例、COM 与 CORBA 的对比。

本书提供了 28 个 IDL 接口定义文件和 376 个独立编写的以及由 IDL 接口定义文件映射生成的 Java 源程序。本书结构清晰, 内容翔实, 实例典型, 边讲边练。特别适合具有一定 Java 语言基础的初、中级用户, 同时也能为高级用户提供有力的参考, 而且可作为大专院校师生教学与自学的参考书以及社会初、中级培训班教材。

需要本书源代码的读者可在 www.b-xr.com 免费下载, 文件名为 3834.zip。

系 列 书 名 : “十五” 国家重点电子出版物规划项目 计算机知识普及和软件开发系列
Java 应用技术丛书 (1)

书 名 : Java 分布式应用程序设计

总 策 划 : 北京希望电子出版社

文 本 著 者 : 苏洋 编写

责 任 编 辑 : 杨如林

出版、发行者 : 北京希望电子出版社

地 址 : 北京市海淀区知春路63号卫星大厦三层 100080

网址: www.bhp.com.cn

E-mail: lwm@bhp.com.cn

电 话 : 010-62520290, 62521724, 62528991, 62630301, 62524940, 62521921, 82610344

(发行) 010-62613322-215 (门市) 010-82675588-501, 82675588-201 (编辑部)

经 销 : 各地新华书店、软件连锁店

排 版 : 希望图书输出中心 全卫

文 本 印 刷 者 : 北京媛明印刷厂

开本 / 规格 : 787 毫米×1092 毫米 1/16 19.75 印张 451 千字

版次 / 印次 : 2002 年 7 月第 1 版 2002 年 7 月第 1 次印刷

印 数 : 1-4000 册

本 版 号 : ISBN 7-900101-39-X

定 价 : 32.00 元

说明: 本社产品如有残缺, 可凭相关凭证与本社调换。

前言

互联网技术及电子商务应用技术的跳跃式发展，极大地推动了能够处理商家对商家（B2B）和商家对客户（B2C）之间交互的新一代企业应用程序的开发。Java 作为创建这种应用程序的主要语言而出现，它能够被大量的分布式应用系统所采用的主要原因在于其面向对象的编程思想和跨平台的移植性以及完善的安全性管理，同时它还提供了一个具有丰富的基础类库和开发工具的应用平台。

作为分布式应用对象构建规范的公共对象请求代理体系结构规范（CORBA）是目前最具生命力的跨平台技术，它独立于网络协议、编程语言和软硬件平台，支持异构的分布式计算和不同编程语言的对象重用。基于目前流行的 C++ 和 Java 等面向对象的程序设计语言，中间件软件开发商开发了相应的 CORBA 规范实现产品，如 VisiBroker、Orbix 等。正是基于 CORBA 产品的编程语言无关性和 Java 语言的平台无关性，目前众多的分布式应用软件系统采用“Java + CORBA 产品 + WEB”的设计模式。

本书采用 Java 语言作为程序设计语言，以实现了 CORBA2.2 规范的 VisiBroker4.5.1 作为中间件产品，详细讲解了 CORBA 体系结构规范的主要内容以及 VisiBroker 的产品实现形式。全书共 19 章，分五个部分系统地介绍了 CORBA 的概念和体系结构、IDL 语义和映射为 Java 语言的规则、CORBA 基本对象和 CORBA 服务等内容。

第一部分：分布式应用基础

早期的分布式应用系统由商务处理部分和信息存储系统构成。这种应用系统模式昂贵的初始设备投资和后期维护成本有力地推动了在逻辑上构成“瘦客户端”形式的分布式三层或多层体系结构的发展。目前的企业应用级软件系统包括前端用户界面、中间层商务规则处理和后台企业信息存储系统等。

第二部分：CORBA 规范的基本内容

CORBA 规范定义了分布式应用客户端和服务实现对象端在构成的分布式应用中的责任和义务。为此，CORBA 定义的客户端存根、服务对象框架等对象为客户端访问服务对象以及服务对象传递方法运行结果提供了事务代理。为了描述服务实现对象的服务功能，CORBA 规范定义了 IDL 接口定义语言并且规范了该描述语言到程序设计语言的映射关系。

第三部分：CORBA 规范的实现

CORBA 规范仅仅是定义了为分布式应用对象提供服务的基本框架，具体的服务实现形式依赖于中间件软件产品。VisiBroker 完整地实现了 CORBA 规范中定义的分布式应用服务对象并且提供了丰富的对象维护和管理工具。

第四部分：程序设计中应用的 CORBA 对象和服务

CORBA 规范中定义的客户端存根和服务实现对象端框架等基本对象，为分布式对象之间提供了信息交换的机制。另外，CORBA 命名服务、定位服务和事件服务等 CORBA 服务定义了应用程序实现分布式应用的多样性。

第五部分：完整示例

掌握一门语言或者一项技术，最重要的是以基于该技术开发的完整应用系统作为范例进行讲解。书中提供了分布式人力资源管理和模拟 ATM 机分布式应用的两个完整应用系统，并且在书中对程序设计中的每个关键步骤进行讲解，希望能够达到抛砖引玉的作用。

衷心地希望本书能够对想理解和掌握 CORBA 体系结构以及 Java 分布式应用程序设计方法的读者有所帮助。

在本书的创作和例子程序调试的过程中，得到了许多朋友和老师的帮助，在此表示感谢。另外，我还要深深地感谢我的妻子，是她在我最困难的时候给了我极大的鼓励和支持并承担了全部家庭重担，使得我能够安心地在外学习和创作。

由于作者水平有限，再加之时间比较仓促，有关书中的内容请读者予以批评、指正。

有关书中的问题愿与读者讨论，我的 E-mail 地址: suyang66@sina.com。

编者
于吉林大学

第1章 分布式应用模式概览	1
1.1 客户机-服务器应用模式	1
1.1.1 客户机-服务器 (C/S) 体系结构	1
1.1.2 浏览器-服务器 (B/S) 体系结构	3
1.1.3 C/S 和 B/S 体系结构的 优势与不足	3
1.2 分布式多层应用体系结构	4
1.2.1 分布式多层体系结构	5
1.2.2 分布式多层体系结构 各逻辑层的特征	7
1.3 中间件的基本概念	7
1.3.1 中间件的定义	8
1.3.2 中间件的基本特征	8
1.3.3 基于中间件的软件系统 开发和部署方式	8
1.4 小结	9
第2章 CORBA 体系结构的基本内容	10
2.1 CORBA 体系结构概述	10
2.1.1 CORBA 的概念	10
2.1.2 CORBA 规范的特点	11
2.1.3 对象请求代理的概念 和作用	11
2.1.4 CORBA 中的对象模型	12
2.2 客户端 CORBA 对象	13
2.3 服务对象端 CORBA 对象	14
2.4 客户端存根	15
2.5 服务对象框架	18
2.6 对象请求代理	22
2.6.1 ORB 传递服务请求的过程	23
2.6.2 ORB 初始化方法	24
2.6.3 解析初始引用方法	25
2.6.4 对象字符串化和字符串 对象化方法	26
2.6.5 ORB 启动	28

2.7 动态调用接口 (Dynamic Invoke Interface, DII)	28
2.7.1 客户端提出服务请求的方式	28
2.7.2 静态调用请求和动态 调用请求的区别	28
2.8 动态框架接口	29
2.9 接口仓库	30
2.10 对象适配器 (Object Adapter)	31
2.11 实现仓库	32
2.12 ORB 互操作	33
2.12.1 通用 ORB 间互操作协议	34
2.12.2 环境相关的 ORB 互操作协议	34
2.13 CORBA 基本服务	35
2.14 CORBA 工具	38
2.15 小结	39
第3章 OMG IDL 语法结构	40
3.1 OMG IDL 接口定义文件举例	40
3.2 词法规则	41
3.2.1 注释风格	41
3.2.2 标识符	41
3.2.3 字面量	42
3.3 基本数据类型	42
3.4 复合数据类型	43
3.4.1 结构类型	43
3.4.2 联合类型	45
3.4.3 枚举类型	45
3.4.4 序列类型	46
3.4.5 字符串类型	47
3.4.6 数组类型	47
3.5 模块	47
3.6 接口	48
3.7 属性和只读属性	49
3.8 方向性说明	49
3.9 方法	49
3.10 单向调用请求	50

3.11 接口继承.....	51	4.7.1 特征接口和方法接口 的定义	84
3.11.1 单继承接口定义方式.....	51	4.7.2 接口继承的映射	84
3.11.2 多继承接口定义方式.....	51	4.8 IDL 异常的映射	86
3.11.3 前置声明	52	4.9 标识符 typedef 定义数据类型 的映射	88
3.11.4 接口的跨模块继承.....	52	4.10 属性定义的映射	90
3.12 异常	53	4.11 CORBA 对象的映射	90
3.12.1 CORBA 异常定义.....	53	4.11.1 环境对象的映射	91
3.12.2 CORBA 系统异常.....	54	4.11.2 名一值对象的映射	91
3.12.3 CORBA 用户异常.....	56	4.11.3 名一值列表对象的映射	92
3.13 上下文.....	57	4.11.4 上下文对象的映射	93
3.14 小结	58	4.11.5 上下文列表对象	94
第 4 章 OMG IDL 与 Java 语言的映射关系 ...	59	4.11.6 请求对象	95
4.1 Helper 类	59	4.11.7 类型码	97
4.1.1 IDL 接口定义映射 生成 Helper 类规范.....	59	4.11.8 对象请求代理	98
4.1.2 IDL 复合数据类型映射 Helper 类	63	4.11.9 Any 数据类型	98
4.1.3 IDL 衍生数据类型映射 Helper 类	65	4.12 小结	100
4.2 Holder 类	67	第 5 章 idl2java 映射文件分析	101
4.2.1 IDL 基本数据类型映射 Holder 类代码	67	5.1 定义和映射 IDL 接口定义文件	101
4.2.2 IDL 定义的接口对象映射 Holder 类代码	68	5.2 接口对象的客户端存根文件	102
4.2.3 IDL 用户定义数据类型映射 Holder 类代码	69	5.3 服务对象端 POA 框架文件	103
4.3 IDL 常量的映射	71	5.4 接口对象定义文件	104
4.3.1 在 IDL 接口定义内部定义的 常量	71	5.5 接口对象映射的 Helper 类文件	104
4.3.2 在 IDL 接口定义外部 定义的常量	72	5.6 接口对象映射的 Holder 类文件	106
4.4 IDL 基本数据类型的映射	72	5.7 接口对象映射的方法类文件	107
4.5 复合数据类型的映射	74	5.8 接口对象的 POA 框架代理类	107
4.5.1 枚举类型的映射	74	5.9 小结	108
4.5.2 结构类型映射	77	第 6 章 VisiBroker 的基本内容	110
4.5.3 联合类型的映射	80	6.1 VisiBroker 中的 ORB 服务工具	110
4.5.4 序列类型的映射	83	6.2 基于 VisiBroker 开发分布式 应用系统过程	111
4.6 模块的映射	83	6.3 VisiBroker 的下载和安装	112
4.7 接口的映射	83	6.3.1 VisiBroker 的下载	112
		6.3.2 VisiBroker 的安装过程	112
		6.4 VisiBroker for Java4.5.1 安装目录的构成	117
		6.5 idl2java	118
		6.6 idl2ir	119

6.7	ir2idl.....	120	7.6.3	创建服务 POA.....	137
6.8	java2idl.....	120	7.6.4	服务实现对象实例化.....	137
6.9	java2iiop.....	121	7.6.5	激活服务实现对象.....	137
6.10	vbjc.....	122	7.6.6	激活根 POA	138
6.11	vbj	122	7.6.7	启动服务对象.....	138
6.11.1	vbj 运行客户端程序选项	122	7.6.8	服务对象应用程序 完整代码	138
6.11.2	vbj 运行服务对象端程序 选项	123	7.6.9	服务实现程序代码.....	139
6.12	Smart Agent	123	7.7	分布式应用的运行	140
6.13	osfind	124	7.8	分布式应用系统的部署	141
6.14	irep	125	7.9	小结.....	141
6.15	oad.....	125	第 8 章 可移植对象适配器	142	
6.16	oadutil	126	8.1	可移植对象适配器的含义	142
6.17	可视化对象管理工具	127	8.2	POA 的服务策略	143
6.17.1	定位服务管理	128	8.2.1	生命期策略	143
6.17.2	命名服务管理	129	8.2.2	对象标识惟一性策略.....	144
6.17.3	实现仓库管理	129	8.2.3	线程策略	144
6.17.4	接口仓库管理	130	8.2.4	服务实现对象的标识 赋值策略	144
6.17.5	服务对象管理	131	8.2.5	服务实现对象维护策略.....	144
6.18	小结	132	8.2.6	请求处理策略.....	144
第 7 章 基于 VisiBroker 开发分布式应用 ..	133		8.2.7	明确激活策略.....	144
7.1	基于 VisiBroker 创建分布式 应用程序过程.....	133	8.2.8	绑定支持策略.....	144
7.2	人力资源管理分布式 应用功能.....	134	8.3	获取对根 POA 的引用和创建 服务 POA	145
7.2.1	系统分析	134	8.3.1	获取根 POA 的引用.....	145
7.2.2	服务实现对象 UML 类图.....	134	8.3.2	创建服务 POA.....	145
7.3	IDL 定义服务对象功能	134	8.4	管理 POA 的状态.....	146
7.4	将 IDL 接口定义映射到 Java 语言 ..	135	8.5	激活服务对象.....	147
7.5	编写客户端应用.....	135	8.6	服务对象与服务管理对象	148
7.5.1	初始化对象请求代理	135	8.7	VisiBroker 的线程策略模型.....	149
7.5.2	定位并绑定服务对象	135	8.7.1	线程池 (Thread Pooling) 模型	149
7.5.3	调用服务实现对象方法.....	136	8.7.2	会话线程 (Thread-per -session) 模型	151
7.5.4	客户端应用程序完整代码....	136	8.8	小结.....	152
7.6	编写服务对象应用程序.....	137	第 9 章 基于动态调用接口和动态框架 接口创建分布式应用	153	
7.6.1	获取服务对象端对象 请求的代理初始化引用	137	9.1	创建基于动态调用接口的	
7.6.2	获取对根 POA 的引用.....	137			

客户端程序.....	153	11.4 命名服务支持工具.....	183
9.1.1 创建动态调用的客户端		11.5 基于命名服务客户端应用	
存根.....	153	程序设计.....	183
9.1.2 获取对服务实现对象引用.....	153	11.6 基于命名服务的服务对象	
9.1.3 创建请求对象.....	154	程序设计.....	184
9.1.4 配置方法调用参数.....	155	11.7 运行基于命名服务应用程序	
9.1.5 设置返回值类型.....	156	的选项.....	187
9.1.6 发出调用请求, 等待		11.8 基于命名服务分布式应用	
返回结果.....	156	程序的编译和运行.....	188
9.1.7 查询服务对象返回结果.....	157	11.9 小结.....	189
9.1.8 返回结果的解析.....	158	第 12 章 事件服务的应用	190
9.2 基于动态调用方式客户端		12.1 事件服务概述.....	190
应用程序完整代码.....	158	12.2 事件处理模型和事件对象接口.....	192
9.3 创建基于动态框架接口的		12.2.1 发送事件模型.....	192
服务实现对象程序.....	159	12.2.2 接收事件模型.....	194
9.4 基于动态调用方式的服务		12.3 事件管理对象.....	196
实现对象程序代码.....	162	12.4 事件通道.....	197
9.5 创建服务对象程序.....	164	12.5 创建事件服务应用程序.....	199
9.6 基于动态调用方式分布式		12.6 小结.....	203
应用程序的运行.....	165	第 13 章 接口仓库的应用	204
9.7 小结.....	166	13.1 接口仓库的概念.....	204
第 10 章 定位服务的应用	167	13.2 IDL 接口名称与接口定义标识.....	204
10.1 定位服务的概念.....	167	13.3 接口仓库中对象定义.....	205
10.2 获取定位服务代理实例.....	167	13.3.1 模块定义对象.....	205
10.3 对象实例信息查询方法.....	168	13.3.2 接口定义对象.....	205
10.4 对象接口定义信息查询完整代码..	170	13.3.3 方法定义对象.....	207
10.5 监听器类方法.....	171	13.3.4 属性定义对象.....	208
10.6 编写监听器句柄.....	172	13.3.5 常量定义对象.....	209
10.6.1 对象就绪状态消息		13.3.6 序列定义对象.....	209
发送方法.....	172	13.3.7 结构定义对象.....	210
10.6.2 对象关闭状态消息		13.3.8 字符串定义对象.....	210
发送方法.....	173	13.3.9 枚举定义对象.....	211
10.7 在对象定位代理中以监听器		13.3.10 异常定义对象.....	211
方式注册实现对象.....	174	13.4 接口仓库的创建和维护.....	213
10.8 小结.....	175	13.4.1 基于命令行方式的接口	
第 11 章 命名服务的应用	177	仓库管理.....	213
11.1 命名服务的概念.....	177	13.4.2 接口仓库对象定义.....	213
11.2 命名上下文.....	178	13.5 接口仓库应用程序设计.....	214
11.3 获取命名服务对象.....	181	13.6 小结.....	215

第 14 章 对象激活进程的应用	217
14.1 服务对象的自动激活机制	217
14.2 启动对象激活进程服务	217
14.3 对象激活进程中注册对象的管理 ..	217
14.3.1 注册实现对象	218
14.3.2 取消对象注册	218
14.3.3 查看对象激活进程的内容 ..	218
14.4 基于对象激活进程应用程序	
设计方法	219
14.4.1 对象激活进程对象的	
IDL 定义	219
14.4.2 实现对象信息描述结构	221
14.4.3 在对象激活进程中注册	
实现对象程序设计	222
14.5 小结	224
第 15 章 URL 命名服务的应用	225
15.1 URL 命名服务	225
15.2 URL 命名服务对象	225
15.3 基于 URL 命名服务的客户	
端程序设计	226
15.4 基于 URL 命名服务的服务	
对象程序	227
15.5 小结	229
第 16 章 基于 JBuilder5 和 VisiBroker	
开发分布式应用	230
16.1 JBuilder5 简介	230
16.2 JBuilder5 Enterprise 版的	
安装和许可证配置	231
16.2.1 JBuilder5 企业版	
的安装	231
16.2.2 JBuilder5 的许可证配置	232
16.3 JBuilder5 和 VisiBroker 的整合	234
16.3.1 配置 ORB 库	234
16.3.2 选择 ORB 产品	236
16.3.3 设置 VisiBroker 工具路径 ..	237
16.3.4 设置 VisiBroker 库集合	
路径	237
16.3.5 设置新建项目的默认设置	
.....	237
16.4 基于 JBuilder5 和 VisiBroker	
创建分布式应用的过程	238
16.5 创建 JBuilder5 工程	239
16.6 定义 IDL 接口	240
16.7 映射客户端存根和服务对象	
框架代码	241
16.8 编写客户端程序	242
16.8.1 创建客户端应用程序	242
16.8.2 设计客户端应用程序	
用户界面	243
16.8.3 编写客户端调用服务	
对象方法代码	245
16.9 编写服务对象和服务对象	
实现代码	246
16.9.1 创建服务对象程序	246
16.9.2 编写服务实现对象中	
商务方法代码	248
16.10 分布式应用的运行和部署	250
16.11 小结	251
第 17 章 模拟人力资源管理分布式	
应用实例	253
17.1 人力资源管理分布式应用	
系统分析	253
17.1.1 系统功能描述	253
17.1.2 分布式应用服务对象的 IDL	
接口定义	254
17.1.3 客户端应用 UML 类图描述	255
17.1.4 服务对象 UML 类图描述 ..	256
17.1.5 分布式应用配置图	257
17.2 服务对象应用程序设计	258
17.2.1 创建服务对象工程	258
17.2.2 编写 IDL 接口定义文件	258
17.2.3 创建服务对象程序	259
17.2.4 编写人力资源经理	
对象 HRManager 实现代码 ..	259
17.2.5 编写 PersonalInfo 对象实现	
代码	262
17.3 客户端应用程序设计	263

17.3.1	创建客户端应用工程.....	263	18.2.3	创建服务对象程序.....	277
17.3.2	编写 IDL 接口定义文件.....	263	18.2.4	编写 ATMServer 对象 实现代码	277
17.3.3	创建客户端应用	263	18.2.5	编写 ATMClient 对象 实现代码	279
17.3.4	创建客户端图形用户界面 设计	263	18.3	ATM 分布式应用系统客户端 程序设计.....	281
17.3.5	插入 CORBA 对象调用 接口	264	18.3.1	创建客户端工程.....	281
17.3.6	编写创建注册员工信息 请求方法	265	18.3.2	编写 IDL 接口定义文件.....	281
17.3.7	编写创建删除员工注册信息 请求方法	267	18.3.3	创建客户端应用.....	281
17.3.8	编写查询员工注册 信息方法	268	18.3.4	图形用户界面设计.....	281
17.4	分布式应用系统的部署和运行.....	269	18.3.5	插入 CORBA 对象 调用接口	282
17.4.1	编译客户端和服务对象端 应用程序	269	18.3.6	编写创建信用卡账户 方法代码	283
17.4.2	分布式应用系统的部署.....	269	18.3.7	编写利用 ATM 系统客户端 存款的方法代码.....	284
17.4.3	启动服务对象应用	269	18.3.8	编写利用 ATM 客户端从 指定账户取款方法代码.....	284
17.4.4	启动客户端应用	270	18.3.9	编写利用 ATM 客户端查询 账户余额方法的代码.....	285
17.5	小结	270	18.4	ATM 分布式应用系统的部署 和运行.....	286
第 18 章	模拟 ATM 机分布式应用实例	271	18.4.1	编译客户端和服务对象 应用程序	286
18.1	模拟 ATM 机的分布式应用 系统分析.....	271	18.4.2	分布式应用系统的部署.....	287
18.1.1	模拟 ATM 机分布式应用 系统功能描述	271	18.4.3	启动服务对象应用.....	287
18.1.2	ATM 机分布式应用系统 功能描述	271	18.4.4	启动客户端应用.....	287
18.1.3	分布式应用服务对象功能 IDL 描述.....	272	18.5	小结.....	287
18.1.4	客户端应用 UML 类图描述	273	第 19 章	COM 与 CORBA 的对比.....	289
18.1.5	服务对象端应用 UML 类图描述	274	19.1	COM 与 CORBA 的产生背景	289
18.1.6	分布式应用 UML 配置描述	275	19.1.1	COM 的产生背景.....	289
18.2	模拟 ATM 分布式应用系统服务 对象端应用程序设计.....	276	19.1.2	CORBA 的产生背景.....	290
18.2.1	创建服务对象工程	276	19.2	COM 与 CORBA 的共同点	290
18.2.2	编写 IDL 接口定义文件.....	276	19.2.1	位置无关性.....	291
			19.2.2	平台无关性.....	291
			19.2.3	编程语言无关性.....	291
			19.3	COM 与 CORBA 的区别	291
			19.3.1	在对象模型方面的区别.....	291

19.3.2 在对象的定位方面	292	19.4 小结.....	293
19.3.3 在对象的持久性管理方面..	292	附录 1 OMG IDL 保留字	294
19.3.4 在组件提供的服务方面.....	292	附录 2 CORBA 系统异常类型	295
19.3.5 在异常定义及抛出方 式方面	292	附录 3 接口仓库对象类型.....	297
19.3.6 在运行平台与组件开发 工具方面	293	附录 4 CORBA 对象定义	298
		附录 5 中英文对照.....	302
		附录 6 相关互联网资源.....	304

第 1 章 分布式应用模式概览

本章主要内容

- ◆ 分布式应用的基本特征
- ◆ B/S、C/S 体系结构
- ◆ 分布式多层体系结构
- ◆ 中间件的概念

1.1 客户机-服务器应用模式

在计算机的早期应用中，人们的应用方式主要集中在利用一台独立的计算机完成数据处理、存储、显示等工作。从早期的应用技术水平和需要计算机处理的实际业务角度来看，客户机-服务器应用方式满足了人们对计算机实现功能的基本要求，即利用一台计算机完成所有工作。

例如计算机在最初投入金融领域应用的时候，客户将货币存入银行后，银行的工作人员将客户的账户记录存储在计算机的硬盘或由磁带驱动器记录的磁带等外部设备中。一旦客户再次来到该银行，工作人员就可以从永久存储设备中恢复客户的账户记录，完成客户提出的业务、将账户记录进行更新并将更新后的账户记录在计算机磁盘或者磁带中。

1.1.1 客户机-服务器（C/S）体系结构

从完成客户的基本需求情况看，C/S 应用方式除了数据的安全性和客户的方便程度较差之外，是当时科技水平下最理想的应用方式。但是，随着科学技术水平的提高，运行一个应用程序涉及的对象越来越多、越来越复杂。另外，随着客户需求的变化、计算机技术水平的提高、银行客户的不断增多，这种单机应用的方式逐渐显示出它自身的缺陷。

首先，客户要求能够在该银行所属的任何一家办事处办理业务。随着 ATM 机的出现，客户统一要求能够以自助方式来完成货币的存储和提取等业务。为了满足人们的需求，特别是 20 世纪 70 年代专用网络技术的出现，计算机工程技术人员逐渐将在单机内存储的客户信息记录转移到银行中心数据库中，在中心数据库中集中存储和管理在该银行注册的所有客户的账户信息。银行办事处的计算机与银行中心计算机通过计算机专网进行连接。这样由办事处计算机与银行中心计算机构成的网络中，涉及到的客户账户信息的显示和银行业务逻辑的处理在办事处本地的计算机中完成，客户账户更新后的结果则需要通过网络存储在银行中心计算机中。这样，将原来在一台计算机中完成的工作分解到多台通过网络相互连接的计算机中，这就是最初的客户机-服务器（Client/Server，C/S）应用模式。客户机-服务器应用模式下计算机网络系统体系结构如图 1-1 所示。

C/S 体系结构是一种重要的计算机业务处理模式。首先，在具有大容量存储空间的计算服务器中将数据集中存储，由数据库服务器完成数据备份、恢复机制和安全性管理等工作，增加了数据资源的稳定性和信息存储的安全性，为实现数据资源的远程共享和访问奠定了坚实的数据资源基础。

通常，C/S 体系结构中的服务器需要完成的主要任务包括：

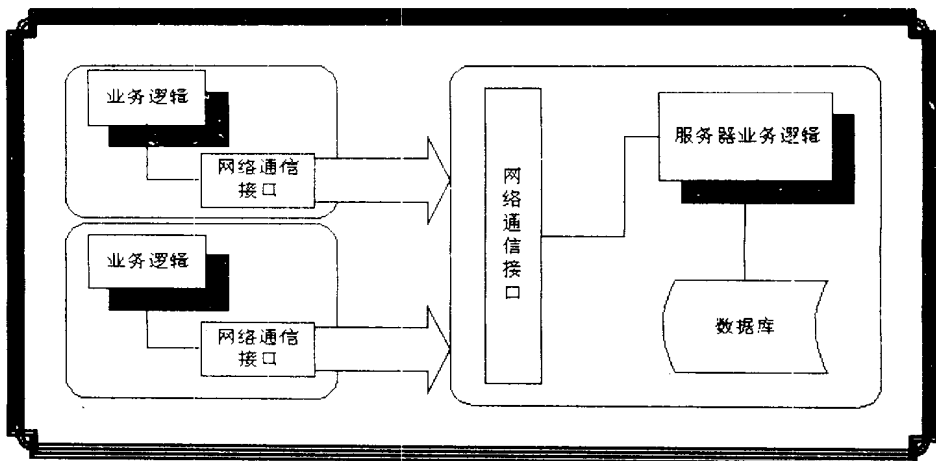


图 1-1 客户机—服务器应用模式系统结构示意图

- ◆ 为数据库提供安全服务；
- ◆ 数据库访问并发性的控制；
- ◆ 数据库的备份与恢复；
- ◆ 验证提交数据库的客户数据合法性。

相对而言，客户机需要完成的功能包括：

- ◆ 提供用户与服务器交互的用户界面；
- ◆ 向服务器提交用户请求并接收来自服务器的计算结果；
- ◆ 利用客户应用程序对服务器端的数据执行用户逻辑。

在 C/S 模式的分布式应用系统的客户计算机中有一个负责提交客户请求及解析服务器计算结果的通信代理模块。与服务器相对应，在客户机中也有一个相应的通信代理模块。该模块的主要作用是控制和调整服务器程序程序和客户应用程序之间的数据传输过程以及维持网络中计算机的数据联系。

以 C/S 模式构建的计算机网络系统在金融领域的应用为例：银行可以将所有办事处的计算机利用网络与银行中心的服务器连接起来，使之构成一个局域网。办事处的柜员机作为网络的客户端通过网络与中心服务器相联系，将创建的客户的个人账户记录存储在银行中心服务器中。这样，客户就可以利用该银行的任意一个柜员机来办理业务。因此，客户机—服务器分布式应用模式的引入，极大地促进了银行业务的开展。

C/S 体系结构模型曾经被大部分分布式应用系统采用，如银行储蓄系统、电子邮件系统（非 web 邮件）等。在 C/S 体系结构构建的分布式应用系统中，通过将处理业务合理地分配到客户计算机和服务器两端，在客户计算机中完成部分业务数据的计算、整理和终端内容的显示，在服务器中进行复杂的数据计算、数据格式合法性验证和生成业务数据的存储、备份等。这样，通过将同一个业务请求的处理过程分解，合理降低了各个计算机系统间的通讯开销，可以充分利用了客户机和服务器在硬件系统性能方面的优势。

1.1.2 浏览器-服务器 (B/S) 体系结构

随着近年来网络技术的发展,浏览器-服务器系统结构在分布式计算领域中逐渐得到推广应用。从系统实现原理角度来看,B/S结构实际上是C/S结构的一种变形。B/S模式下的客户端为浏览器(如HTTP客户端)、服务器被称为Web服务器(如HTTP服务器)。在以B/S模式构建的分布式计算体系结构中,客户端以浏览器的形式提供基本的图形用户界面(GUI)。用户通过浏览器创建用户逻辑并且将生成的客户请求发给WEB服务器。WEB服务器在对用户请求进行处理后,将结果返回客户端并由客户端在判断服务器处理结果到来后对浏览器页面进行刷新。这种应用结构目前被绝大部分企业信息系统所采用,其重要性日益突出而被独立称之为浏览器/服务器(Browser/Server,B/S)应用结构。

综合C/S、B/S体系结构的特点可以看出,在两层结构的分布式应用体系结构中,以具有大容量存储空间、高效数据处理性能的计算机为服务器,以具有用户界面和简单的数据处理能力的计算机为客户端,通过网络将客户机和服务器联系起来。图1-2为典型的客户机-服务器模式下C/S和B/S体系结构模型图。

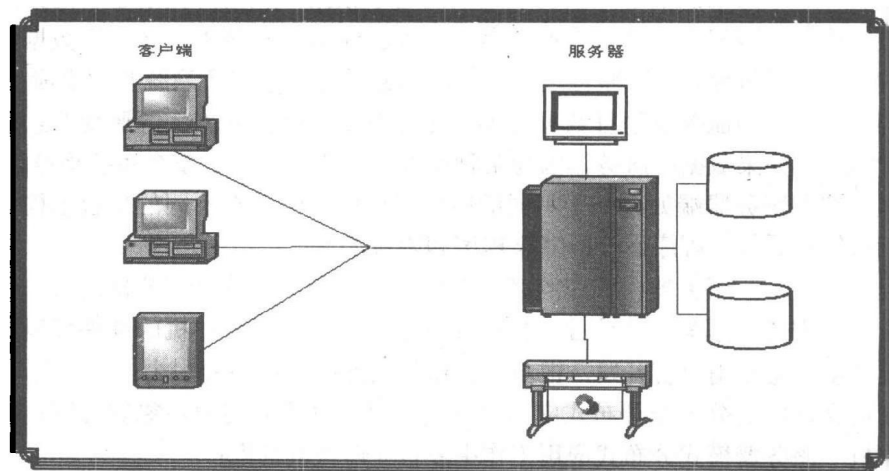


图 1-2 客户机-服务器模式体系结构图

Internet/Intranet 技术的普及和发展,推动了基于网络的分布式应用技术的发展。为了降低分布式应用系统客户端维护的复杂程度,网络系统应用中客户端“零”管理成了众多企业级应用系统的追求目标。在B/S结构中,客户端并没有与特定应用相关的应用程序,所有的用户界面的显示数据均通过WWW浏览器解析服务器端传来的数据实现。这种应用技术的发展成为推动分布式应用的重要里程碑。

1.1.3 C/S 和 B/S 体系结构的优势与不足

基于网络的C/S体系结构分布式计算系统与独立的计算应用相比具有如下优势:

- ◆ 增强了计算机间的连通性和协作性:所有客户端计算机对于记录在服务器中的信息都能够迅速地、无障碍地访问,从而增强了计算机间的通信能力并将计算提交到具有强大计算能力的服务器中完成;
- ◆ 系统性能、可扩展性和容错性显著提高:根据当前和未来系统的要求合理分配计

算机硬件资源。在新的硬件被添加到系统中或相关硬件被更新后（比如替换故障系统）不会涉及到分布式系统中其他的应用单元；

- ◆ 降低了系统构建成本：多个用户或应用程序共享那些文件服务器、高容量数据服务器、彩色打印机等昂贵的外围设备，显著降低系统的构成成本。另外，分立的服务功能单元有利于二进制级的代码重用，减小系统后续开发代价。

也许读者会认为这种应用系统模式就是最理想的应用方式。确实，在相当长的一段时间里，这种分布式应用方式曾经成为构建分布式应用系统的首选，直到计算机应用逐渐普及、计算机系统技术、软件技术、网络技术的发展以及已有分布式软件系统的待处理数据量激增等问题出现时，人们才逐渐认识到这种应用模式的缺陷和不足：

- ◆ 随着服务器访问次数和系统待处理数据量的激增，网络传输速度和数据服务器的处理能力成为决定系统效率的主要因素。由于客户端和服务端直接连接，服务器将消耗大部分系统资源用于处理与客户端的连接工作。那么，如果服务器在某一时间段内接受到大量的客户端数据请求，服务器有限的系统资源将被用于频繁应付客户端的连接请求、接收客户端的请求数据，从而降低了系统的数据处理能力，使得从客户端角度来看服务器不能及时响应客户请求。客户端数据请求堆积的直接后果将导致系统整体运行效率的大幅降低甚至整个系统全面崩溃；
- ◆ 客户端直接与服务器进行数据交换，服务器中存储的数据资源遭受恶意攻击的机会增大。如果数据库服务器因为某种原因停止工作，那么整个系统将停止运行；
- ◆ 客户端和服务端如果处于不同的应用平台中，则要在不同的平台上构筑分布式应用软件系统，程序设计的复杂程度和困难增大；
- ◆ 由于客户机—服务器模式中的客户端具有一定的数据计算和分析能力，使得整个系统的构建成本增大。这种结构的业务逻辑需要针对客户机的硬件环境和操作系统环境采用专用语言开发，很难再移植到其他的应用系统中去；
- ◆ 如何合理的划分一个分布式应用中的服务对象功能和客户对象的功能，是构建客户机—服务器模式分布式应用工作中要首先解决的问题；
- ◆ 客户端应用程序分发工作的烦琐程度令人难以接受。系统开发过程完毕，随之而来的程序分发工作中除了要为每台客户机安装客户端应用程序的可执行文件外，还要安装客户端程序运行所必须的系统支持文件、通信环境配置以及相关的软件配置。如果以后系统服务功能升级，则针对客户端应用程序的修改又不得不重复上述过程；
- ◆ 在网络环境下如何划分服务器功能和客户机功能并在服务器中为客户机提出的服务请求提供负载均衡成为又一个新课题。

正是由于上面列举的有关传统的客户机—服务器应用系统体系结构的缺陷和不足，推动了新型分布式应用模型的创建和发展，这就是三层及多层分布式应用模式。

1.2 分布式多层应用体系结构

读者可以想像：在一个拥有众多的“客户机”的分布式应用环境中，如果采用两层体系结构，客户机中除了提供软件系统所必须的图形用户界面外，还需要具有一部分数据计

算能力。这样,对于每个客户机中的数据计算单元,无论是硬件部分还是软件部分在整个应用系统中是重复配置的,系统中拥有的客户机数量越多,对资源的浪费越大。

另外,客户机和服务器需要对每一个应用分别开发相应的软件模块,使得构建的软件系统缺乏灵活性,系统升级和可移植难度增大。随着系统应用向 Internet 扩展,系统处理信息量迅速增加,使得专用的客户端形式已经无法满足多种功能的要求。因此,人们在原有的客户机—服务器分布式计算模式基础上,将客户机中进行用户业务处理和计算的模块分离出来,并且在原来的两层结构之间增加一个被称为“业务逻辑层”的软件单元,用于处理用户的业务逻辑、提供数据库服务器的负载均衡。这样,构成了分布式软件系统的三层或多层应用模式。

1.2.1 分布式多层体系结构

在新型的分布式应用系统体系结构中,软件系统的构建一般分为“用户层”、“业务逻辑层”和“数据层”,分别部署在客户机、中间层计算机和数据库服务器中。典型的三层应用体系结构如图 1-3 所示。

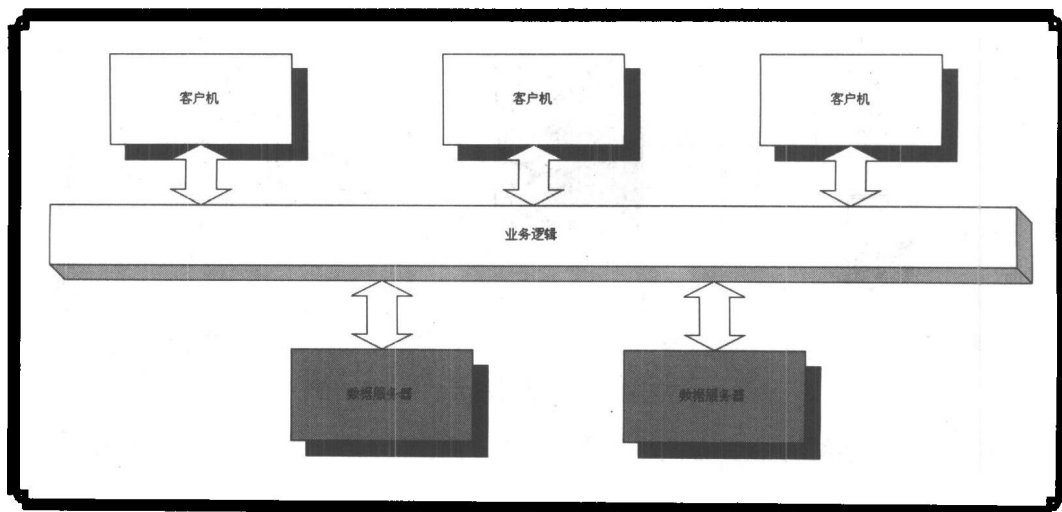


图 1-3 多层分布式应用体系结构示意图

在图 1-3 所示分布式应用模型中,用于显示客户信息的图形用户界面(GUI)驻留在客户端计算机中,原来的数据服务器端成为数据库服务器,专门负责系统应用数据的存储和维护,在原有两层体系结构的客户机和服务器之间增加了称为“业务逻辑层”的软件结构。

在分布式多层应用软件系统中,业务数据的计算和处理程序驻留在称为“业务逻辑层”的中间层,用于对客户端的请求进行响应、处理用户通过客户端提出的业务逻辑、调度多客户的请求、向数据库服务器提交计算结果、提出数据查询请求、处理输出结果直至将处理结果发送回客户端。

在多层分布式应用系统模型中,客户端程序既可以是一个浏览器,又可以是一个独立的应用程序,其主要功能是处理用户逻辑、表现系统处理结果,而不再进行较复杂的业务数据处理。因此,这种模式下客户端程序结构相对简单,进而被称之为“瘦客户机”形式。

在客户端以浏览器形式构建的分布式软件系统中,无论客户端运行的是 HTML 页面还是 Java Applet,均在运行时刻从“业务逻辑层”动态下载,使得客户端需要维护的程序代码量大大减少。如果系统功能发生变化或者需要对软件功能进行扩充,只需在业务层中增加相应的业务逻辑代码而无需对每个客户端进行更改就能够适应新扩充系统的要求。

在构建的分布式多层软件系统中,由于实现系统不同功能的软件模块运行在不同的硬件设备上,所以系统构建的灵活性很高,能够适应客户机数量的增加和业务处理负荷的变化。例如,在银行开展新业务时,只需要在业务层增加处理新业务的代码而无需对客户端通用的图形用户界面进行更改。因此,与两层应用模式相比分布式应用系统规模越大在系统维护方面的优势越明显。典型的三层结构模型计算机系统体系结构如图 1-4 所示。

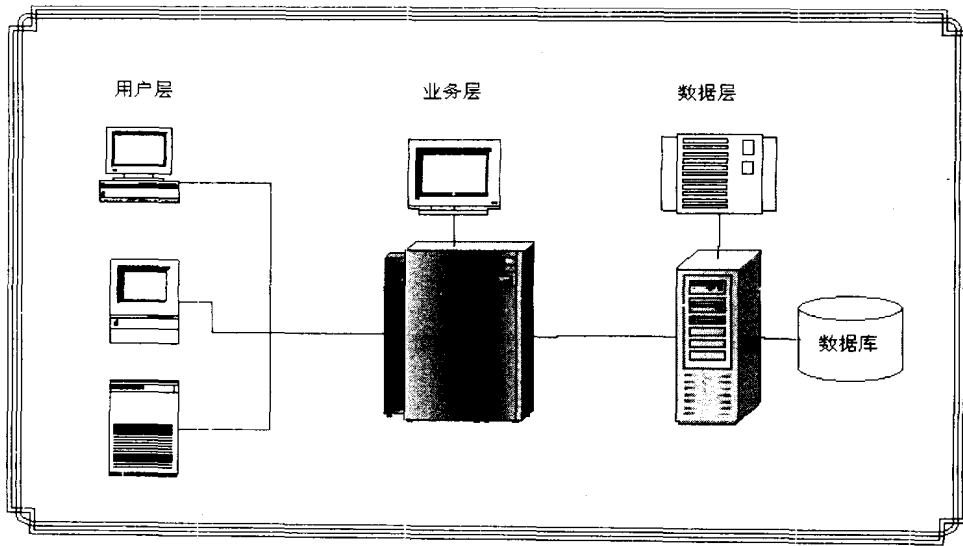


图 1-4 三层结构模型计算机系统体系结构图

在多层应用体系结构中系统的硬件设备构成方面,不同的逻辑单元对于构成计算机的计算能力、图形能力和数据存储能力的要求不同。例如:与用户直接交互的客户端只需具有较强的图形能力和简单的数据处理能力、承载业务层的计算机则要求能够进行大数据量、复杂的数据处理并且能够支持客户端多线程连接、数据库服务器则要求具备大容量的数据存储空间并能够进行数据“灾难”恢复等。因此,在硬件系统集成过程中可以有针对性地选择性能不同的计算机来承载相应的逻辑功能单元。

根据对分布式软件系统不同的逻辑功能单元层功能的划分,在实施系统集成过程中可以根据不同的逻辑功能来选择不同的计算机硬件设备,用较少的资源建立起具有很强伸缩性的分布式应用系统。

在多层分布式应用体系结构中,各逻辑部分之间的通信效率是影响整个系统效率的关键因素。如果各个逻辑层之间的通信效率低,即使分配给各层的硬件能力很强、各层之间的逻辑功能配置得很理想,整个应用系统的效率仍然不会很高。这就是在系统设计过程中必须慎重考虑逻辑层之间的通信方法、通信频度及数据量的重要原因。这和提高各层的独立性一样是多层分布式应用体系结构设计中需要解决的关键问题。