

J2EE电子商务

系统开发从入门到精通

——基于 Struts 和 Hibernate 技术实现

陈 松 编 著
冼 进 审



附光盘

- 应用程序源代码文件
- 10个视频教学文件
- 软件下载方法



清华大学出版社

TP312

2381D

2007

J2EE电子商务

系统开发从入门到精通

——基于 Struts 和 Hibernate 技术实现

陈 松 编 著
冼 进 审



附 光 盘

- 应用程序源代码文件
- 10个视频教学文件
- 软件下载方法

清华大学出版社

· 北京 ·

内 容 简 介

Struts 架构和 Hibernate 对象-关系映射工具相结合开发 Web Service 应用,是目前主流的 Web 开发模式。本书详细介绍了 Struts 和 Hibernate 基础理论,并大量地采用与实例相结合的方式,形成本书“理论+案例,理论案例相辅相承”的特色。

本书分为两大部分,第一部分从理论入手,介绍软件架构开发发展,Struts 和 Hibernate 原理及运行机制;第二部分穿插第一部分的理论知识,提供了一个完整实用的企业信息化管理网站平台的设计和实现,使开发者能够从多个角度深入了解 Struts 和 Hibernate 开发,并能快速上手独立开发 Web 应用系统。随本书附赠的光盘内容为本书所有范例程序源代码和程序效果图。

本书内容循序渐进,结构清晰,理论翔实,实用性强,既适合作为高等院校计算机专业教学参考用书,也可供有关技术培训及工程技术人员自学参考之用。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13501256678 13801310933

图书在版编目(CIP)数据

J2EE 电子商务系统开发从入门到精通:基于 Struts 和 Hibernate 技术实现/陈松编著. —北京:清华大学出版社,2007.5

ISBN 978-7-302-14082-5

I. J… II. 陈… III. ①电子商务—系统开发 ②JAVA 语言—程序设计 IV. ①F713.36 ②TP312

中国版本图书馆 CIP 数据核字(2006)第 129070 号

责任编辑:夏兆彦 林都佳

责任校对:张 剑

责任印制:李红英

出版发行:清华大学出版社 地 址:北京清华大学学研大厦 A 座

<http://www.tup.com.cn> 邮 编:100084

c-service@tup.tsinghua.edu.cn

社 总 机:010-62770175 邮购热线:010-62786544

投稿咨询:010-62772015 客户服务:010-62776969

印 刷 者:清华大学印刷厂

装 订 者:三河市新茂装订有限公司

经 销:全国新华书店

开 本:185×260 印 张:27 字 数:641 千字

附光盘 1 张

版 次:2007 年 5 月第 1 版 印 次:2007 年 5 月第 1 次印刷

印 数:1~5000

定 价:49.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。联系电话:(010)62770177 转 3103 产品编号:023518-01

在 Web Service 如火如荼的飞速发展过程中,涌现出众多的主流开发架构,Struts 架构便是其中闪亮的一员。Struts 架构是 Apache 基金会 Jakarta 项目中推出的一个子项目。正如其响亮的名字 Struts——支撑、支柱,Struts 架构在 Web Service 开发过程中起到了举足轻重的作用。Struts 架构为 Web Service 提供了一个开放源码的通用的开发模式,运用这个架构可以使程序开发人员把精力主要集中在业务逻辑的开发实现上。Struts 架构采用 MVC 设计模式,它是 MVC 设计模式的具体实现。遵循 MVC 设计模式,Struts 架构被分为低耦合度的三部分:控制器组件、视图组件和模型组件。控制器组件作为中间纽带将视图组件与模型组件联系在一起,视图组件完成与 Web 页面表单元素的绑定,模型组件完成具体的业务处理和数据库相关操作。采用 Struts 架构开发的 Web Service 应用,能极大地提高代码的重用性,缩短开发周期,提升 Web Service 开发效率。

在 Web Service 开发过程中,面对一个主流的关系型数据库,往往需要开发人员娴熟地掌握相关的数据库语言——SQL 语言,这不仅提高了 Web Service 开发的技术门槛,同时也有悖于现在软件开发面向对象思想中一个关键的技术即是数据持久化处理技术。为了解决这些问题,业界便提出了 Hibernate 这一优秀的基于 Java 语言的对象/关系数据库映射工具,同时它也成为了数据持久化问题的完美解决方案。Hibernate 做为 Web 应用与关系数据库之间的桥梁,它对 JDBC (Java Data Base Connectivity, Java 数据库连接)进行了轻量级的封装,并将基于面向对象模型开发的对象映射到基于 SQL 语言的关系型数据库的数据结构中。由于在 Struts 架构中只是提供了模型组件这一概念,没有为开发者提供具体的标准的开发类库,所以选用 Hibernate 映射工具生成模型组件即完成数据持久化处理成了必然的选择。

Struts 架构与 Hibernate 的结合方式实现具体的 Web Service 应用,已成了 Web Service 完美解决方案。

本书作者把多年开发 Struts 与 Hibernate 的经验编写成本书,通过深入浅出的基础理论讲解和丰富的案例分析的方式,引导读者在学习 Struts 架构和 Hibernate 基础理论的同时,将其运用于实际的 Web Service 应用开发当中。纵观全书,本书有如下特点:

- ❑ 本书介绍了 Struts 和 Hibernate 最基本的知识,读者只需具备基本的 Java 编程语言、JSP 页面开发和数据库设计知识即可。
- ❑ 采用基础理论知识结合实例分析的模式,清晰完整地展示了 Struts 和 Hibernate 的运行机制,这对读者的学习将有很大的帮助。
- ❑ 本书每章的标题就是对该章内容的高度概括,在接下来的内容中对其进行准确、详实的解释。
- ❑ 本书大量结合实际应用进行分析设计,对具体应用的开发都作了较详细的介绍和代码分析。

本书读者对象

本书适合从事 Web 应用开发的初中级程序员作为学习指南，也是高级程序员开发 Web 应用的有益补充。通过本书的学习，开发人员不仅可以掌握 Struts 和 Hibernate 开发的基础理论，同时也能积累大量的实际项目开发经验和技巧。本书可以作为高等院校计算机专业的教学参考用书，也可以作为 IT 培训机构的培训教材。

本书主要内容

本书分为基础理论与实际项目开发两个部分。

第一部分，包括第 1 章～第 3 章，在该部分中主要介绍相关的基础知识。第 1 章介绍软件开发采用架构技术的意义；第 2 章从 MVC 软件设计模式，引入 Struts 架构，并通过 Struts 组件等的介绍深入了解 Struts 架构的运行机制；第 3 章深入介绍 Hibernate 技术，以及利用软件工具生成 Struts 架构中的模型组件的完整过程。

第二部分，包括第 4 章～第 12 章，系统地介绍一个完整的信息管理网站——企业在线综合信息管理网站。第 4 章介绍整个网站的总体设计和实现效果图；第 5 章介绍系统登录模块的实现；第 6 章介绍企业人事信息管理模块的实现；第 7 章介绍企业新闻发布中心的实现；第 8 章介绍企业资产管理模块的实现；第 9 章介绍企业库存信息管理模块的实现；第 10 章介绍企业 BBS 系统的实现；第 11 章介绍企业电子投票系统的实现；第 12 章介绍企业电子阅览室系统的实现。结合这部分的实例开发全过程，提供大量的开发源代码与分析，读者将真正深入了解 Struts 和 Hibernate 在 Web Service 应用开发的运行机制。

本书由陈松负责编写，冼进审订。在编写过程中，感谢黄雷所给予的启发和鼓舞。参与本书编写工作的还有：邹素琼、冼进、赵秋云、赵继军、彭艺、曲辉辉、周章、蒋波、徐留旺、曹振宇、张婷、温凌霜、鲁得翠、蒋泽平、魏乐、韩翔、程小英、谭小丽、卢丽娟、李小琼、周宏、罗吉、许翔燕、陈春、张忠、方小马、黄姹英、周明、宋晶、邓勇等，在此一并表示感谢！

为方便读者阅读本书，随本书附赠的光盘内容为本书所有范例程序源代码和开发环境配置视频文件。由于作者水平所限，加之计算机技术发展迅速，书中错误和不妥之处在所难免，恳请读者批评指正。我们的联络方式：china_54@tom.com。

编 者

第 1 章 J2EE 概论	1	2.8 Struts 开发环境搭建	46
1.1 简单双层架构到复杂多层架构	1	2.8.1 Java 平台开发包下载	46
1.1.1 双层 (C/S) 软件架构设计	1	2.8.2 Struts 开发包下载及在线文档	46
1.1.2 传统三层软件设计架构	2	2.8.3 Tomcat 下载及安装	46
1.1.3 n 层软件架构及 Web 软件架构	3	2.8.4 Struts 与 IDE 的集成	48
1.2 J2EE 简介	4	2.9 小结	49
1.2.1 什么是 J2EE	4	第 3 章 Hibernate 基础	50
1.2.2 J2EE 体系架构概述	5	3.1 开发环境搭建	50
1.2.3 J2EE 主要技术	6	3.1.1 开发包下载	50
1.3 小结	7	3.1.2 Hibernate 与 IDE 集成	50
第 2 章 Struts 基础	8	3.2 体系结构	52
2.1 MVC 模式概述	8	3.2.1 Hibernate 架构	52
2.1.1 MVC 设计模式	8	3.2.2 重要 jar 文件介绍	53
2.1.2 JSP Model 1 与 JSP Model 2	9	3.2.3 Hibernate 与 Struts	53
2.2 Struts 的体系架构	10	3.3 配置文件	54
2.3 Struts 控制器组件类	12	3.3.1 基本配置	54
2.3.1 ActionServlet 类	13	3.3.2 连接池配置	56
2.3.2 RequestProcessor 类	16	3.3.3 数据源配置	57
2.3.3 Action 类	26	3.3.4 其他重要属性配置	57
2.4 Struts 视图组件类	30	3.4 事务	58
2.4.1 ActionForm 类	30	3.4.1 JDBC API 操作事务	59
2.4.2 ActionForward 类	32	3.4.2 Hibernate API 操作事务	60
2.4.3 ActionMapping 类	34	3.5 OR 映射	61
2.5 Struts 模型组件类	35	3.6 Middlegen 工具	63
2.6 Struts 的配置文件	38	3.7 hbm2java 工具	66
2.6.1 web.xml 文件	38	3.8 分页显示的实现	66
2.6.2 Struts-config.xml 文件	39	3.9 小结	67
2.7 Struts 标签库	43	第 4 章 企业在线信息系统网站概述与架构设计	68
2.7.1 Bean 标签库	43	4.1 企业在线综合信息管理网站概述	68
2.7.2 HTML 标签库	44		
2.7.3 Logic 标签库	45		

4.1.1	网站设计	69
4.1.2	编程实现	69
4.1.3	测试、发布与维护	69
4.2	模块划分	70
4.2.1	员工登录模块	70
4.2.2	人事信息管理	70
4.2.3	新闻发布中心	70
4.2.4	资产信息管理	70
4.2.5	库存信息管理	71
4.2.6	BBS 论坛系统	71
4.2.7	电子投票系统	71
4.2.8	电子阅览室系统	71
4.3	运行环境	71
4.4	界面效果	72
4.4.1	员工登录页面	72
4.4.2	网站首页	72
4.4.3	人事信息管理页面	72
4.4.4	新闻发布中心系统页面	73
4.4.5	企业资产信息在线管理 系统页面	73
4.4.6	企业在线库存管理页面	74
4.4.7	企业 BBS 页面及 BBS 管理 页面	74
4.4.8	电子投票系统投票页面	75
4.4.9	电子投票系统管理页面	75
4.4.10	电子阅览室系统浏览页面	76
4.4.11	电子阅览室系统管理页面	76
4.5	小结	77

第 5 章 员工登录系统 78

5.1	系统分析和设计	78
5.1.1	需求分析	78
5.1.2	登录页面	79
5.2	数据库设计及持久化	79
5.2.1	逻辑视图	79
5.2.2	数据库建表	79
5.2.3	Hibernate 配置文件基本 配置	80
5.2.4	封装 Hibernate 数据库操作 方法	81

5.2.5	数据持久化处理	84
5.2.6	Hibernate 配置文件	90
5.3	员工登录模块	91
5.3.1	实现效果	91
5.3.2	视图组件的实现	92
5.3.3	控制器组件的实现	93
5.3.4	struts-config.xml 配置	94
5.4	中文问题	95
5.5	小结	95

第 6 章 人事信息管理系统 97

6.1	系统分析和设计	97
6.1.1	流程设计	97
6.1.2	页面效果	97
6.2	数据库设计及持久化	98
6.2.1	逻辑视图	98
6.2.2	数据库建表	99
6.2.3	数据持久化处理	101
6.2.4	Hibernate 配置文件	113
6.3	个人信息修改模块	113
6.3.1	实现效果	113
6.3.2	视图组件	122
6.3.3	控制器组件实现	127
6.3.4	提交员工更改记录	128
6.3.5	struts-config.xml 文件配置	129
6.4	员工权限操作模块	130
6.4.1	实现效果	130
6.4.2	视图组件的实现	132
6.4.3	控制器组件实现	133
6.4.4	struts-config.xml 配置	133
6.5	小结	134

第 7 章 在线新闻发布系统 135

7.1	系统分析和设计	135
7.1.1	需求分析	135
7.1.2	流程设计	136
7.1.3	功能划分	137
7.2	数据库设计	138
7.2.1	逻辑视图	138

7.2.2	数据库建表	138
7.2.3	数据持久化处理	140
7.2.4	分页浏览的实现	153
7.2.5	Hibernate 配置	156
7.3	新闻信息管理模块	156
7.3.1	实现效果	156
7.3.2	视图组件设计及实现	167
7.3.3	控制器组件实现	170
7.3.4	struts-config.xml 配置	176
7.4	小结	177
第 8 章	在线资产管理系统	178
8.1	系统分析和设计	178
8.1.1	需求分析	178
8.1.2	流程设计及功能划分	178
8.2	数据库设计	179
8.2.1	逻辑视图	179
8.2.2	数据库建表	180
8.2.3	数据持久化处理	181
8.2.4	Hibernate 配置	203
8.3	在线资产管理模块	203
8.3.1	实现效果	203
8.3.2	视图组件设计及实现	213
8.3.3	控制器组件实现	218
8.3.4	struts-config.xml 配置	221
8.4	小结	223
第 9 章	库存信息管理	224
9.1	系统分析和设计	224
9.1.1	需求分析	224
9.1.2	功能划分	224
9.2	数据库设计	226
9.2.1	逻辑视图	226
9.2.2	数据库建表	226
9.2.3	数据持久化处理	228
9.2.4	Hibernate 配置文件	251
9.3	客户信息操作模块	251
9.3.1	实现效果	251
9.3.2	视图组件设计及实现	259

9.3.3	控制器组件实现	264
9.3.4	struts-config.xml 文件 配置	268
9.4	入库信息操作模块	269
9.4.1	实现效果	269
9.4.2	视图组件设计及实现	281
9.4.3	控制器组件实现	286
9.4.4	struts-config.xml 文件 配置	291
9.5	小结	292

第 10 章	企业 BBS 系统	293
10.1	系统分析和设计	293
10.1.1	需求分析	293
10.1.2	流程设计及功能划分	294
10.2	数据库设计	295
10.2.1	逻辑视图	295
10.2.2	数据库建表	296
10.2.3	数据持久化处理	297
10.2.4	Hibernate 配置	318
10.3	BBS 系统浏览模块	319
10.3.1	实现效果	319
10.3.2	视图组件设计及实现	326
10.3.3	控制器组件实现	327
10.3.4	struts-config.xml 配置	329
10.4	主题管理模块	330
10.4.1	实现效果	330
10.4.2	视图组件设计及实现	334
10.4.3	控制器组件实现	335
10.4.4	struts-config.xml 配置	340
10.5	小结	340

第 11 章	企业电子投票系统	341
11.1	系统分析和设计	341
11.1.1	需求分析	341
11.1.2	流程设计	342
11.1.3	功能划分	343
11.2	数据库设计	344
11.2.1	逻辑视图	344

11.2.2 数据库建表	344	12.1.2 流程设计	386
11.2.3 数据持久化处理	347	12.1.3 功能划分	387
11.2.4 Hibernate 配置	361	12.2 数据库设计	387
11.3 议题管理模块	361	12.2.1 逻辑视图	387
11.3.1 实现效果	361	12.2.2 数据库建表	388
11.3.2 视图组件设计及实现	367	12.2.3 数据持久化处理	390
11.3.3 控制器组件实现	369	12.2.4 Hibernate 配置	403
11.3.4 struts-config.xml 配置	371	12.3 文档管理模块	403
11.4 电子投票系统投票模块	373	12.3.1 实现效果	403
11.4.1 实现效果	373	12.3.2 视图组件设计及实现	407
11.4.2 视图组件设计及实现	381	12.3.3 控制器组件实现	409
11.4.3 控制器组件实现	382	12.3.4 struts-config.xml 配置	413
11.4.4 struts-config.xml 配置	383	12.4 文档浏览（记录）模块	414
11.5 小结	384	12.4.1 实现效果	414
第 12 章 企业电子阅览室系统	385	12.4.2 视图组件设计及实现	420
12.1 系统分析和设计	385	12.4.3 控制器组件实现	420
12.1.1 需求分析	385	12.4.4 struts-config.xml 配置	423
		12.5 小结	424

第 1 章 J2EE 概论

知识点:

- 软件层次架构的概念及发展
- 分布式系统简介
- J2EE 体系结构及技术概览

本章导读:

企业级应用程序架构历经了多年的发展演进,其设计思想已发生了巨大变化。回顾其发展历程,大型机和个人计算机盛行的早期,大多数企业级应用程序安装在单个机器上,并采用双层架构即客户端/服务器端架构。随着软件分层思想的引入以及 Internet 技术的迅猛发展,目前的企业级软件架构模型逐渐演变成三层架构甚至是 $n(n>3)$ 层架构,其中尤以基于 Web 的应用开发为典型代表。Java 2 平台企业版 (Java 2 Platform Enterprise Edition, J2EE) 作为美国 Sun 公司推出的一种全新的概念模型,即提供了三层架构及 Web 开发的企业级应用程序的高级支持。本章将讨论企业级软件架构模型的发展过程、J2EE 体系结构和技术以及基于 Web 应用开发的 J2EE 技术。

1.1 简单双层架构到复杂多层架构

1.1.1 双层 (C/S) 软件架构设计

企业级应用程序开发早期,我们通常会看到一个功能齐备、极少进行维护的应用软件安装在单一的大型机或者是 PC 机上。在这个“全能型”软件中,囊括了所有的用户交互 GUI (Graphic User Interface, 图形用户接口)、数据验证、多种多样的业务逻辑处理和数据库通信及操作等,数据则存储在集中式管理的服务器中,这种软件设计架构模型即被称为双层软件设计架构,也就是通常所说的 C/S 架构,如图 1-1 所示即双层软件设计架构模型。采用这种 C/S 架构的软件,有其显而易见的优点:表示逻辑和业务处理逻辑处于同一进程中,使应用程序易于开发;适用于用户数目(通常比较少)可以预估,且同时可以进行简单的资源分配的情况;多数数据计算处理工作集中于客户端 PC 机,服务器接收经过计算处理的数据,因此服务器承受的压力小,从而客户端也获得了相对较高的响应速度。

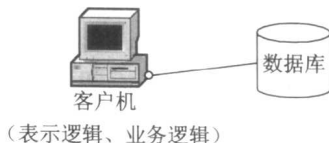


图 1-1 双层软件设计架构模型

然而, C/S 架构的软件自身的先天弊端无情地掩盖了其优点带来的好处,导致了一系列的问题,其主要表现在客户端过分臃肿、数据完整性不可控、不能适应复杂网络环

境、安全性得不到保障、功能可扩展性低、代码维护困难重重和服务器端资源浪费等方面。

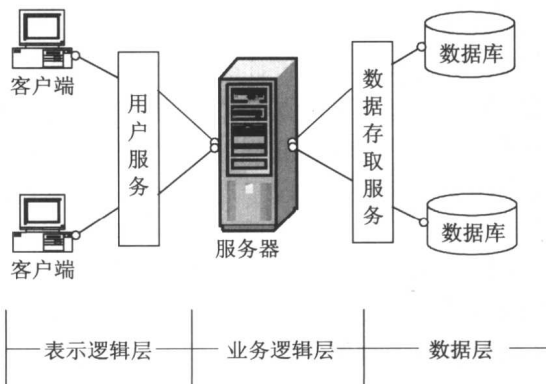
- ❑ **模块独立性低** 由于客户端嵌入过多的业务代码，客户端显示过分臃肿。模块之间偶合度过高而模块内聚度下降，导致团队并行开发工作效率降低，增加了测试和后期维护的工作量。代码维护难度随软件系统开发规模呈指数上升趋势，最终很可能造成难于维护或是干脆放弃维护的窘迫局面。
- ❑ **容易破坏数据的完整性和一致性** 客户端程序的错误或是缺陷、设计不合理的多线程控制机制、与实际不相符的事物（业务流程）完整性控制机制等都能够很轻易地破坏企业数据库数据的完整性和一致性。
- ❑ **无法胜任分布式应用系统的复杂性** 客户端软件系统只能运行在单一操作系统平台上，采用相同的网络传输协议等局限都大大影响了软件系统的可部署环境，这样就增加了企业在硬件及网络资源重复性投入的成本。
- ❑ **暴露严重的安全问题** 客户端采用与后台数据库服务器直接访问的方式，加上脆弱的网络安全环境，数据丢失的事故概率大。
- ❑ **可扩展性受到限制** 软件系统被安装在客户端，一旦软件发现 Bug、软件需要升级，对软件的修改将是企业的一笔不小的开支。出于成本上的考虑，系统进行二次开发的可能性不大。

双层软件架构诸如以上种种弊端，带来了如此多的局限性，严重地束缚着企业的发展。伴随着软件架构设计方法的不断演进，软件架构师提出了三层软件架构设计的思想，缓解并弥补了双层软件架构带来的局限性和不可集成性。

1.1.2 传统三层软件设计架构

传统的三层软件架构模型设计思想的提出克服了双层架构的某些局限性和缺陷，增强了软件架构的健壮性、可扩展性和可复用性。

相对于双层架构模型，传统三层架构模型的设计思想引入了许多的改进。其中尤为突出一点就是，架构模型首次将软件表示逻辑与业务逻辑分层构建：表示逻辑层位于架构中的第一层，即部署在客户端 PC 机上；业务逻辑层位于架构中的第二层，即部署在一个专用的服务器上；第三层即数据层，这一层基本沿用双层架构设计中第二层数据层，如图 1-2 所示即三层软件架构模型。模型的每个层次都各司其职，包含不同的代码逻辑：表示逻辑层包含表示代码，用户交互 GUI、数据验证；业务逻辑层包含业务规则处理代码；数据层包含数据处理代码和数据存储代码。应特别指出的是，三层架构模型仅表示各层次之间的逻辑关系，而不表示物理关系。如图 1-2 所示的客户端、服务器和数据库服务器三部分不一定是物理上分离



的三个部分。

- **表示逻辑层** 又常被称为“用户层”，用于向客户端用户提供 GUI 交互。它允许用户在显示系统中输入和编辑数据，同时系统提供数据验证功能。需指出的是，此处的数据验证只是数据格式等表示逻辑上的有效性检查，不包含数据在任何业务逻辑方面的有效性检查。表示逻辑层提供的数据验证是对用户输入数据的第一次过滤。此层也为用户提供数据结果的显示。
- **业务逻辑层** 用于执行业务流程和制订数据的业务规则。业务逻辑层作为表示逻辑层的下层，为表示逻辑层提供服务：接收表示逻辑层输入数据和应服务器的计算结果。业务逻辑层面向业务应用，而非针对任何特殊用户或是 GUI 交互的绑定。业务流程是指业务的相关专业算法、业务政策等，通常在开发过程中能够形成单独的代码模块，并且会作为公用模块方便开发人员调用以提高代码复用率；数据业务规则保证用户在表示逻辑层输入的数据具备业务合法性以及完整性，同时作为数据层的上层，为数据层数据存储传送最终有效的数据。
- **数据层** 主要包括数据存取服务。它封装了数据存取的底层操作，即对某个数据库管理系统（或是文件存储系统、以其他形式存在的存储系统）进行的存储、查询、删除和更新等数据库操作。

采用三层软件设计架构后，软件系统在可扩展性和可复用性方面得到极大提高，在资源分配策略设计合理运用的同时，软件的性能指标也得到提升，系统的安全性和易管理性也得到改善。虽然三层软件架构模型缓解甚至消除了双层架构模型的某些缺陷，但三层架构模型也存在某些缺陷：存在大量系统级编程的复杂性、软件缺乏高可移植性、各软件系统的不兼容性以及与 Web 不兼容，和双层架构一样不能很好地胜任异构的网络环境，这样就对软件设计架构又提出了新的要求。

1.1.3 n 层软件架构及 Web 软件架构

随着 Internet 和 intranet/extranet 的快速增长以及 PC 机普及率的惊人提升采用传统三层架构模型的软件存在的诸多缺陷更加突现出来且不堪重负，已不能适应现代网络技术高速的发展节奏。企业级应用正朝着多层架构的设计思想转变，涉及范围更加广泛的、基于复杂结构的分布式应用正在 Web 环境中悄然兴起。网络技术的日新月异也衍生出许许多多新的软件架构模型，带来了软件发展的春天。Web 技术的引入和推广改变了软件发展的窘境。Web 应用能提供支持所有类型内容连接的信息发布，容易为最终用户存取。Web 的流行无所不在，使许多传统的信息和数据库系统正在被移植到互联网上，同时也促成了电子商务的迅速增长，早已超过了国界。

如图 1-3 所示为 Web 软件设计架构模型。

在这个架构模型中，客户端负责用户的输入和结果的显示；Web 服务层提供 Java Server Page (JSP) 或者 Servlet 等的开发、部署和运行，为客户端准备前端显示页面代码；应用服务层实现企业业务逻辑和业务实体，并提供开发、部署和运行环境，同时该层也提供对分布式网络环境的支持，使得 Web 软件架构具备跨操作系统平台和跨网络传输协议等优点；数据层封装了数据库的底层操作，包括添加、查询、修改和删除等。

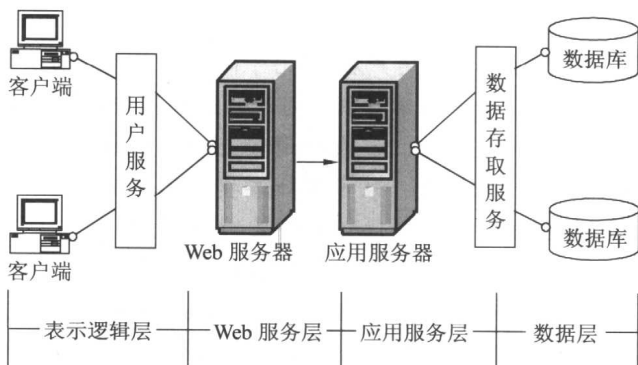


图 1-3 Web 软件设计架构模型

目前主流的多层架构包括了两种：微软公司的 Dot Net 和 Sun 公司的 J2EE。其中 J2EE 以其著名的跨平台等特性更加符合大型企业级应用系统架构解决方案的需求，也正在成为工业标准。J2EE（Java 2 Platform Enterprise Edition，Java 2 平台企业版）提供了三层架构、多层架构以及 Web 开发的企业级应用程序的高级支持。在 J2EE 平台下，流行的架构有 Struts、Spring 架构等，本书将以 Struts 架构为重点进行深入分析。

1.2 J2EE 简介

1.2.1 什么是 J2EE

Sun Microsystems 公司在 1996 年推出了一种新的纯面向对象的编程语言，命名为 Java。Java 技术已成为 20 世纪软件业界出现的最重要的技术之一，这种与平台无关的编程语言的出现，立刻掀起了软件工业的一场革命，如今已是互联网上最为流行的编程语言。Java 是一种通过解释方式来执行的语言，其特点为：简单、面向对象、不依赖于机器的结构，具有可移植性、鲁棒性、安全性，并且提供了并发的机制以及与现代网络技术的完美结合。目前，按照 Sun Microsystems 官方网站（<http://java.sun.com>）公布的标准，根据不同的应用领域将 Java 语言被划分为三大平台：

- ❑ **Java 2 Platform Micro Edition** 简称 J2ME，Java 2 平台微型版。用于开发掌上电脑、手机等移动通信设备上使用的嵌入式系统。
- ❑ **Java 2 Platform Standard Edition** 简称 J2SE，Java 2 平台标准版。主要用于开发一般台式机应用程序。
- ❑ **Java 2 Platform Enterprise Edition** 简称 J2EE，Java 2 平台企业版。主要用于快速设计、开发、部署和管理企业级的软件系统。

J2EE 是使用 Java 进行企业开发的一套扩展规范，它提供基于组件设计、开发、部署和管理企业应用的解决方案。J2EE 规范必须基于 J2SE 平台，它不仅巩固了标准版中的许多优点，例如“一次编译、随处运行”的跨平台特性、方便高效的数据库提供无缝连接的技术 JDBC（Java Data Base Connectivity，Java 数据库连接）、能够在 Internet 应用中确保数据完整性的安全模式以及支持多线程技术等，同时还针对 Servlet/JSP、EJB、

JMS、JTS/JTA、JCAT 和 XML 等技术提供了全面的支持。J2EE 已建立了一套能够使企业级应用开发者缩短软件产品开发周期的标准架构体系。

事实上, J2EE 已成为企业级开发的工业标准和主流平台。J2EE 为建立具有高可用性、高可扩充性、易维护性的企业级系统提供了完美的机制和架构模型。在 J2EE 提供的统一开发平台下, 可以大大降低开发多层应用的成本和复杂性, 对于需要获得高可用性、高安全性以及高可扩充性软件服务而又缺乏资金的企业来说, J2EE 的中间层集成框架完全可以胜任并满足其要求。同时, J2EE 自身具备的完备的目录服务支持、高度的安全机制、高效的性能展现等众多优势技术, 为企业平滑集成已有应用系统发挥了巨大作用。

基于 J2EE 的企业级应用系统, 不受任何特定操作系统、网络环境、硬件平台的束缚。应用系统可开发并部署在异构的环境中, 并能确保系统的可移植性。因此, 只要符合 J2EE 规范的系统开发一次即可部署到各种操作系统平台和网络平台上。J2EE 作为长期的性能稳定的企业级应用系统平台, 能保障 7×24 小时的不间断服务, 以满足企业客户、合作伙伴的需要, 是企业理想的运行平台也是 Internet 全球化发展的客观必然要求。J2EE 高伸缩性的特点在如今“按需应变”(IBM 公司 2002 年底提出) 电子商务战略大行其道的时代, 做出了卓越贡献。J2EE 采用负载平衡策略, 使系统软硬件资源利用率及性能表现最大化, 消除了系统中的瓶颈。允许多台服务器集成部署, 实现可高度伸缩的系统, 在保存企业现有 IT 资产减少重复性投入的同时也满足了未来商业应用的需要。

1.2.2 J2EE 体系架构概述

J2EE 使用多层的分布式架构模型, 模型由组件组成, 应用程序的逻辑根据功能划分被封装在各组件中。由于是采用分布式架构, J2EE 应用中的大量组件部署在不同的机器中, 通常这些机器具有不同的物理地址。J2EE 架构一般分为 4 层: 客户层、Web 应用层、企业组件层(业务层)、企业信息系统层。如图 1-4 所示描述了 J2EE 架构中典型的多层应用架构模型。

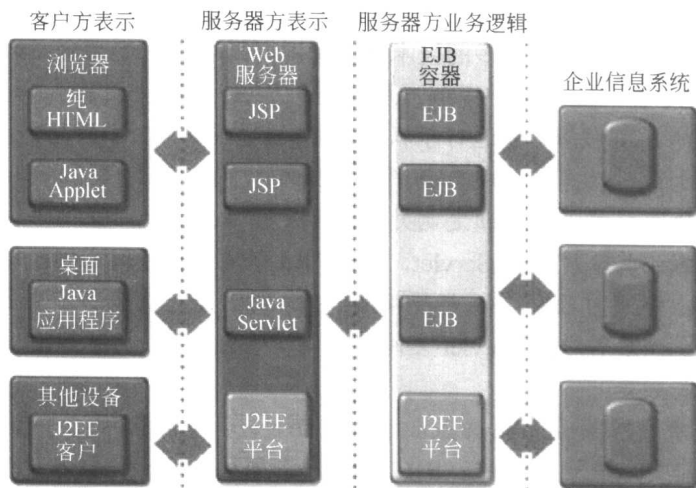


图 1-4 J2EE 架构的多层应用模型

一个 J2EE 客户层软件可以是基于 Web 的，也可以是基于 Application（应用）的。Web 客户端由 Web 应用层的 Web 组件动态生成 Web 页面，主要是 HTML 文件和 XML 文件。客户端用户在使用 J2EE 服务时，无需安装多余软件，只要计算机上装有一个 Web 浏览器即可，如 MS Internet Explorer、NetScape、Mozilla Firefox，等等。基于应用程序的客户端，可以为用户展示比 Web 客户端更为丰富的人性化的图形界面，如 Java AWT/Swing 开发的图形用户界面。Web 客户端通过 Web 应用层中的 JSP/Servlet 进行通信，而应用程序客户端则可以跳过 Web 组件层直接访问运行在企业组件层的 Enterprise JavaBeans。

Web 应用组件层提供的服务主要依赖于 JSP 页面或是 Servlet。JSP 和 Servlet 都是 J2EE 应用技术中的重要组成部分。JSP 页面由 HTML 文件、Java 程序片段和 JSP 标签构成，以 Servlet 的方式执行。Servlet 是 Web 应用开发的基础。它完全用 Java 语言开发，是 Java 平台上的 CGI 技术。它运行于服务器端的 J2EE 容器中，具有很高的效率，能够动态地加载，动态地生成 Web 页面。Servlet 擅长于流程控制和事务处理，提供的扩展服务功能可为企业定制灵活的开发策略。Web 应用层主流的服务器有：Tomcat、Resin 等。

企业组件层中主要是负责处理企业的业务流，并且将相关的业务逻辑封装在 Enterprise JavaBeans 中。这一层提供了 EJB 的开发、部署和运行时管理环境。每一个 EJB 都是一个可重用的组件。EJB 架构定义了 3 种 Enterprise JavaBean 类型：会话 Bean、实体 Bean 和消息驱动 Bean。其中，会话 Bean 又分为有状态会话 Bean 和无状态会话 Bean。实体 Bean 可以进行自行状态管理，也可以交由 EJB 容器进行管理，两种方式都为了完成实体 Bean 的持久性。消息驱动 Bean 始终处于无状态。三种类型的 Enterprise JavaBean 有着不同的生命周期。企业组件层采用的主流服务器有：Weblogic、JBoss、Sun Application Server、WebSphere，等等。

企业信息系统层的特点是有数据库系统的支持。企业信息系统软件主要包括企业内传统信息系统如财务系统、CRM 系统、数据库系统和 ERP 等。

1.2.3 J2EE 主要技术

在 J2EE 平台中，包括了众多的技术，以便对开发者提供更为全面的支持。

- **Servlet 技术** Servlet 是 Web 应用开发的基础。它完全用 Java 语言开发，是 Java 平台上的 CGI 技术。它运行于服务器端的 J2EE 容器中，具有很高的效率，能够动态地加载，动态地生成 Web 页面。Servlet 擅长于流程控制和事务处理，提供的扩展服务功能可为企业定制灵活的开发策略。
- **JSP 技术** 其本质仍是 Servlet，只是两者创建方式不同。JSP 文件由 HTML 文件、Java 程序片段和 JSP 标签构成。JSP 将首先通过分析器生成相应的 Servlet 源文件，然后再对此 Servlet 文件进行编译运行。JSP 与 Servlet 一样，具有相当高的运行效率。
- **EJB 技术** Enterprise Java Bean 简称 EJB，它运行在 EJB 服务器上，是由客户端调用的非可视化远程对象。EJB 拥有一系列协议，允许自身被远程访问或在特定服务器上安装或部署。允许将服务安全性、事务行为、并发性（支持多个客户端同时访问的情况）和持久性（其状态的生命周期）的主要方面授权给

EJB 服务器上其所在的容器。EJB 提供的事务机制非常周全，但事务机制带来的缺点是性能的降低。EJB 支持分部式应用系统。在 EJB 中定义了三种不同类别的 Enterprise Bean：会话 Bean、实体 Bean、消息驱动 Bean。EJB 的完整执行过程为：客户端通过 JNDI 检索 Home 对象的引用，JNDI 返回 Home 对象的引用，请求创建一个新的 EJB 对象，创建 EJB 对象，返回 EJB 对象，调用相关业务逻辑方法，调用 Enterprise Bean。一般 J2EE 应用服务器都是 EJB 服务器，比如：Weblogic、Jboss、Sun Application Server、WebSphere 等。

- **JMS 技术** Java Message Service, Java 消息服务。JMS 是 J2EE 平台的重要组成部分。它是一组标准 Java 应用接口，可以在各种企业通信系统间使用，提供创建、发送、接收、读取消息等服务。为确保 Java 应用能够和各种消息中间件进行异步通信，JMS 定义了一组公共的应用程序接口和相应语法来满足开发需求。通过使用统一的 JMS 接口集，开发人员可以无须掌握消息产品的使用方法，就来编写和操纵各种消息中间件。JMS 支持点对点、发布/订阅式的异步消息通信机制，并能够最大限度地提升消息应用的可移植性。
- **JNDI 技术** Java Naming and Directory Interface，简称 JNDI。J2EE 应用程序组件通常分布在不同物理地址的机器上，所以需要一种机制以方便客户端使用者查找和引用 J2EE 组件及企业系统资源。在 J2EE 体系中，使用 JNDI 技术定位各种对象，包括 EJB 对象、数据库驱动对象、JDBC 数据源对象及消息连接对象等。JNDI 函数库为应用程序提供了一个统一的接口来完成规范定义的目录操作，如通过对象属性来查找和定位该对象。

1.3 小结

本章首先概要地讨论了各种企业应用系统架构，从简单双层架构模型、三层架构模型、多层架构模型和 Web 开发应用，到 J2EE 架构，结合图例描述了它们的优缺点，并引出本书的讨论重点：Struts 架构。

1.2 节中，重点介绍了 Java 的三种平台分类：J2ME、J2SE 和 J2EE。最后结合图例，介绍 J2EE 平台经典体系架构中的四个层次及其作用，以及 J2EE 发展过程中的一些主要技术如 Servlet、EJB 等。

第2章 Struts 基础

知识点:

- 介绍 MVC 设计模式
- 介绍 Struts 体系结构
- 讨论 Struts 三大核心组件
- 介绍 Struts 标签库

本章导读:

在早期企业级 Web 应用的开发过程中, 由于没有一个通用的开发架构标准, 开发人员不得不将大量的精力耗费在整个应用的架构设计上, 众多的 Web 应用开发衍生出众多的开发架构, 不同的架构间并不能完全的兼容, 这使得 Web 开发过程中在代码移植、程序可扩展性和应用后期维护等方面效率大打折扣。Struts 架构的出现为开发人员提供了一个现成的标准的 Web 开发架构, 开发人员可以集中精力解决实际的业务问题。

如今, Struts 架构已逐渐成为一个安全可靠、成熟的 Web 开发架构业界标准。开发大型复杂的企业级 Web 应用系统, Struts 架构将是开发人员的首选。

本章将从软件设计模式 MVC 引出 Struts 架构的概念。接着, 介绍 Struts 架构的体系结构。然后, 着重讨论 Struts 架构中的核心组件。最后, 简要介绍 Struts 提供的标签库。

2.1 MVC 模式概述

MVC 设计模式是 Model-View-Controller 模式的缩写, 它代表了三层模型: Model(模型)、View(视图)和 Controller(控制器)。20 世纪 80 年代, Xerox PARC 第一次提出了 MVC 软件设计模式, 至今已被广泛应用于软件架构设计, 是 Sun 公司 J2EE 采用的关键模式之一。MVC 设计模式为软件开发发挥着极大的作用, 并推动软件开发设计不断向前发展。

2.1.1 MVC 设计模式

基于 MVC 设计模式的程序, 通常按不同功能被划分为相对独立但可以有机结合的三个部分:

- **Model(模型)** 业务逻辑层。实现系统信息、状态记录功能, 对业务逻辑的封装。
- **View(视图)** 表示逻辑层。通常就是指 GUI, 实现可视化界面供用户输入数据, 并提供结果展示功能。
- **Controller(控制器)** 控制层。负责协调模型与视图即业务逻辑层与表示逻辑