编 号

本科生毕业设计（论文）

题目：基于 JavaEE 架构网上书店的设计与实现

信息工程 学 院 计算机科学与技术 专 业

学　　号 0304050214

学生姓名 徐　顺

指导教师 蒋　敏　副教授

二〇〇九年六月

摘 要

近年来，随着信息的全球化和国际互联网的普及化，商业的运行模式和人们的消费观念也随之改变，电子商务已经成为一种时尚.网上书店的出现使人们足不出户就可以买到自己需要的图书，从而打破了传统书店的经营模式。

本文设计的网上书店系统在实现一般网上销售图书的功能的基础上突出了图书的信息服务，例如，提供方便的图书查询、提供最新引进的图书信息、提供推荐图书等服务。

Struts2是一个MVC模式的框架，它将业务代码与视图代码分离，有效地优化了系统结构，提高了系统的扩展性，降低系统复杂度与维护难度。Hibernate是一个对象关系数据库映射工具，通过Hibernate的使用，能够很好地解决面向对象编程语言与关系数据库一起使用可能带来的种种麻烦，使得程序设计人员可以减少对关系数据模型的依赖，致力于业务逻辑的开发。Spring框架的使用将JavaEE层次结构中的业务层分离为业务逻辑层和数据持久层，这样业务逻辑便交给Spring处理，而数据访问则交给Hibernate处理，使得层次结构更加清晰，便于系统的维护和扩展。

本系统的主要特点是:操作简便，平台适应性广，在网站上的运行速度较快。

关键词：Struts2；Spring；Hibernate；网上书店

ABSTRACT

In recent years， along with the information globalization and the Internet universalization, the commercial operation model and the people’s thought about consuming have changed a lot, and the electronic commerce has become a kind of fashion.The appearance of online bookstore let people get what they need without walkingout of the house，so that it breaks the mode of traditional bookstores.

So the system in the paper emphasizes the information service of books besides realizing the function of selling books on-line, for example, offers the convenient book searching, offers the newest book information, offers the recommended books and so on.

Struts2 is a framework based on MVC model, which separates the business code and the view code, optimizes the system structure effectively, enhances expansibility of complexity system, and reduces complexity and maintainability of the system. Hibenrate is an object-relational mapping tool,via using Hibernate,can resolve the problems. produced by using object-oriented language and relational database. Spring separates the business tier into business logic tier and data persistence tier in JavaEE hiberarchy, then the business logic is handled by Spring,and the data access is handled by Hibenrate,which make the hiberarchy clear,in favor of maintainability and expansibility of the system.

The main character of the website is that convenient cooperating, extensive adaption of platform, and has a fast running speed.

Keywords: Struts2; Spring; Hibernate; online bookstore;

目 录

第 1 章 绪论

1.1 课题背景

图书是一种在整个社会生活中都很普及的精神消费品，在整个商品市场上占据不同于其它商品的特殊地位。基于 Java EE 的网上书店系统正是针对图书这样一类特殊的商品而建立起来的一个电子商务系统。如何迎合图书这种商品的特殊性，如何降低销售成本以及便利读者购书，便自然而然的成为这个系统设计和实现所追求的目标。

近些年来，随着网络通信技术的不断发展和社会信息化建设水平的不断提高，电子商务系统在社会生活和经济生活中得到了越来越广泛的应用。电子商务大大改变了企业的经营方式，规范了内部流程和交易手续，减少了交易的中间环节，降低了企业的经营成本;并使经营活动不再受地域和时间的限制，方便了客户，密切了企业和客户的关系。因此越来越多的企业采纳电子商务作为交易模式。随着互联网上的电子商务网站大量出现，电子商务作为一种交易形式已经在社会经济生活中占据一定的地位。另一方面，互联网技术的发展在推动电子商务进步的同时，也使电子商务系统的构建实现技术面临新的挑战。作为网络信息技术前沿的 Java，已经是软件界的一个热门话题，它提供的跨平台性、网络和数据库支持为 Web 应用系统的开发提供了新的途径。

21 世纪以来，人类经济高速发展，人们的生活发生了日新月异的变化，特别是计算机的应用及普及到经济和社会生活的各个领域。为了让消费者网的购物过程变得简单、方便、安全、快捷，网上商城购物成了一新型而热门的购物方式，开发该系统的好处有：一是现在的电脑普及率越来越高了，物流业发展迅速，这就造就了网上销售成为新兴而热门的行业。二是网上销售不受时间限制，只要将产品信息放在网上，就可以 24 小时营业了。三是费用低，房租，水电，装修，员工薪水统统不用考虑，只需一个办公室和存储商品的仓库就可运行了。四是不受区域限制，只要是上网的用户都可能成为顾客。网上销售的好处远不止这些。

因此，网上商城购物系统是一种具有交互功能的商业信息系统，它在网络上建立一个虚拟的购物商城，使购物过程变得轻松、快捷、方便。

1.2 课题意义

技术的进步对传统书店上网解决方案提出更严格的要求和挑战。为了保护传统书店的投资，书店上网解决方案应切合传统书店实际的需求和发展的趋向，使投入回报和管理效益最大化，传统书店在实施上网方案的之前，必须对一系列问题进行科学的论证，如书店上网的需求分析、书店上网总体规划、网上书店系统的功能和实施方案、网上书店的传播与推广、运行网上书店系统的软件和硬件配置、网上书店的管理系统和管理方法等等。网上书店具体实施的质素直接影响传统书店在 Internet 的实际效果和经济效益，这不仅是技术问题，同时也涉及到管理的因素。

通过构建一个销售图书的电子商务网站，利用日益繁荣的互联网，为传统的书店打造一个新的销售平台。

综上所述，网上书店已经成为互联网时代传统书店的必由之路。

1.2 国内外研究的现状及本系统的特色

网上购物商城系统实际上是基于企业与客户模式的电子商务，该模式在近些年来都有比较成熟的应用，世界上比较知名的企业如戴尔公司采用这种B-C模式让客户能够通过网络订购公司的产品，国内的一些公司如当当购物网等也通过B-C模式建立了一种购物平台。实际上，网上购物商城的模式基本相同，只是各个具体的商家(客户)的具体应用有所不同，因此开发该系统要求能够充分满足客户的需要。本系统主要为利用网络这个平台，建立网上销售图书系统。利用流行的Java框架，良好的层次架构，实现网上书店系统。

1.3 论文的研究工作内容

系统的架构主要由基于Java EE技术来完成，本课题利用了动态网站的常用技术以及MVC，DAO等常用设计模式，研究了基于MVC的Struts2框架的设计，Spring的IoC功能维护业务对象之间的关联及声明式事务管理，基于Hibernate的持久层解决方案，AJAX技术的运用，提高用户体验。并根据软件工程流程，对网上书店系统的分析与设计，实现了销售图书的购物网站。

采用Struts2、Spring和Hibernate相结合的多层架构充分发挥三者的优点，为我们研究的电子商务系统提供了便利条件。

1.4 论文的组织结构

第1章 绪论。主要介绍课题背景及研究意义。

第2章 Web应用体系结构和Java EE技术。介绍网络环境的的两种基本模式、JavaEE技术和流行框架等。

第3章 网上书店系统的设计。主要介绍系统需求分析、模块设计和数据库设计等。

第4章 网上书店系统现。包括该系统的实现原理及部分功能模块的编写代码及其实现的界面截图。

第5章 总结。对全文做总结与展望。

第 2 章 Web 应用体系结构和 Java EE 技术

2.1 Web 应用体系结构

2.1.1 C/S 体系结构

C/S模式是一种两层结构的系统，第一层是在客户机上处理表示逻辑与业务逻辑，第二层则是通过网络运行的数据库等服务器系统。结构如图2-1所示。

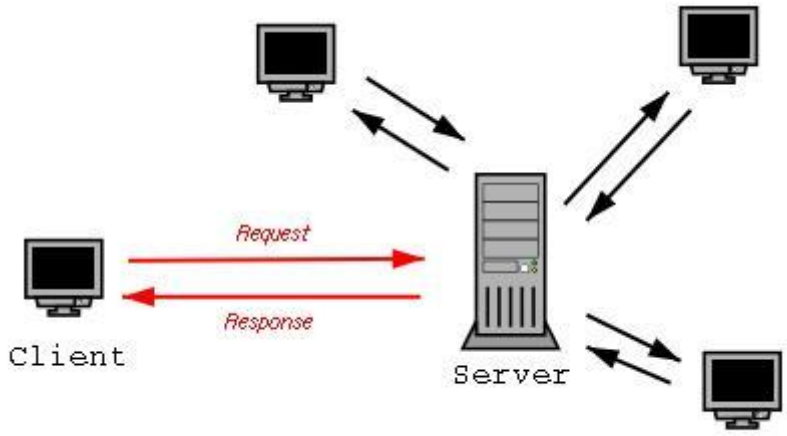

图2-1 C/S二层体系结构

它是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到Client端和Server端来实现，降低了系统的通讯开销。目前大多数应用软件系统都是Client/Server形式的两层结构，由于现在的软件应用系统正在向分布式的Web应用发展，Web和Client/Server应用都可以进行同样的业务处理，应用不同的模块共享逻辑组件;因此，内部的和外部的用户都可以访问新的和现有的应用系统，通过现有应用系统中的逻辑可以扩展出新的应用系统。这也就是目前应用系统的发展方向。

C/S模式将事务分开进行处理，实现了网络的分布式计算，很长时间里也帮助企业实现了局域网建设，完善了企业内部业务管理，提高了工作效率. 然而C/S模式在系统的集成与维护、操作界面一致性、系统的扩展性等方面都存在明显的局限性，所以就像主机/终端式网络被C/S模式的网络系统所取代一样，在InternetMtranet技术环境里，也会出现更新的系统模式。

2.1.2 B/S 体系结构

B/S结构(Browser/Server结构)结构即浏览器和服务器结构。它是随着Internet技术的兴起，对C/S结构的一种变化或者改进的结构。

在这种结构下，用户工作界面是通过WWW 浏览器来实现，极少部分事务逻辑在前端(Browser)实现，但是主要事务逻辑在服务器端(Server)实现，形成所谓三层3-tier结构。这样就大大简化了客户端电脑载荷，减轻了系统维护与升级的成本和工作量，降低了用户的总体成本(TCO)。特别是在Java这样的跨平台语言出现之后，B/S架构管理软件更是方便、快捷、高效。典型的B/S结构属于三层体系结构，它把应用分解为三个不同的逻辑层次，各有一套定义好的接口。第一层是表示层，由某种图形用户接口组成;中间层由应用逻辑组

成，由用户通过表示层调用代码，来检索所需的数据;第三层是数据层，这些数据可以由任何信息资源组成，包括。Oracle,SQL Server, MySQL这样的企业数据库，还有XML文档。三层结构把用户接口、应用逻辑、数据分开，大大提高了设计应用的灵活性。

B/S结构使数据及应用可通过不同平台、不同网络存取，与平台无关，伸缩性大，为企业提供了开放的标准的综合性计算环境。B/S集成了C/S的优点，把C/S模式的服务器端进一步深化，分解成Web服务器和数据库服务器，同时简化了客户端，仅保留表示功能，将其计算功能移至应用服务器，从而形成了由表示层、业务逻辑层、数据服务层构成的典型的三层分布式结构。表示层为用户提供人机交互界面，所有的数据录入、显示操作都在此完成，当用户需要进行数据交换时，是不允许直接访问数据库服务器，而是通过业务逻辑层提供的接口进行访问，这样保证了后台数据的安全性，同时实现了真正意义上的瘦客户;业务逻辑层负责对输入的数据按照业务逻辑进行加工处理，并实现对数据库服务器的访问：数据服务层包含应用程序需要的数据，为应用提供数据来源，保证数据的低冗余、结构性、完整性和一致性。B/S三层体系结构如图2-1所示。

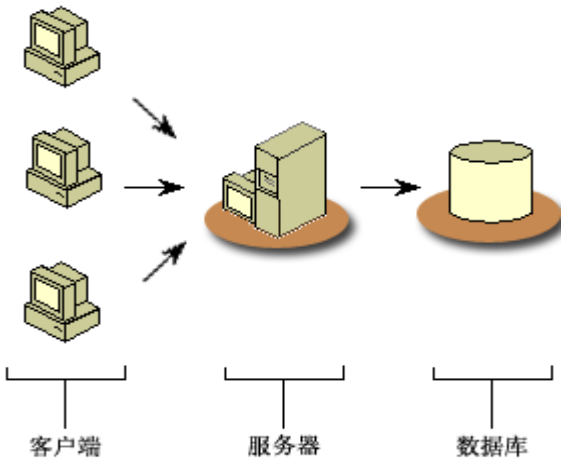

图2-2 B/S三层体系结构

B/S结构最大的优点就是可以在任何地方进行操作而不用安装任何专门的软件。只要有一台能上网的电脑就能使用，客户端零维护。系统的扩展非常容易，只要能上网，再由系统管理员分配一个用户名和密码，就可以使用了。在三层结构的Web技术中，数据库不是直接向每个客户机提供服务，而是与Web服务器沟通，实现了对客户信息服务的动态性、实时性和交互性。通过使用三层体系结构，性能、网络流量和维护问题都可以解决，但是还缺少可重用性和可伸缩性，因此又提出了N层体系结构。

当前，三层和N层结构是Web应用中的主流体系结构具有如下优势：

(1) 优化了系统结构：将系统分为三层(或N层)，业务逻辑放在应用服软件的维护集中在应用服务层，客户端的维护就相对简单多了，有利于软件维护及系统管理。

(2) 提高了应用系统的安全性：将客户端与数据库隔离起来，客户端无权限自己访问数据库，有利于安全管理。

(3) 卓越的扩展能力：若要提高系统性能、处理速度，可增加应用服务器，分担一部分

应用服务器上的工作即可，而原来的应用服务器几乎可以不用改变。

(4) 减少网络数据流量和提高数据库响应速度：基于Web的三层应用体系结构中，应用服务器层的引入有效地解决了网络瓶颈和数据库连接数过多引起数据库性能下降的问题。应用服务器层往往有多台服务器，可有效地解决客户机访问服务器层的瓶颈。应用服务器与数据库服务器可方便地采用宽带网连接，不会产生与数据库服务器层网络瓶颈。

(5) 提高系统性能：基于Web的三层体系结构能更好地调整应用体系。可以利用中间件的特点来选择路由、平衡负载，提高整个系统的性能。

2.2 Java EE 基础

2.2.1 Java EE 的基本概念

Java EE是一种利用Java平台来简化诸多企业级应用解决方案的开发、部署和管理相关的复杂问题的体系结构，提供了一个企业级的计算模型和运行环境用于开发和部署多层体系的调用。它通过提供企业计算环境所必需的各种服务，使得部署在Java EE平台上的多层应用可以实现高可用性、安全性、可扩展性和可靠性。不同计算平台都可以支持Java语言，使得基于Java EE标准开发的应用可跨平台地移植，且Java语言的安全、严格，使开发者可编写出非常可靠的代码。Java EE提供了企业计算中需要的所有服务，且更加易用;并为多数标准定义了接口，如JNDI, JDBC, Java Mail等，可与许多厂商的产品配合，容易得到广泛的支持;树立了一个广泛而通用的标准，大大简化了应用开发和移植过程。Java EE提供的多层的分布式应用模型、组件重用、一致化的安全模式以及灵活的事务控制，加快了应用程序的设计和开发，可以容易快速地建立融合了Intemet技术尤其是Web技术的N层(N-Tiers)结构的分别布式企业应用。基于Java EE技术的B/S结构具有可维护性好、可扩展性好、安全性好等优势，较好地解决了US结构所固有的可扩充性差、可维护性差、安全性差、部署麻烦等弊端。

2.2.2 Java EE 的对象模型

典型的Java EE三层模型包括表示层、业务逻辑层和数据层。在实际应用中，一个复杂的系统可能要多于这三层，而一个简单的系统则要少一些。

(1) 表示层。用户界面的开发和简单的业务逻辑都可以在这层得到实现，相对于其它两层较为简单。主要用来处理客户请求，调用相应的逻辑模块，并把结果以动态网页的形式返回到客户端。在Java EE中，表达层包括Servlet和JSP等技术。

(2) 业务逻辑层。业务逻辑层中的组件要协同工作，来解决诸如结帐、处理订单等商业逻辑。作为解决或满足某个特定业务领域(如 银行、零售或金融业)需求的逻辑业务代码。

(3)数据层。数据层运行企业信息系统软件，包括企业基础设施系统，如数据库系统、企业资源计划(Enterprise Resource Plans)、大型机事务处理(M ainframe Transaction Processing)及其它遗留信息系统(Legacy Information Systems).JAVA EE中系统的开发接口，它提供一个通用的访问SQL数据库和存储结构的机制，支持基本SQL功能的一个通用底层的应用程序编程接口。它在不同的数据库界面上提供了一个统一的用户界面，提供了多种数据库连接方式。

2.2.3 Java EE 的特点

Java EE技术的基础是Java平台的标准版。Java EE不仅巩固了标准版中的许多优点，例如“Write Once,Run Anywhere”的特性，方便数据库存取的JDBC API，还提供了在Inerent应用中保护数据的安全模式，并对EJB, Servlet, JSP以及XML技术全面支持其特点是：

(1) 面向对象的编程语言。Java EE平台是建立在Java语言基础上的，Java是真正面向对象的语言，具有丰富的数据类型和强大的功能，能完成许多复杂的功能，这是一般的Web的CGI等编程语言所无法比拟的。面向对象的设计方法，不但可以设计庞大而复杂的系统，还可以使Web应用程序具有良好的扩展性和维护性，深受Web开发人员的欢迎。

(2) 平台的独立性。Java是一个跨平台的语言，在任何平台上，只要有JVM(Java Virtual Machine)，就能在不同平台上执行同一个Java程序。Java EE标准的平台独立性使得任何符合JAVA EE标准的应用服务器之间可以共用标准的组件，从而在应用软件的开发中可以任意选择或购买符合标准的通用组件来加快开发的过程。

(3) 提供标准的系统框架和服务。Java EE平台提供了事务处理、对象生存控制、状态维护、并发控制、安全检测、资源共享等系统服务。获取这些服务的代价并不高，不用编程，而只要通过较简单的配置就行。这使得开发者从繁琐的系统设计中解脱出来，将精力主要放在商业逻辑上，以提高应用的质量和加快开发的速度。

(4) 适合团体开发。Java EE的构架非常适合团体开发的模式。它将应用分成表示层、业务逻辑层和数据层，可以使企业开发中的美工、系统分析员、编程人员各司其职，发挥各自的长处，特别是Java EE构架通用的MVC模式，能够将系统各个层面的功能独立开来，这种构架非常适合团队开发的模式，提高了工作的效率。

(5) 可控性好。Java EE安全控制和状态控制机制非常完善，这种控制机构使得整个应用拥有统一的状态转换规则。这样，不会让用户进入到不该进入的页面而引起状态的棍乱，增加了系统的安全性。在Java EE中状态的可控性使电子商务的开发更加简单和可靠，为顾客提供更好的服务。

(6) 与其他资源的集成性好。Java EE平台以其丰富的系统功能，通过JDBC,JTA, JMS, XML, JNDL CORBA等API几乎可以与所有关系型数据库、事务处理服务器、消息处理服务器、目录服务器和邮件服务器等进行无缝的集成，完美地结合成一个整体，保护原有的投资，并且为将来的发展留有广阔的空间。

2.3 JSP 技术

JSP的英文全称是Java Server Page，中文全称是Java服务器端语言。自JSP推出后，众多大公司都支持JSP技术的服务器，如IBM、Oracle、Bea公司等，所以JSP迅速成为商业应用的服务器端语言。JSP技术能让Web开发员和网页设计员快速地开发出容易维护的动态Web主页。用JSP开发的Web应用是跨平台的，即能在Llnux下运行，也能在其它操作系统上运行。

JSP技术使用Java编程语言编写类XML的tags和scirptlets来封装产生动态网页的处理逻辑。网页还能通过tags和scriptlets访问存在于服务端的资源(例如JavaBesns)的应用逻辑。JSP

将网页逻辑与网页设计和显示分离，支持可重用的基于组件的设计，使基于Web的应用程序的开发变得迅速和容易。

JSP技术是servlet技术的扩展。Servlet是Java技术对CDI编程的回答。Servlet程序在服务器端运行，动态地生成Web页面。与传统的CGI和许多其他类似CGI的技术相比，Java servlet具有更高的效率，更容易使用，功能更强大，具有更好的可移植性，更节省投资。

2.4 Struts2

Struts2号是在另一个赫赫有名的框架WebWork基础上发展起来的。从某种程度上来讲，Struts2没有继承Struts1的血统，而是继承WebWork的血统。或者说，WebWork衍生出了Struts2，而不是Struts1衍生了Struts2。因为Struts2是WebWork的升级，而不是一个全新的框架，因此稳定性、性能等各方面都有很好的保证：而且吸收了Struts 1和WebWork两者的优势，因此，是一个非常值得期待的框架。

Apache Struts2是一个优雅的，可扩展的Java EE web框架。框架设计的目标贯穿整个开发周期，从开发到发布，包括维护的整个过程。

2.4.1 Struts2 特征

Struts2 Action类可以实现一个Action接口，也可实现其他接口，使可选和定制的服务成为可能。Struts2提供一个ActionSupport基类去实现 常用的接口。Action接口不是必须的，任何有execute标识的POJO对象都可以用作Struts2的Action对象。

(1) Struts2 Action对象为每一个请求产生一个实例，因此没有线程安全问题。（实际上，servlet容器给每个请求产生许多可丢弃的对象，并且不会导致性能和垃圾回收问题）

(2) Struts2 Action不依赖于容器，允许Action脱离容器单独被测试。如果需要，Struts2 Action仍然可以访问初始的request和response。但是，其他的元素减少或者消除了直接访问HttpServetRequest 和 HttpServletResponse的必要性。

(3) Struts2 Action可以通过初始化、设置属性、调用方法来测试，“依赖注入”支持也使测试更容易。

(4) Struts2直接使用Action属性作为输入属性，消除了对第二个输入对象的需求。输入属性可能是有自己(子)属性的rich对象类型。Action属性能够通过web页面上的taglibs访问。

(5) Struts2可以使用JSTL，但是也支持一个更强大和灵活的表达式语言－－"Object Graph Notation Language" (OGNL).

(6) Struts 2使用 "ValueStack"技术，使taglib能够访问值而不需要把你的页面（view）和对象绑定起来。ValueStack策略允许通过一系列名称相同但类型不同的属性重用页面（view）。

(7) Struts2使用OGNL进行类型转换。提供基本和常用对象的转换器。

(8) Struts2支持通过validate方法和XWork校验框架来进行校验。XWork校验框架使用为属性类类型定义的校验和内容校验，来支持chain校验子属性。

(9) Struts2支持通过拦截器堆栈（Interceptor Stacks）为每一个Action创建不同的生命周期。堆栈能够根据需要和不同的Action一起使用。

2.5 Spring IoC 与 AOP

Spring是一个开源框架，是为了解决企业应用程序开发复杂性而创建的。框架的主要优势之一就是其分层架构，分层架构允许您选择使用哪一个组件，同时为 Java EE 应用程序开发提供集成的框架。

Spring 框架是一个分层架构，由 7 个定义良好的模块组成。Spring 模块构建在核心容器之上，核心容器定义了创建、配置和管理 bean 的方式，如图2-3 所示。

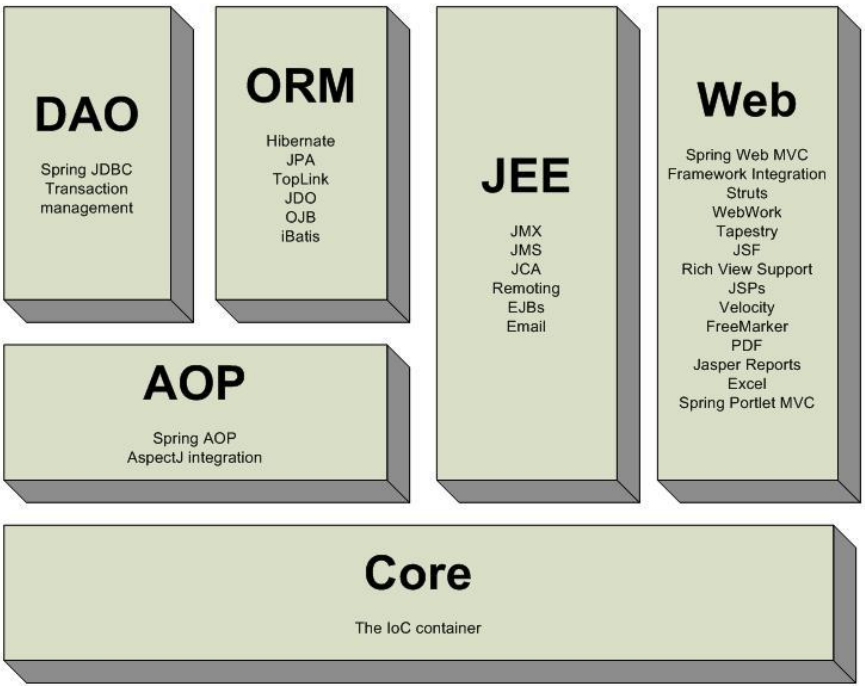

图2-3 Spring 框架概述

Spring框架的功能可以用在任何Java EE服务器中，大多数功能也适用于不受管理的环境。Spring的核心要点是：支持不绑定到特定 Java EE服务的可重用业务和数据访问对象。毫无疑问，这样的对象可以在不同 Java EE 环境 （Web 或 EJB）、独立应用程序、测试环境之间重用。

控制反转模式（也称作依赖性介入）的基本概念是：不创建对象，但是描述创建它们的方式。在代码中不直接与对象和服务连接，但在配置文件中描述哪一个组件需要哪一项服务。容器（在 Spring 框架中是 IOC 容器） 负责将这些联系在一起。

依赖注入（DI）背后的基本原理是对象之间的依赖关系（即一起工作的其它对象）只会通过以下几种方式来实现：构造器的参数、工厂方法的参数，或给由构造函数或者工厂方法创建的对象设置属性。因此，容器的工作就是创建bean时注入那些依赖关系。相对于由bean自己来控制其实例化、直接在构造器中指定依赖关系或者类似服务定位器（Service Locator）模式这3种自主控制依赖关系注入的方法来说，控制从根本上发生了倒转，这也正是控制反转（Inversion of Control， IoC） 名字的由来。

应用DI原则后，代码将更加清晰。而且当bean自己不再担心对象之间的依赖关系（甚至不知道依赖的定义指定地方和依赖的实际类）之后，实现更高层次的松耦合将易如反掌。DI主要有两种注入方式，即Setter注入和构造器注入。

在典型的 IOC 场景中，容器创建了所有对象，并设置必要的属性将它们连接在一起，决定什么时间调用方法。下面列出了 IOC 的一个实现模式。

面向方面的编程，即 AOP，是一种编程技术，它允许程序员对横切关注点或横切典型的职责分界线的行为（例如日志和事务管理）进行模块化。AOP 的核心构造是*方面*，它将那些影响多个类的行为封装到可重用的模块中。

AOP 和 IOC 是补充性的技术，它们都运用模块化方式解决企业应用程序开发中的复杂问题。在典型的面向对象开发方式中，可能要将日志记录语句放在所有方法和 Java 类中才能实现日志功能。在 AOP 方式中，可以反过来将日志服务*模块化*，并以声明的方式将它们应用到需要日志的组件上。当然，优势就是 Java 类不需要知道日志服务的存在，也不需要考虑相关的代码。所以，用 Spring AOP 编写的应用程序代码是松散耦合的。AOP的功能完全集成到了Spring事务管理、日志和其他各种特性的上下文中。

2.6 Hibernate 对象持久化

Hibernate是一个开放源代码的对象关系映射框架，它对JDBC进行了非常轻量级的对象封装，使得Java程序员可以随心所欲的使用对象编程思维来 操纵数据库。Hibernate可以应用在任何使用JDBC的场合，既可以在Java的客户端程序使用，也可以在Servlet/JSP的Web应用中使用，最具革命 意义的是，Hibernate可以在应用EJB的JAVA EE架构中取代CMP，完成数据持久化的重任。

Hibernate的结构体系图如图2-4所示：

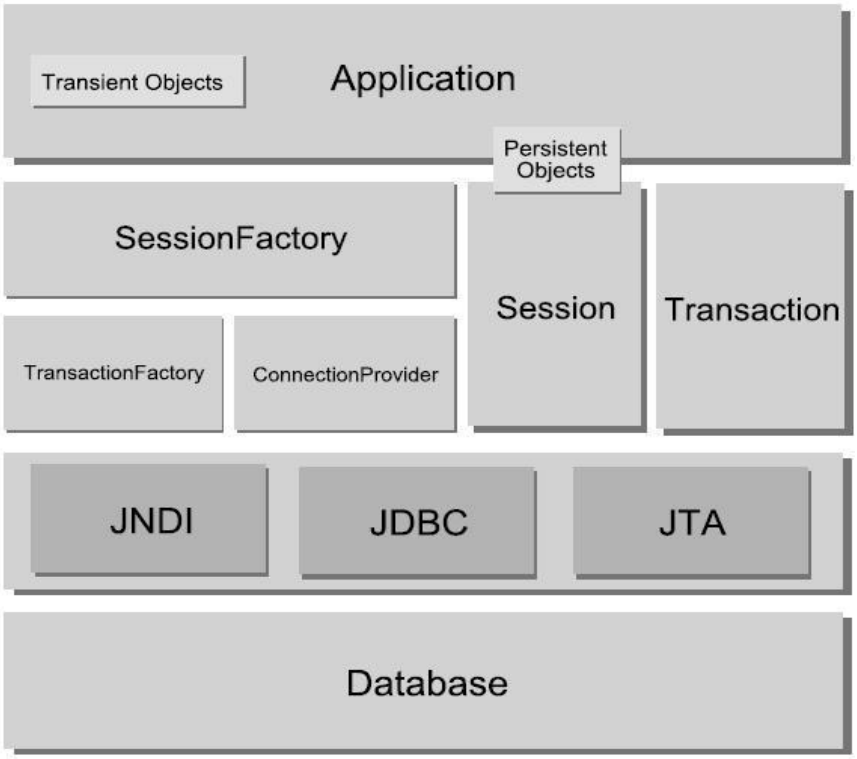

图2-4 Hibernate体系结构

2.5.1 在 Hibernate 应用中 Java 对象的状态

一个持久化类的实例可能处于三种不同状态中的某一种。 这三种状态的定义则与所谓的持久化上下文(persistence context)有关。Hibernate 的 Session 对象就是这个所谓的持久化上下文：

瞬态（transient）：该实例从未与任何持久化上下文关联过。它没有持久化标识（相当于主键值）。

持久化(persistent)：实例目前与某个持久化上下文有关联。 它拥有持久化标识（相当于主键值），并且可能在数据库中有一个对应的行。 对于某一个特定的持久化上下文，Hibernate *保证*持久化标识与 Java 标识（其值代表对象在内存中的位置）等价。

脱管(detached)：实例曾经与某个持久化上下文发生过关联，不过那个上下文被关闭了，或者这个实例是被序列化(serialize)到另外的进程。 它拥有持久化标识，并且在数据库中可能存在一个对应的行。 对于脱管状态的实例，Hibernate 不保证任何持久化标识和 Java 标识的关系。

2.5.2 Hibernate 缓存

缓存是介于应用程序和物理数据源之间，其作用是为了降低应用程序对物理数据源访问的频次，从而提高了应用的运行性能。缓存内的数据是对物理数据源中的数据的复制，应用程序在运行时从缓存读写数据，在特定的时刻或事件会同步缓存和物理数据源的数据。

Hibernate 的缓存包括 Session 的缓存和 SessionFactory 的缓存，其中 SessionFactory 的缓存又可以分为两类：内置缓存和外置缓存。Session 的缓存是内置的，不能被卸载，也被称为 Hibernate 的第一级缓存。SessionFactory 的内置缓存和 Session 的缓存在实现方式上比较相似，前者是 SessionFactory 对象的一些集合属性包含的数据，后者是指 Session 的一些集合属性包含的数据。SessionFactory 的内置缓存中存放了映射元数据和预定义 SQL 语句，映射元数据是映射文件中数据的拷贝，而预定义 SQL 语句是在 Hibernate 初始化阶段根据映射元数据推导出来，SessionFactory 的内置缓存是只读的，应用程序不能修改缓存中的映射元数据和预定义 SQL 语句，因此 SessionFactory 不需要进行内置缓存与映射文件的同步。SessionFactory 的外置缓存是一个可配置的插件。在默认情况下，SessionFactory 不会启用这个插件。外置缓存的数据是数据库数据的拷贝，外置缓存的介质可以是内存或者硬盘。SessionFactory 的外置缓存也被称为 Hibernate 的第二级缓存。

第 3 章 网上书店系统的设计

3.1 网上书店需求分析

3.1.1 网上书店系统可行性分析

随着计算机网络的发展，上网人数日益增多，人们已经习惯通过网络商城的方式在网络上进行购物和接受服务。因此，我们通过这样的方式，在计算机网络上提供各种商品的销售，能够被广大用户所接受，可以拓宽企业的客户群。

开发网上商城购物系统的技术已经非常成熟，首先，从硬件上讲，计算机硬件速度现在已经不是问题，大容量高速度的硬盘十分普遍，同时网络的速度普遍可以达到100M，这些为电子商务的运行打下坚实的基础。从软件的角度上讲，数据库技术已经相当成熟(目前用得比较多的有SQLserver、Oracle和MySQL等)，并且处理能力也非常强，这为海量数据的存储和处理打下了坚实的基础，同时，开发网站的工具也非常多(比如：ASP,JSP,PHP等)，并且相当成熟。有了这些技术的支持，我们成功开发一个网上商城购物系统没有任何技术风险。

开设网上书店可以大大降低成本，包括租赁店面成本、管理费用、办公费用等，同时，由于网络的广泛性，大大提高了顾客的范围，而且摆脱了时间限制，从经济角度来看有很大的发展潜力。

3.1.2 网上书店系统概要分析

网上书店同传统的店面书店相比，网上书店的经营方式和销售渠道是全新的，它解决了许多传统书店的局限性：它24小时的全天候和全方位服务是店面书店所不能比及的；成本低廉更是开设网上书店的主要原因；而与其他商品相比，书籍运送几乎不怕碰撞碎裂，不具时效性；同时书本具有功能单一，形式简单，易于判断和选择而独具优势，最适合于网上交易；再次是单价低，降低了消费者第一次在网络购物的门槛，所以开设网上书店为广大读者提供了很大的便利性，具有很大的发展潜力。

3.1.3 网上书店系统功能分析

网上书店系统共分两个模块：前台模块和后台模块两部份。在前台模块中，是用户购物操作及其个人信息维护，包括用户在线注册登录、个人信息维护，浏览图书、查询图书信息、购物车管理、订单查看等操作。后台模块是管理员对整个系统的维护，包括：会员管理、图书信息管理、分类管理、订单管理等功能。

3.2 网上书店系统流程图

用户在网上书店购物按如图3-1进行购物，下订单。

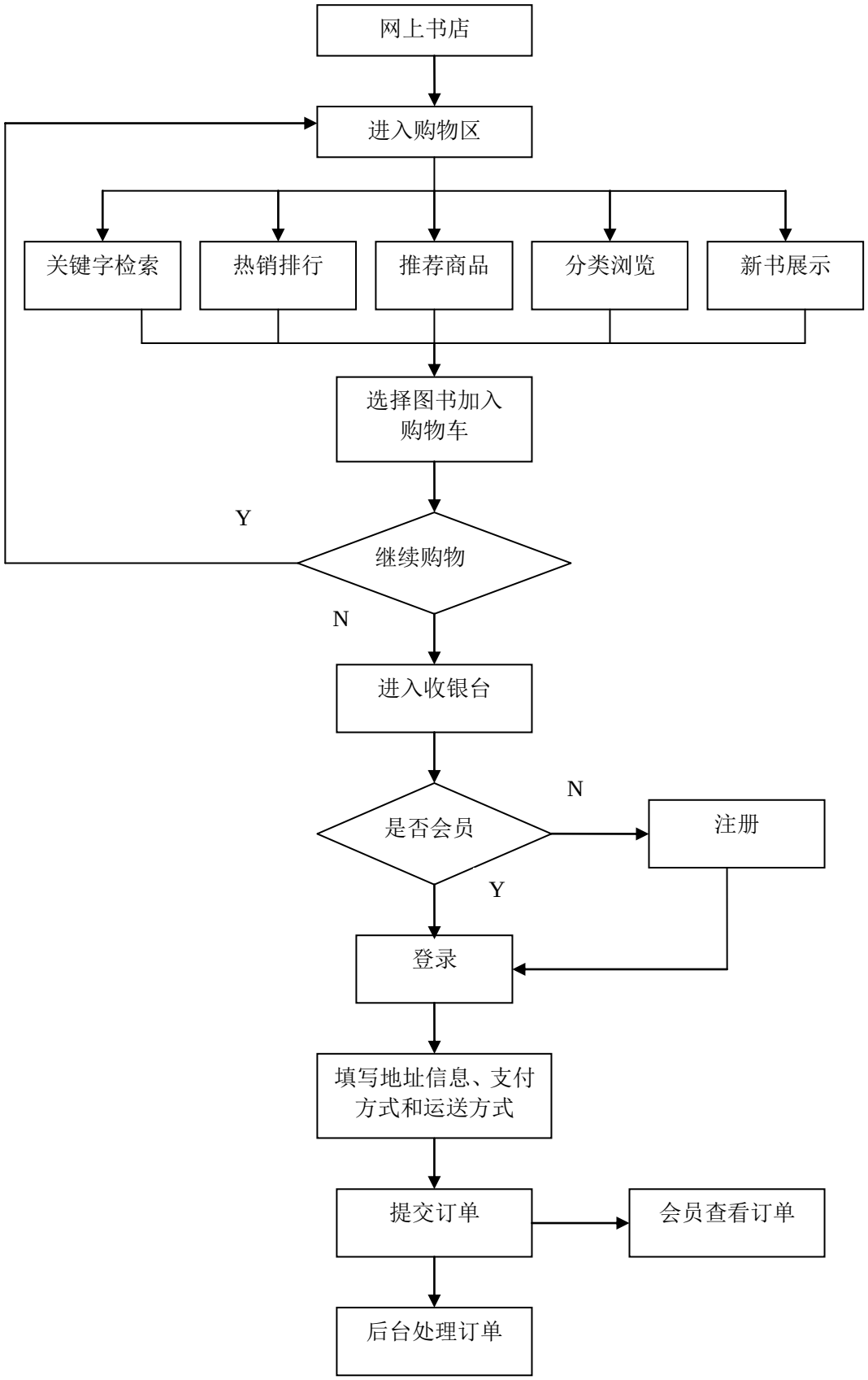

图3-1 网上书店系统购物流程图

3.3 系统功能模块划分

网上书店系统的开发，主要是对用户进行购物为中心进行功能模块划分的。一个典型的网上书店系统主要实现网上查询用户登录注册、图书浏览、图书查询、购物车管理、订单生成、添加图书、分类管理、订单管理等功能。模块的划分使系统清晰明了，容易升级和维护。

系统总功能模块：分为前台模块和后台模块，前台主要为游客和会员进行操作的模块，后台是管理员进行会员、商品、订单等信息进行维护和操作。

网上书店系统结构如图3-2所示。

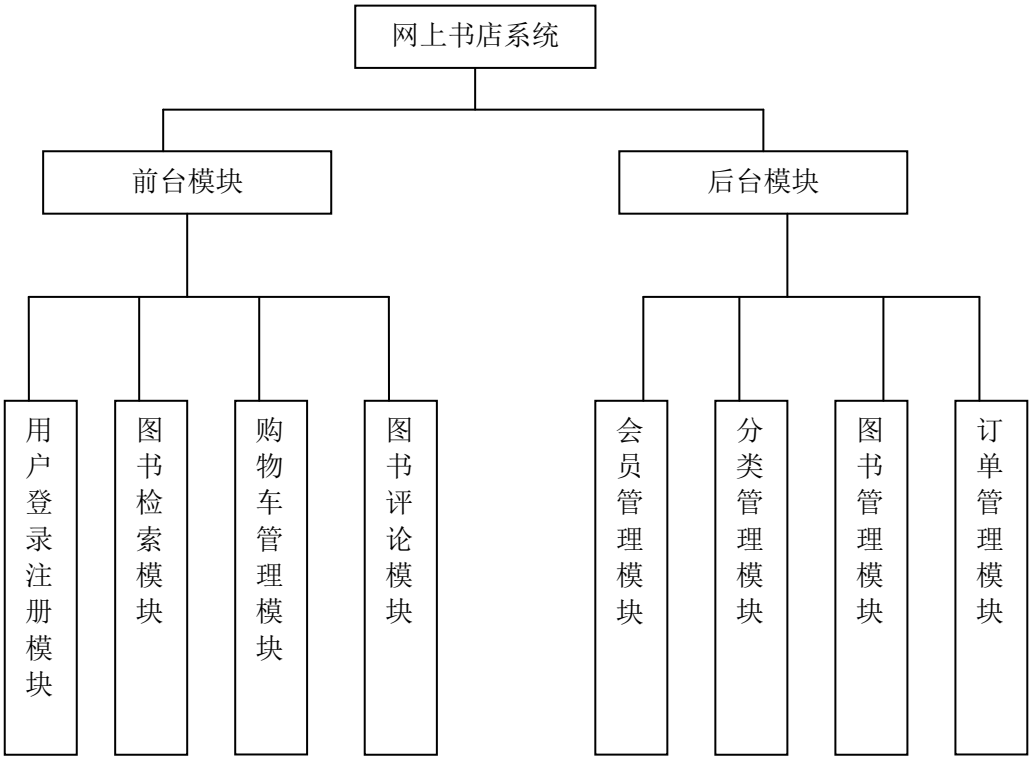

图3-2 网上书店系统总功能模块

3.3.1 用户注册登录模块

用户注册登录模块用于建立网上书店用户群体，并能详细记录用户档案。用户可在网上在线注册，登录。然后浏览图书，找到自己所需要的图书，并购买。

该功能模块能够提供以下几个子功能：

(1) 用户在线注册

(2) 用户登录管理

(3) 用户资料维护

只有进行登录并通过身份验证的用户，才可以进行订单提交等后续的处理。在用户购物结束离开该网站时，可以选择退出，清空对应的登录成功信息。

3.3.2 图书检索模块

图书检索查询模块用于为用户提供便捷的搜索所需图书，并了解相关的图书信息。以引导客户的购物选择，方便用户快速方便找到自己满意的图书。其结构如图3-3所示。该功

能模块提供一下几个子功能

(1) 可以根据分类进行检索。查看指定分类下图书。

(2) 关键字检索。根据关键字匹配快速找到用户满意的图书。

(3) 推荐图书检索

(4) 热销图书检索

(5) 新书检索

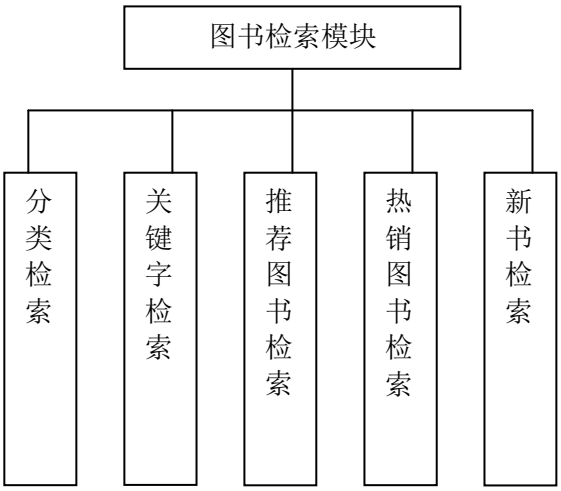

图3-3 图书管理模块

3.3.3. 图书评论模块

用户可以根据自己对图书的理解，进行评论，以便信息相互交流，选择适合自己的图书。用异步技术实现。

3.3.4 购物车管理模块

用户在购物车里添加图书、更改图书数量、删除购物车图书和清空购物车等。结构如图3-4所示。

购物车是网络购物系统中的一种重要功能，与在真实商店中的购物过程类似。在网络环境下，用户可能在某个网络商店网站的不同页面之间跳转，以选购自己喜爱的商品，最后将选中的所有商品放在购物车中统一到付款台结账。服务器通过会话(Session)追踪每个用户的行动，以保证在结账时每件商品都物有其主。购物车的功能包括以下几项：

(1) 把图书添加到购物车，即订购

(2) 删除购物车中已订购的图书

(3) 修改购物车中某一本图书的订购数量

(4) 显示购物车中的图书清单及数量、价格

(5) 清空购物车

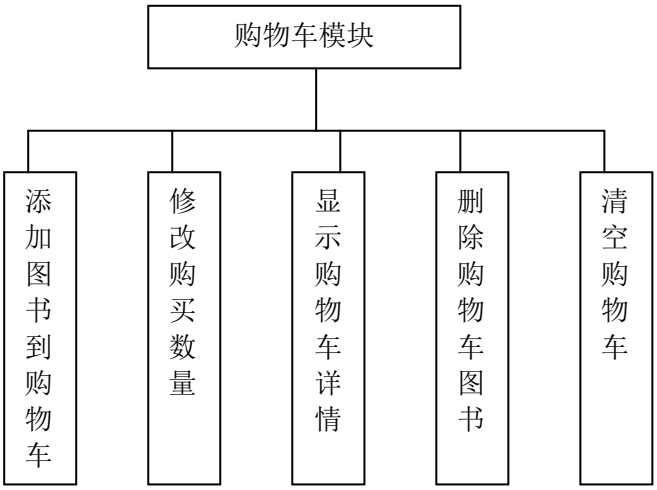

图3-4 购物车管理模块

3.3.5 会员管理模块

该模块主要完成的是用户对自己一些基本信息进行维护的操作和管理员对已经注册的用户资料进行维护的操作。结构如图3-5所示。

会员管理模块包括以下功能：

(1) 会员信息维护：在注册时用户的基本资料可能不是完全正确或者不是很完整，那么在以会员的身份登录后，仍然可以修改个人资料，也就是通过该模块进行操作，该模块还可以修改用户当初设定的密码。

(2) 查看会员资料：该模块主要完成的是管理员对注册用户资料信息进行维护的操作，具体功能就是查询用户信息，并返回给管理员。查看用户资料模块为站点准确地发货给用户提供了保证。

(3) 会员检索：管理员可以输入关键字对匹配的会员进行搜索。

(4) 会员状态设置：冻结用户或激活用户，若一个会员被冻结了，其不能登录直到重新激活为止。

(5) 删除会员：该功能是管理员对于资料不齐全或是一些恶意的注册帐号管理员可以对其进行删除。

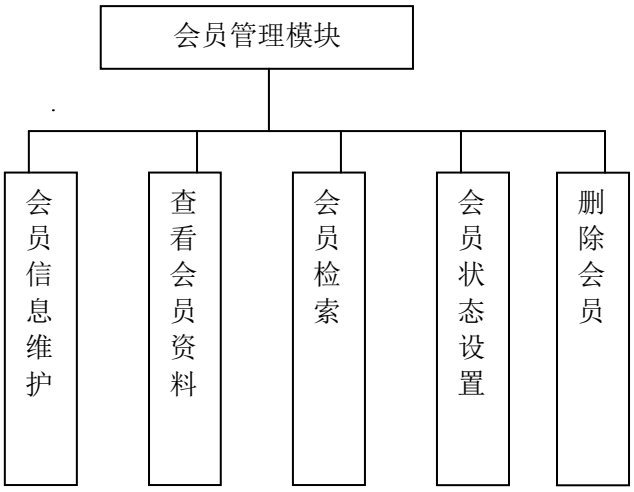

图3-5 会员管理模块

3.3.6 图书分类管理模块

图书总共设二级分类，以便能对图书进行较好的分类，管理员可以添加，删除图书分类，在同级目录下不能存在相同的分类。用户可以根据自己的需求在特定分类里查找自己所需图书。

3.3.7 图书管理模块

图书管理模块为管理员可对图书进行添加、查看、编辑、图书必要信息设置等功能。结构如图3-6所示。

(1) 添加图书：该功能是在数据库中添加新的图书信息，还可设置该图书是否为新书，是否推荐给用户，以便用户了解更多的信息。系统可以从数据库中读出图书信息并显示。用户可以在购物界面看到图书信息，然后进行购买操作。添加图书功能是该系统的一个重要功能。

(2) 查看图书资料：该功能主要有用户查看图书，管理员查看图书。用户查看图书可以看图书的相关信息及评论，决定是否购买图书。管理员查看图书资料可以核实有没有错误的信息输入，方便管理员对图书信息的维护;管理员可以随时查看图书的资料(销售量等信息)，方便管理员判断是否应该给顾客推荐或是不再销售该书籍。

(3) 修改图书资料：该模块主要功能是修改图书的信息。管理员如果发现该图书的信息有误，可以通过该模块对图书信息进行修改，还有可以修改是否是新书和是否是为顾客推荐的图书。

(4) 删除已有图书：该模块的主要功能是让图书下架。如果一些图书市场已经饱和不再需要出售该图书，则该图书需要下架，管理员可以通过该功能模块完成此操作.

(5) 图书检索：可以方便用户方便的快速的寻找自己的图书，管理员可也进行图书检索方便进行查看和编辑。

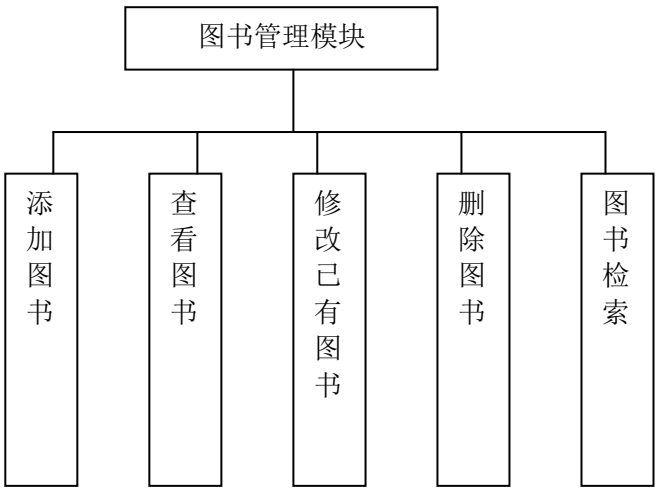

图3-6 图书管理模块

3.3.8.订单管理模块

该模块的主要功能是完成对用户向系统所提交定单的操作。结构如图3-7所示

用户查看订单：用户在提交购物车后，可以查看订单信息，查看那是否订单已经处理。

管理员查看订单：该功能是系统从数据库中提取用户所提交的定单信息。管理员可以从订单列表中可以看到订单号、品种数、真实姓名、付款方式、运送方式、订书日期等信息.

更改订单状态：在用户已经提交订单后，管理员进行一些信息的核对，若信息完整正确，则对其进行处理、发货。

删除订单：对一些信息不完整或已经失效的订单，管理员可对其进行删除。

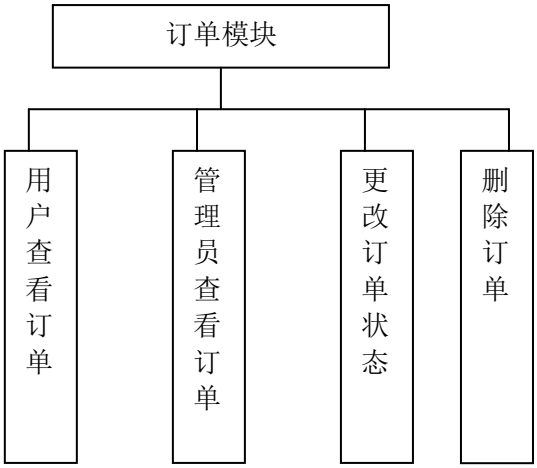

图3-7 订单模块

3.4 数据库设计

3.4.1 数据库的选择

MySQL是一个小型关系型数据库管理系统。由于其体积小、速度快、总体拥有成本低且开放源代码，MySQL被广泛地应用在Internet上的中小型网站中。

与其他的大型数据库例如Oracle、DB2、SQL Server等相比，MySQL自有它的不足之处，如规模小、功能有限（MySQL Cluster的功能和效率都相对比较差）等，但是这丝毫也没有减少它受欢迎的程度。

MySQL有如下特点：能够工作在众多不同的平台上；提供了事务性和非事务性存储引擎；极快的基于线程的内存分配系统；SQL函数是使用高度优化的类库实现的，运行很快；通常，在完成查询初始化后，不存在存储器分配等。

对于一般的个人使用者和中小型企业来说，MySQL提供的功能已经绰绰有余，而且由于MySQL是开放源码软件，因此可以大大降低总体拥有成本。

3.4.2 数据库的逻辑设计

网上书店后台数据库名为bookstore，共设八张数据表。

其各个数据表的结构如下所示：

(1) 会员表

会员表存放用户个人基本信息，包括地址信息、用户当前状态（激活或冻结）等。其

字段设置如表3-1所示。

表3-1 会员表

字段名	字段类型	字段长度	字段限制	注释
membersId	int		非空	用户 ID
loginName	varchar	20	非空	用户登录名
trueName	varchar	20		用户真实姓名
password	varchar	20	非空	密码
gender	varchar	2	非空	性别
address	varchar	100		地址
postcode	varchar	10		邮编
email	varchar	50		邮箱
amount	double			消费总额
tel	varchar	20		联系电话
status	int		非空	状态
registerTime	bigint		非空	注册时间
lastModifyTime	bigint		非空	资料更新时间

(2) 图书表

图书表存放图书基本信息，销售量，库存量等，是显示和维护图书信息的依据，可根据表里面的一些字段进行排序如。可根据isCommend查询推荐图书。其字段设置如表3-2所示。

表3-2 图书信息表

字段名	字段类型	字段长度	字段限制	注释
bookId	int		非空	图书 Id
isbn	varchar	20	非空	ISBN
typeId	int		非空	分类 Id
bookName	varchar	60	非空	书名
publisher	varchar	50	非空	出版社
author	varchar	60	非空	作者
introduce	varchar	2000	非空	简介
picture	varchar	60		封面图片
price	double		非空	单价
rebate	double		非空	折扣
publishDate	bigint		非空	出版日期
isCommend	char	1	非空	是否推荐
isNewBook	char	1	非空	是否为新书
saleCount	int		非空	销售量
nowCount	int		非空	库存量
inTime	bgint		非空	入库时间

(3) 第一级分类表

对图书进行分类，可以更好的管理和查询图书。若只存在一级分类图书并不是能很好

的表示，分类过多会造成复杂和难以管理，所以分成二级。第一级分类下存在第二级分类表。其字段设置如表3-3所示。

表3-3 第一级分类表

字段名	字段类型	字段长度	字段限制	注释
typeId	int		非空	分类 Id
typeName	varchar	20	非空	分类名
lastModifyTime	bigint		非空	最后修改时间

(4) 第二级分类表 其字段设置如3-4所示。

表3-4 第二级分类表

字段名	字段类型	字段长度	字段限制	注释
typeId	int		非空	分类 Id
superId	int		非空	上级分类 Id
typeName	varchar	20	非空	分类名
lastModifyTime	bigint		非空	最后修改时间

(5) 图书评论表

用户可以对图书进行评论，可以询问相关读者对此书的评价，以能更好的了解图书。其字段设置如表3-5所示。

表3-5 图书评论表

字段名	字段类型	字段长度	字段限制	注释
commented	int		非空	评论 Id
bookId	int		非空	图书 Id
memberId	int		非空	会员 Id
loginName	varchar	20	非空	登录名
content	varchar	200	非空	内容
commentTime	bigint		非空	评论时间

(6) 管理员表

管理员是后台操作的执行者。其字段设置如表3-6所示

表3-6 管理员表

字段名	字段类型	字段长度	字段限制	注释
adminId	int		非空	管理员 Id
loginName	varchar	20	非空	登录名
trueName	varchar	20	非空	真名
password	varchar	20	非空	密码
createTime	bigint		非空	创建时间
lastModifyTime	bigint		非空	最后修改时间

(7) 订单表

存放具体订单信息，是网上购物系统的一个重要组成部分。其字段设置如表3-7所

示。

表3-7 订单表

字段名	字段类型	字段长度	字段限制	注释
ordered	int		非空	订单 Id
membersId	int		非空	用户 Id
trueName	varchar	20	非空	收货人姓名
address	varchar	100	非空	地址
postcode	varchar	10	非空	邮编
tel	varchar	20		联系电话
carry	varchar	20	非空	运送方式
pay	varchar	20	非空	付款方式
orderTime	bigint		非空	下单时间
status	int		非空	状态
amount	double		非空	订单金额
dealTime	bigint		非空	处理时间

(8) 订单明细表

存放订单里每一项的具体信息，如商品号、价格、数量等。其字段设置如表3-8所示。

表3-8 订单明细表

字段名	字段类型	字段长度	字段限制	注释
odId	int		非空	订单项 Id
ordered	int		非空	订单 Id
boodId	int		非空	图书 Id
bookName	varchar	60	非空	图书名
price	double		非空	单价
rebate	double		非空	折扣
Count	int		非空	数量

3.4.3 ER 图

ER 图提供了表示实体型、属性和联系的方法，用来描述现实世界的概念模型。是以实体(个体,类)为基础的物理语言,因为关系是实体之间的关系，是由实体来(联合)定义的,所以是实体在先,关系在后的。

第一级分类和第二级分类之间是一对多关系，两级分类使图书能够较好的管理。

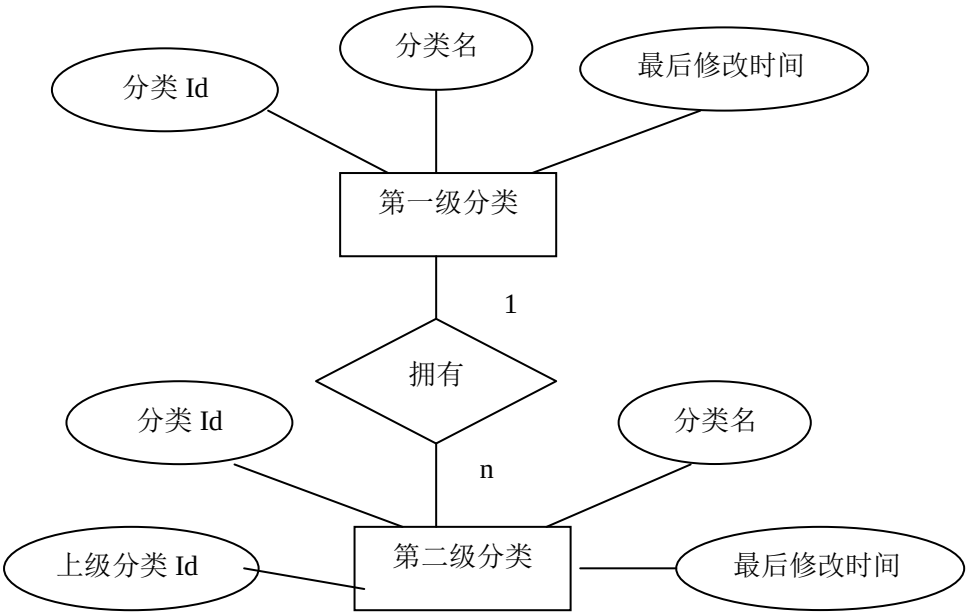

图 3-8 分类之间 ER 图

用户在网上书店可多次购书，即可以拥有多个订单。用户与订单的关系是一对多的关系。

订单与订单明细之间是一种一对多关系，即一个订单对应多个订单项，订单项中关联到一个订单号和其商品的标识、数量和单价。

其结构如图3-9所示。

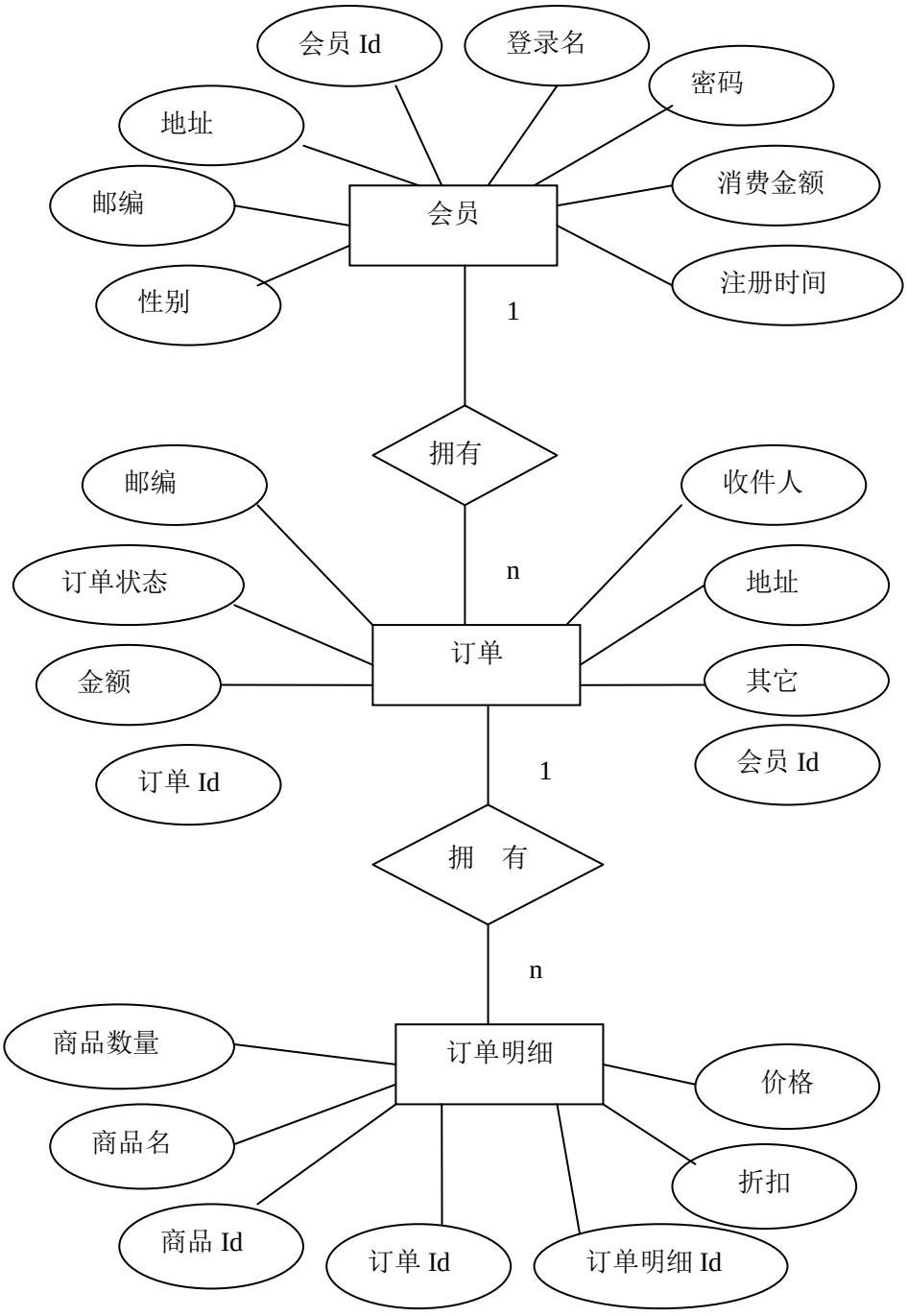

图 3-9 用户、订单和订单明细 ER 图

第 4 章 网上书店系统现

4.1 设计模式

在JavaEE开发中，最常用的两个设计模式是MVC和DAO设计模式。这两个模式的运用，能使系统有很好的层次架构。

4.1.1 MVC 设计模式

MVC(Model-View- Controller)是八十年代为编程语言Smalltalk-80发明的一种软件设计模式。MVC模式将交互式应用分成模型（Model）、视图（View）和控制器（Controller）三部分。MVC模式的结构如图4-1所示。

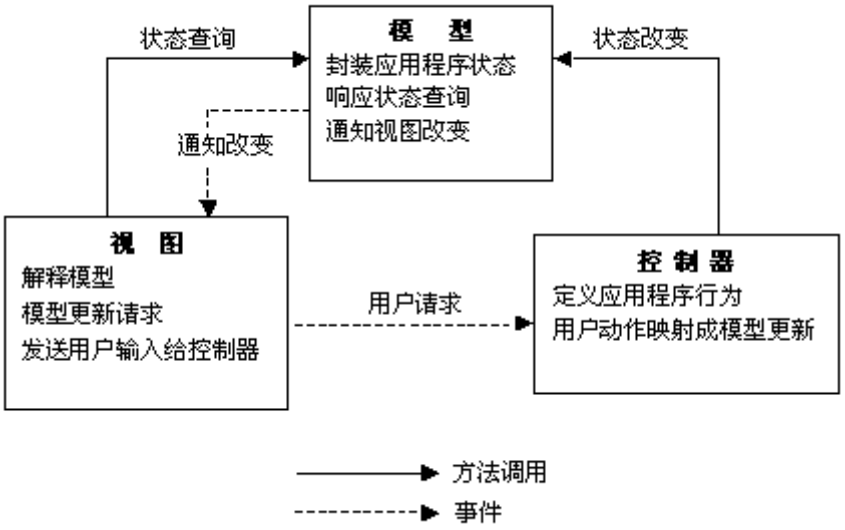

图4-1 MVC模式图

模型是指从现实世界中挖掘出来的对象模型，是应用逻辑的反映。一个模型能为多个视图提供数据。由于同一个模型可以被多个视图重用，所以提高了应用的可重用性。模型封装了数据和对数据的操作，是实际进行数据处理的计算的地方。就是业务流程/状态的处理以及业务规则的制定。业务流程的处理过程对其它层来说是黑箱操作，模型接受视图请求的数据，并返回最终的处理结果。业务模型的设计可以说是MVC最主要的核心。

视图代表用户交互界面，是应用和用户之间的接口，它负责将应用显现给用户和显示模型的状态。

控制器负责视图和模型之间的交互，控制对用户输入的响应响应方式和流程，它主要负责两方面的动作:用户的请求分发到相应的模型;将模型的改变及时反应到视图上。MVC将这些对象分离以提高灵活性和复用性。

现在来总结MVC处理过程。首先控制器接收用户的请求，并决定应该调用哪个模型来处理;然后模型根据用户请求进行相应的业务逻辑处理，并返回数据，最后控制器调用相应的视图来格式化模型返回的数据，并通过视图呈现给用户。

由于不同的层各司其职，每一层不同的应用具有某些相同的特征，有利于通过工程化、工具化管理程序代码。

4.1.2 DAO 设计模式

DAO(Data Access Object)模式实际上是两个模式的组合，即Data Accessor 模式和Active Domain Object 模式，其中 Data Accessor 模式实现了数据访问和业务逻辑的分离，而Active Domain Object 模式，其中Data Accessor模式实现了数据访问和业务逻辑的分离，而Active Domain Object 模式实现了业务数据的对象化封装。

DAO模式通过对业务层提供数据抽象层接口，实现了以下目标：

(1) 数据存储逻辑的分离

通过对数据访问逻辑进行抽象，为上层机构提供抽象化的数据访问接口。业务层无需关心具体的select,insert,update操作，这样，一方面避免了业务代码中混杂JDBC调用语句，使得业务落实实现更加清晰，另一方面，由于数据访问几口语数据访问实现分离，也使得开发人员的专业划分成为可能。某些精通数据库操作技术的开发人员可以根据接口提供数据库访问的最优化实现，而精通业务的开发人员则可以抛开数据曾德繁琐细节，专注于业务逻辑编码。

(2). 数据访问底层实现的分离

DAO模式通过将数据访问计划分为抽象层和实现层，从而分离了数据使用和数据访问的地称实现细节。这意味着业务层与数据访问的底层细节无关，也就是说，我们可以在保持上层机构不变得情况下，通过切换底层实现来修改数据访问的具体机制，常见的一个例子就是，我们可以通过仅仅替换数据访问曾实现，将我们的系统部署在不同的数据库平台之上。

(3). 资源管理和调度的分离

在数据库操作中，资源的管理和调度是一个非常值得关注的主题。大多数系统的性能瓶颈往往并非集中于业务逻辑处理本身。在系统涉及的各种资源调度过程中，往往存在着最大的性能黑洞，而数据库作为业务系统中最重要的系统资源，自然也成为关注的焦点。DAO模式将数据访问逻辑从业务逻辑中脱离开来，使得在数据访问层实现统一的资源调度成为可能，通过数据库连接池以及各种缓存机制(Statement Cache,Data Cache等，缓存的使用是高性能系统实现的一个关键所在)的配合使用，往往可以保持上层系统不变的情况下，大幅度提升系统性能。

(4) 数据抽象

在直接基于JDBC调用的代码中，程序员面对的数据往往是原始的RecordSet数据集，诚然这样的数据集可以提供足够的信息，但对于业务逻辑开发过程而言，如此琐碎和缺乏寓意的字段型数据实在令人厌倦。

DAO 模式通过对底层数据的封装，为业务曾提供一个面向对象的接口，使得业务逻辑开发员可以面向业务中的实体进行编码。通过引入DAO模式，业务逻辑更加清晰，且富于形象性和描述性，这将为日后的维护带来极大的便利。

4.2 开发平台

操作系统：windows XP

JDK：JDK1.6.0_05

WEB 服务器：Tomcat 6.0

数据库：MySQL 5.0

IDE：Eclipse3.3 ，MyEclipse6.5

Java 框架：Struts2.0.11，Spring2.5，Hibernate3.2

4.3 项目目录结构规划

在一个 Web 项目中，目录结构需要进行良好的规划，通常包括 Java 文件、JSP 页面、图片、配置文件、Javascript 文件等元素，需要把它们放置到不同的目录，才能很好地管理文件。本项目目录结构如图 4-2 所示。

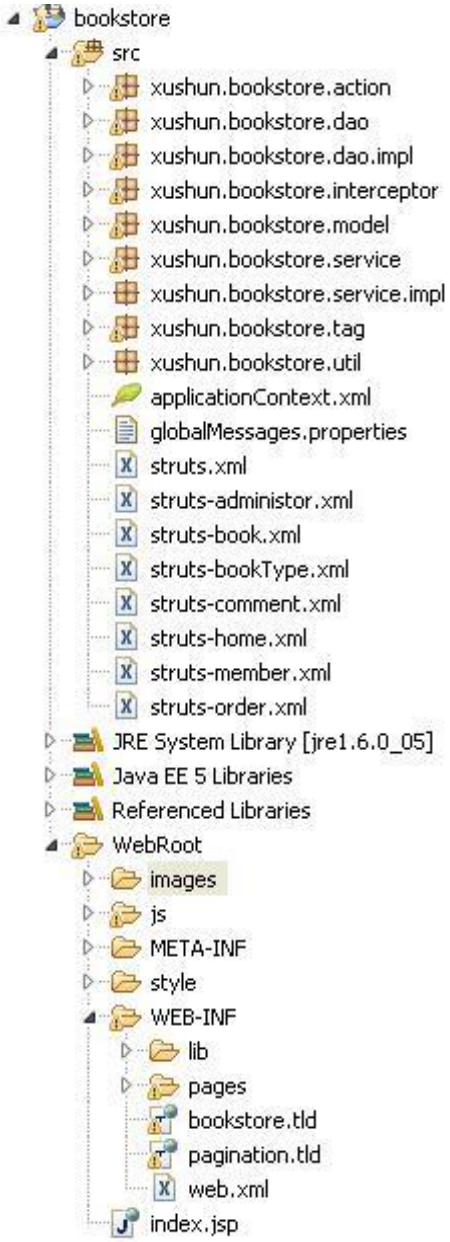

图 4-2 bookstore 项目目录结构图

对 java 类文件不同的作用进行归纳分配，可以更有效地管理 Java 类，按照其作用分包存放。该项目是一个严格分层架构的项目。对上面目录结构进行简约的解析如表 4-1 所示。

表 4-1 项目目录结构解析表

路径	说明
src	该目录下存放 Java 类文件
xushun.bookstore.action	控制层包，调用业务对象对请求处理，返回逻辑视图
xushun..bookstore..dao	定义 Dao 接口
xushun..bookstore..dao.impl	Dao 层实现包，包含对数据库进行操作的类
xushun..bookstore..model	JavaBean 包
xushun..bookstore.service	业务层接口包，定义业务对象操作的行为，但不实现
xushun..bookstore..service.impl	业务层实现包，调用 Dao 实现业务逻辑
xushun..bookstore..interceptor	拦截器包，用于权限验证的功能
xushun..bookstore..tag	自定义标签包，包括分页标签，时间显示标签等
xushun..bookstore..util	定义一些实用类
struts-*.xml	控制层类的配置文件，逻辑结果与实际视图的对应
webRoot	该目录下主要存放 JSP 页面、JS 文件、CSS 文件等
images	存放项目所需图片文件，包括图书封面原图，缩略图等
js	javascript 文件存放路径
style	CSS 文件存放路径
WEB-INF	
lib	第三方 jar 包存放路径
pages	JSP 页面存放路径(除 intex.jsp 外所有 jsp 都存放在此)
web.xml	部署文件

4.4 项目实现

4.4.1 实现原理

本项目是基于Struts2 + Spring + Hibernate架构实现的。

Struts2 框架是一个MVC框架，是对WEB层的解决方案，在多层体系结构中，Struts2与其他框架整合能更好的发挥作用，Struts2 通过一种可插拔式的插件，实现了与Spring框架的整合。本项目利用的整合方法是利用Spring插件的自动装配方式，当Spring插件创建Action实例后，立即将Spring容器中对应的业务逻辑组件注入到Action实例，这样Action就可以访问到Spring容器中的业务逻辑组件了。

Spring容器作为系统的IoC容器，将系统中业务逻辑组件放在Spring容器中管理，并且充分利用Spring IoC容器的功能，采用依赖注入来管理系统各组件的依赖关系，避免了各组件之间的硬编码的耦合，提高了系统的灵活性和可扩展性。

持久层解决方案借助于Hibernate 框架实现，借助Hibernate的功能，允许上层程序采用面向对象的方式编程，消除了面向对象语言与关系型数据库之间的不协调性。Hibernate负责把面向对象的持久化操作转化成对应的JDBC操作，但它对用户是完成透明的。Hibernate作为O/R Mapping框架，简化了数据库的访问，以面向对象的方式操作数据库，更加符合面向对象程序设计的思路。

系统严格采用Java EE的三层体系结构，分为表现层、业务逻辑层和数据逻辑层。三层

体系将业务规则、数据访问等工作放到中间层处理，客户端不直接与数据库交互，而是通过控制层与中间层建立连接，再由中间层与数据库交互。这样能够使表示层和业务逻辑层解耦，是系统更加明了，且有利于代码的重用。

表现层使用JSP来实现，JSP页面使用Struts 2标签来显示数据和进行逻辑判断，生成页面显示效果(JSP只显示数据，而不参加业务逻辑)。

DAO层的实现使用Spring的HibernateDaoSupport作为基类，继承它实现DAO更加方便，而无须管理Hibernate的SessionFactory，Session等，通过Spring的HibernateTemplate完成数据库操作。

事务采用Spring的声明式事务。通过声明式事务，无须将事务策略以硬编码的方式和代码耦合在一起，而是放在配置文件中声明。业务逻辑组件可以更加专注于业务的实现，从而简化开发。

为了分离控制层与业务逻辑层，又可细分为：

(1) Web层，就是MVC模式里面的“C”(Controller)，负责表现层与业务逻辑层的交互，调用业务逻辑层， 并将业务数据返回给表现层来显示。

(2) Service层(业务逻辑层)，负责实现业务逻辑， 对DAO对象进行封装(因为一个事务有可能跨越几个DAO对象)。

(3) DAO层(数据访问对象层)，负责与持久化对象交互，封装了数据的增、删、查、改等操作。

(4) PO层(持久化对象层)，通过实体/关系映射工具将关系型数据库的数据映射成对象，实现以面向对象方式操作数据库。

通过使用Struts2，提供了良好的MVC模式，所有用户的请求都由Struts 2的Action负责拦截，然后通过Spring提供的自动装配功能，定位Spring容器中的业务逻辑组件，调用业务逻辑组件方法来处理用户请求。

由于MySql方便功能又强大，采用MySQL存储数据。

用户的所有请求都由Struts2的FilterDispatcher 过滤，通过这种方式将系统的所有请求都转入Struts2系统内，从而保证所有请求都交给Struts2的Action处理。本项目把所有jsp页面都放入了WEB-INF目录下的pages文件夹里面，可以避免用户直接访问JSP页面，提高系统的安全性。

本系统采用AJAX(jQuery)技术，提高用户体验。

jQuery 是一个JavaScript 库，它有助于简化 JavaScript以及 Asynchronous JavaScript + XML (Ajax) 编程。与类似的 JavaScript 库不同，jQuery 具有独特的基本原理，可以简洁地表示常见的复杂代码。

4.4.2 项目架构思路

本项目是基于Struts2 + Spring + Hibernate四层架构实现的。

数据逻辑层用Hibernate实现。由于 Java™ 5 泛型的采用，有关泛型类型安全 Data Access Object (DAO) 实现的想法变得切实可行。对于大多数开发人员，为系统中的每个

DAO 编写几乎相同的代码到目前为止已经成为一种习惯。可以使用许多 ORM 工具来避免代码重复。例如，使用 Hibernate，您可以简单地为所有的持久域对象直接使用会话操作。这种方法的缺点是损失了类型安全。

系统中的所有数据库访问都通过 DAO 进行以实现封装。每个 DAO 实例负责一个主要域对象或实体。如果域对象具有独立生命周期，它应具有自己的 DAO。DAO 负责域对象的创建、读取（按主键）、更新和删除（creations, reads, updates, and deletions，CRUD）。

DAO 不负责处理事务、会话或连接。这些不由 DAO 处理是为了实现灵活性。

自 Java 语言中出现泛型以来，单个泛型类型安全 DAO 的概念已经成为主题。使用泛型机制编写的程序代码要比那些杂乱的使用Object变量，然后再进行强制类型转换的代码具有更好的安全性和可读性。

本项目中HibernateGenericDao继承来自Spring框架的HibernateDaoSupport。实现对于每个实体最基本的添加、查找、修改和删除(CURD)等操作。其中还提高了一个非常重要的方法，分页查询。分页查询是网页设计经常需要实现的基本功能。但有的分页直接嵌在jsp页面上，不仅工作量较大，代码也难以重用。本项目将分页查找写入泛型DAO来实现通用的分页查找，其子类只需提供其hql语句，其余工作在基类中完成，返回分页对象。将通用的代码写入基类，体现了Java中面向对象和代码重用的思想。

本项目四层架构中，事务不是放在Dao层，而是放入业务逻辑层，因为一个事务可以跨越多个Dao操作，为了保持事务的一致性和原子性等特点，在业务逻辑层配置事务是很合理的。Spring提高了编程式事务和声明式事务，编程式事务硬编码到代码中，不利于维护，声明式事务只需简单的配置即可，本想使用Spring的声明式事务。

利用Struts2实现控制层和表现层。所有的Action继承BaseAction，BaseAction继承Struts2的ActionSupport。BaseAction中提供了注入的业务逻辑对象，和分页相关的属性，以利于共赏。Struts2的参数拦截器把提交给Action的参数解析出来，赋给其Action中对应的属性，而无需通过大量的HttpServletRequest.getParamter()来获取其值，然后进行转换，异常处理。Struts2简化了其基本操作。

表现层通过Struts2提供的大量标签来操作。自定义了分页标签，分页标签可以很好的实现分页，具有重用性。

数据库中用整数存储时间，整数不丢失其精度，最后通过自定义标签来解析日期整数，把其转换成相应的值。

本项目中应用AJAX技术提高用户体验。

4.4.3 部分核心代码展示

分页，是一种将所有数据分段展示给用户的技术.用户每次看到的不是全部数据，而是其中的一部分，如果在其中没有找到自习自己想要的内容，用户可以通过制定页码或是翻页的方式转换可见内容，直到找到自己想要的内容为止.其实这和我们阅读书籍很类似. 在数据库中操作最多的行为是查询，对于数据量较多的记录，还需进行分页显示，就得分页查询。分页涉及的对象好多，如图书需分页，订单需分页。

(1) 分页对象：存储分页信息。

```
/**
 * 分页对象. 包含当前页数据及分页信息如总记录数.
* @author xushun
 */
@SuppressWarnings("serial")
public class DataPage<T> implements Serializable {
    private static int DEFAULT_PAGE_SIZE = 10;
    private int pageSize = DEFAULT_PAGE_SIZE; // 每页的记录数
    private int start; // 当前页第一条数据在 List 中的位置，从 0 开始
    private List<T> data; // 当前页中存放的记录，类型一般为 List
    private int totalCount; // 总记录数
/**
 * 构造方法，只构造空页.
 */
public DataPage() {
    this(0， 0， DEFAULT_PAGE_SIZE， new ArrayList<T>());
}

/**
 * 默认构造方法.
* @param start 本页数据在数据库中的起始位置
 * @param totalSize 数据库中总记录条数
 * @param pageSize 本页容量
 * @param data 本页包含的数据
 */
public DataPage(int start， int totalSize， int pageSize， List<T> data) {
    this.pageSize = pageSize;
    this.start = start;
    this.totalCount = totalSize;
    this.data = data;
}

/**
 * 取总页数.
 */
public int getTotalPageCount() {
    if (totalCount % pageSize == 0)
        return totalCount / pageSize;
```

```
        else
            return totalCount / pageSize + 1;
    }

    /**
     * 取该页当前页码，页码从 1 开始.
     */
    public int getCurrentPageNo() {
        return start / pageSize + 1;
    }
    /**
     * 该页是否有下一页.
     */
    public boolean hasNextPage() {
        return this.getCurrentPageNo() < this.getTotalPageCount() ;
    }

    /**
     * 该页是否有上一页.
     */
    public boolean hasPreviousPage() {
        return this.getCurrentPageNo() > 1;
    }

    /**
     * 获取任一页第一条数据在数据集的位置，每页条数使用默认值.
    * @see #getStartOfPage(int，int)
     */
    protected static int getStartOfPage(int pageNo) {
        return getStartOfPage(pageNo，  DEFAULT_PAGE_SIZE);
    }

    /**
     * 获取任一页第一条数据在数据集的位置.
     * @param pageNo     从 1 开始的页号
     * @param pageSize  每页记录条数
     * @return  该页第一条数据
     */
    public static int getStartOfPage(int pageNo，  int pageSize) {
        return (pageNo - 1) * pageSize;
```

```
}
// 部分 getter setter 方法省略。
}
```

(2) 分页查询：基于 Hibernate 的分页查询

此项目中数据库 Dao 层是基于泛型 Dao 的，泛型 Dao 可以把一些共同操作写入泛型类中，参数化可以使代码更容易理解和重用。泛型 Dao(HibernateGenericDao)继承 Spring 提供的 HibernateDaoSupport。下面提供用于分页查询的核心代码。

```
/**
 * Hibernate Dao的泛型基类. <p/> 继承于Spring的<code>HibernateDaoSupport</code>
 * 提供分页方法和若干便捷查询方法，并对返回值作了泛型类型转换.
 * @author xushun
 * @see HibernateDaoSupport
 */

@SuppressWarnings("unchecked")
public class HibernateGenericDao<T> extends HibernateDaoSupport implements BaseDao<T>
{
    protected Class<T> entityClass;// DAO 所管理的 Entity 类型.
    public HibernateGenericDao() {
entityClass = GenericsUtils.getSuperClassGenricType(getClass());
}
/**
 * 创建Query对象.
 * @param values 可变参数.
 */
public Query createQuery(String hql，  Object... values) {
    Assert.hasText(hql);
    Query query = getSession().createQuery(hql);
    for (int i = 0; i < values.length; i++) {
        query.setParameter(i，  values[i]);
    }
    return query;
}
/**
 * 分页查询函数，使用hql.
 * @param pageNo 页号，从1开始.
 * @param pageSize 每页记录数
 */
```

```
public DataPage<T> pagedQuery(String hql，  Integer pageNo，  Integer pageSize，Object...
values) {
    // 是否设置分页号，每页的分页数
    if (pageNo == null || pageSize == null) {
        List list = createQuery(hql，  values).list();
        // 如果查询结果为空，则返会没有数据内容的分页，否则返回全部结果
        if (list == null || list.size() == 0) {
            return new DataPage();
        } else {
            return new DataPage(0，  list.size()，  list.size()，  list);
        }
    }
    Assert.hasText(hql);
    Assert.isTrue(pageNo >= 1，  "pageNo should start from 1");
    // Count查询 ，查询符合条件的总记录数
    String countQueryString = " select count(*) "+ removeSelect(removeOrders(hql));
    List countlist = getHibernateTemplate().find(countQueryString，  values);
    int totalCount = ((Long) countlist.get(0)).intValue();
    if (totalCount < 1)
        return new DataPage();
    // 实际查询返回分页对象
    int startIndex = DataPage.getStartOfPage(pageNo，  pageSize);
    Query query = createQuery(hql，  values);
    List list = query.setFirstResult(startIndex).setMaxResults(pageSize).list();
    return new DataPage(startIndex，  totalCount，  pageSize，  list);
}
private static String removeSelect(String hql) {
    Assert.hasText(hql);
    int beginPos = hql.toLowerCase().indexOf("from");
    Assert.isTrue(beginPos != -1，  " hql ：  " + hql + " must has a keyword 'from'");
    return hql.substring(beginPos).replace("fetch"，  "");
}
}
```

项目中存在AJAX应用。AJAX全称为“Asynchronous JavaScript and XML”（异步JavaScript和XML），是指一种创建交互式网页应用的网页开发技术。AJAX的最大机遇在于用户体验。在使应用更快响应和创新的过程中，定义Web应用的规则正在被重写;因此开发人员必须更注重用户。现在用户已经逐渐习惯 如何使用Web应用了。例如用户通常希望每一次按钮点击会导致几秒的延迟和屏幕刷新，但AJAX正在打破这种长时间的状况。因

此用户需要重新体验按钮点击 的响应了。

AJAX由HTML，JavaScript，DHTML和DOM组成，它使浏览器可以为用户提供更为自然的浏览体验。这一方法把WEB界面转化成交互性的 Ajax应用程序。AJAX提供与服务器异步通信的能力，使用户从请求响应的循环中解脱出来。借助AJAX可以在用户点击按钮时，使用 JavaScript和DHTML立即更新页面，并向服务器发出异步请求，以持行更新或查询数据库。当请求返回时，就可以使用JavaScript和CSS来相应的更新UI，而不是刷新整个页面，使得用户对WEB站点拥有即时响应体验。

本项目应用AJAX是基于jQuery库。

(3) AJAX异步修改购物车数量

客户端JavaScript代码如下：

```
function modifyCartItemCount(obj) {
    var bookId = $(obj).attr("bookId");
    var count = $(obj).attr("value");
    if(count <= 0) {
        alert("数量不能小于或等于0");
        $(obj).attr("value"， $(obj).attr("currentCount"));
        return false;
    }
    $.post(
        "shopping_modifyCartItemCount.action"，
        {"bookId"： bookId， "count"： count}，
        function(data) {
            $(obj).attr("currentCount"， count)
            $("#amount").text(data.amount);
        }，
        "json"
    );
}
```

服务器端代码如下：

```
/**
 * 修改订单项数量
*/
@SuppressWarnings("unchecked")
public String modifyCartItemCount() throws Exception {
    if(count < 0)
        return SUCCESS;
    Map session = ActionContext.getContext().getSession();
```

```
    orderDetails = (List<OrderDetail>)session.get("orderDetails");

    if(orderDetails != null) {
        for(OrderDetail orderDetail：orderDetails) {
            if(bookId == orderDetail.getBookId()) {
                orderDetail.setCount(count);
                break;
            }
        }
    }
    session.put("orderDetails"，  orderDetails);
    getOrderAmount();
    return SUCCESS;
}
```

4.5 系统特点

本系统的开发基于 Java 平台，设计采用基于 Struts2+Spring+Hibernate 的多层体系结构，实现了 MVC 设计模式，使用 Struts2 实现表示层和控制层，Spring 实现业务逻辑层，Hibernate 实现持久层。以下从移植性、可维护性和系统性能几个方面来总结本系统的特点:

(1).移植性

在跨平台方面，由于 Java 语言本身的平台无关性及 JavaEE 标准的平台无关性，本系统可以在不同的操作系统之间进行切换。在数据库方面，由于数据库之间的差异，一般的采用 JDBC 编码实现数据持久化的系统在数据库平台之间的迁移将会遇到阻力，必需重新编写相关的 JDBC 代码以支持新的数据库平台。由于 Hibernate 设计上良好的隔离，提供了对不同数据库的良好支持，本系统只需要简单地修改数据库配置参数，即可实现底层数据库的切换。

(2).可维护性

系统采用基于 Struts2+Spring+Hibernate 的多层结构设计，使用 MVC 模式分离了表示层和业务层，隐藏了业务逻辑，使得两层间松散祸合，各自的修改不影响对方，提高了可维护性。Spring 框架的使用将 JavaEE 层次结构中的业务层分离为业务逻辑层和数据持久层，这样业务逻辑便交给 Spring 处理，而数据访问则交给 Hibernate 处理，使得层次结构更加清晰，便于系统的维护和扩展。

(3).系统性能

持久层框架 Hibenrate 提供了优秀的性能优化机制，如内置的数据库连接池支持、PreparedStatement 缓存、数据缓存等，这些优化机制的综合使用大大提升了系统性能。

在代码级别上，Hibernate 的性能比普通 Java 程序员写的 JDBC 代码性能高很多。原因在于 Hibernate 本质上还是包装了 JDBC 来进行数据库操作的，由于 Hibernate 在调用 JDBC 上面花了很大力气，尽可能的使用最优化、最高效的 JDBC 调用，所以在性能上表现相当令人满意。

第 5 章 界面设计

5.1 XHTM 和 CSS 简介

XHTML是The Extensible HyperText Markup Language(可扩展超文本标识语言)的缩写。HTML是一种基本的WEB网页设计语言，XHTML是一个基于XML的置标语言，看起来与HTML有些相象，只有一些小的但重要的区别，XHTML就是一个扮演着类似HTML的角色的XML，所以，本质上说，XHTML是一个过渡技术，结合了部分XML的强大功能及大多数HTML的简单特性。

2000年底，国际W3C组织(World Wide Web Consortium)组织公布发行了XHTML 1.0版本。XHTML 1.0是一种在HTML 4.0基础上优化和改进的的新语言，目的是基于XML应用。XHTML是一种增强了的HTML,它的可扩展性和灵活性将适应未来网络应用更多的需求。XML虽然数据转换能力强大，完全可以替代HTML，但面对成千上万已有的基于HTML语言设计的网站，直接采用XML还为时过早。因此，在HTML4.0的基础上，用XML的规则对其进行扩展，得到了XHTML。所以，建立XHTML的目的就是实现HTML向XML的过渡。目前国际上在网站设计中推崇的WEB标准就是基于XHTML的应用（即通常所说的XHTML＋DIV）。

CSS是Cascading Style Sheets（层叠样式表单）的简称。更多的人把它称作样式表。顾名思义，它是一种设计网页样式的工具。借助CSS的强大功能，网页将在您丰富的想象力下千变万化。

HTML 标签原本被设计为用于定义文档内容。通过使用 <h1>、<p>、<table> 这样的标签，HTML 的初衷是表达“这是标题”、“这是段落”、“这是表格”之类的信息。同时文档布局由浏览器来完成，而不使用任何的格式化标签。

由于两种主要的浏览器（Netscape 和 Internet Explorer）不断地将新的 HTML 标签和属性（比如字体标签和颜色属性）添加到 HTML 规范中，创建文档内容清晰地独立于文档表现层的站点变得越来越困难。

为了解决这个问题，万维网联盟（W3C），这个非营利的标准化联盟，肩负起了 HTML 标准化的使命，并在 HTML 4.0 之外创造出样式（Style）。

所有的主流浏览器均支持层叠样式表。

样式表定义如何显示 HTML 元素，就像 HTML 3.2 的字体标签和颜色属性所起的作用那样。样式通常保存在外部的 .css 文件中。通过仅仅编辑一个简单的 CSS 文档，外部样式表使你有能力同时改变站点中所有页面的布局和外观。

由于允许同时控制多重页面的样式和布局，CSS 可以称得上 WEB 设计领域的一个突破。作为网站开发者，你能够为每个 HTML 元素定义样式，并将之应用于你希望的任意多的页面中。如需进行全局的更新，只需简单地改变样式，然后网站中的所有元素均会自动地更新。

5.1.1 XHTML 和 CSS 布局优点

一般我们使用FrontPage或者DreamWeaver，或者其他可视化工具来制作网页。用这种

方法制作网页，一般都是用表格来进行排版，虽然操作方便，但是网页中存在大量的垃圾代码。这些垃圾代码的存在，使得网页变得很臃肿，用户在浏览这种网页时，需要下载很多无用的代码，如果时间过长，会对该网页失去兴趣。对于搜索引擎来说，网页中的有效文字内容所占比例减小，搜索引擎需要花更多的资源来分析，如果网页中的代码过于杂乱，搜索引擎可能会对这样的网页产生反感，导致该网页在搜索引擎中的表现不好。

采用WEB标准，也就是使用XHTML+CSS技术来制作网页，最大的优点是将网页代码和格式彻底分离，格式代码存放于一个独立的文件中，保证了网页代码的干净、整洁。用XHTML+CSS技术来制作网页，减少了网页中格式代码，网页变得简洁，网页中有效文字的比率大大增加。

当用户浏览这种网页时，由于下载内容减少，网页可以更快的显示于浏览器中。对于搜索引擎来说，有利于爬虫的抓取，有利于搜索引擎对该网页进行分析。使用XHTML+CSS技术制作的网站，无论对于用户的浏览感受，还是对于搜索引擎的优化，都具有很大的优点。

对于习惯于用表格排版的人来说，会觉得用XHTML+CSS技术制作网站非常难。这是一种思维定式造成的。用层来定位，比用表格来定位，从实现方法上来说，要简单得多。用XHTML+CSS技术，可以完全实现用表格所能实现的内容。

用XHTML+CSS技术来制作网站，对于搜索引擎优化还有一个巨大的优点。用表格定位，各个部分在网页文件中出现的位置，都是和显示顺序时一致的。一般都是先顶部、其次左边、然后右边、最后顶部。这种自然循序，会导致网页中的重要内容并非出现在网页的开始部分。而用XHTML+CSS技术，可以改变这一顺序，在同样的显示格式下，可以使重要的内容首先出现，提高了重要内容在网页中的权重。在以后的文章中，yoo会详细介绍这个问题。

对于搜索引擎优化人员来说，用XHTML+CSS技术制作网页，是非常必要的。

由于各种浏览器对WEB标准的支持不同，尤其是大家普遍使用的IE浏览器，对于WEB标准支持得并不好，因此用这种方式制作网页，可能在各个浏览器的显示略有差别。在制作的时候，yoo建议最少使用两种不同的浏览器进行测试，IE、Firefox、Opera等，以避免网页在某些浏览器里严重变形，影响用户的浏览感受。

5.2 界面设计

本系统所有页面布局是基于XHTML和CSS的，结构和表现相分离，便于维护。本系统采用的大部分布局如图5-1所示：

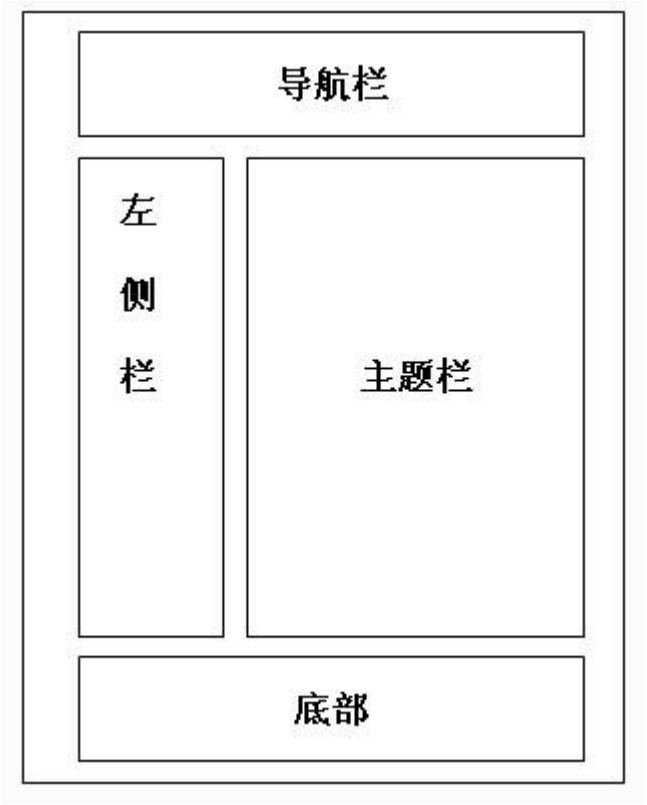

图5-1 基本布局

导航栏包括logo，用户登陆注册链接，购物车信息。新书、推荐和销售排行等信息，还有图书搜索框，以便用户能快速找到自己想要的图书。

针对用户的左侧栏是图书的分类信息，针对管理员的左侧栏是管理导航栏，包括用户管理、图书管理、订单管理等。

主题栏是内容信息，底部是一些网站基本信息。

头部、底部、左侧栏是可重用的。布局简洁，优美。

5.3 运行结果

本系统所有页面布局是基于XHTML和CSS的，结构和表现相分离，便于维护。

(1) 网上书店是电子商务系统，电子商务系统首页的设计至关重要，用户进来第一眼看见的是主页，主页设置是否友好影响着用户的第一感觉，所以主页设置的必须友好，信息量多，易于操作等。进入网上书店系统主页：系统主页显示其新书展示、经典推荐、热销排行和分类查询等。

图5-2 网上书店主页

(2) 用户注册。在对网上购物时，每个人都需有一个唯一的账号，以便进行购物。用户信息包括用户的真名、地址和电话等。

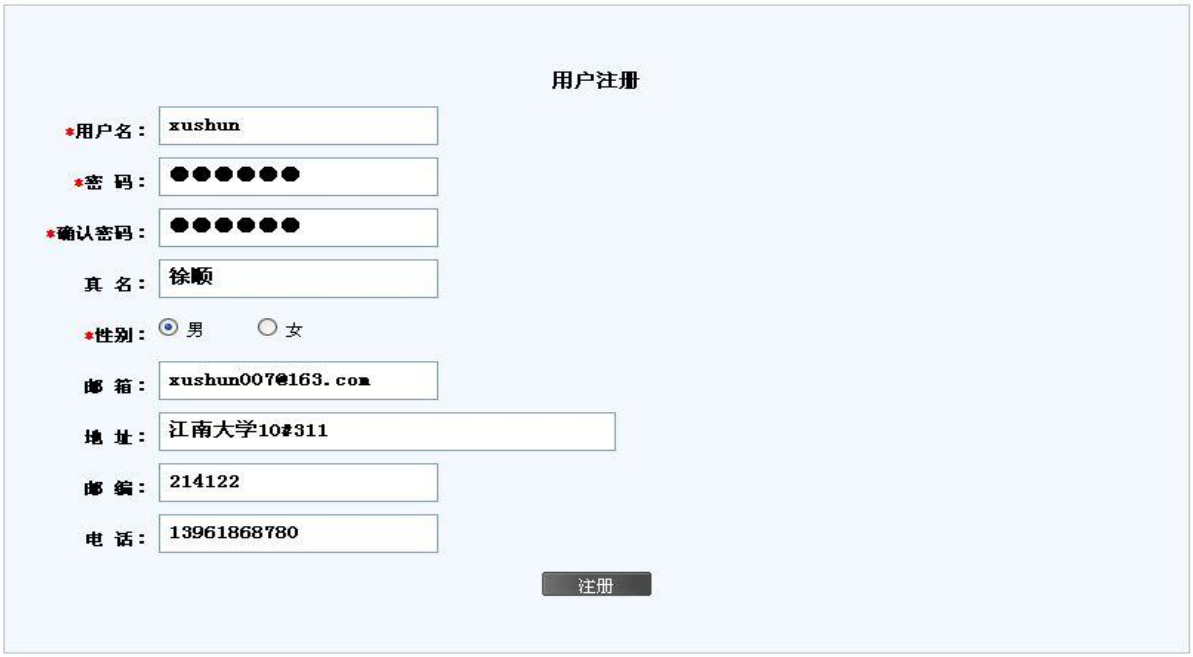

图5-3 用户注册

(3) 搜索图书。用户可以通过关键字匹配快速搜索到自己所要寻找的图书。

图 5-4 图书搜索

(4) 查看图书。用户可以查看图书的具体信息及其评论。

图 5-5 图书信息查看

(5) 购物车。购物车图书信息，包括所购图书及其数量。

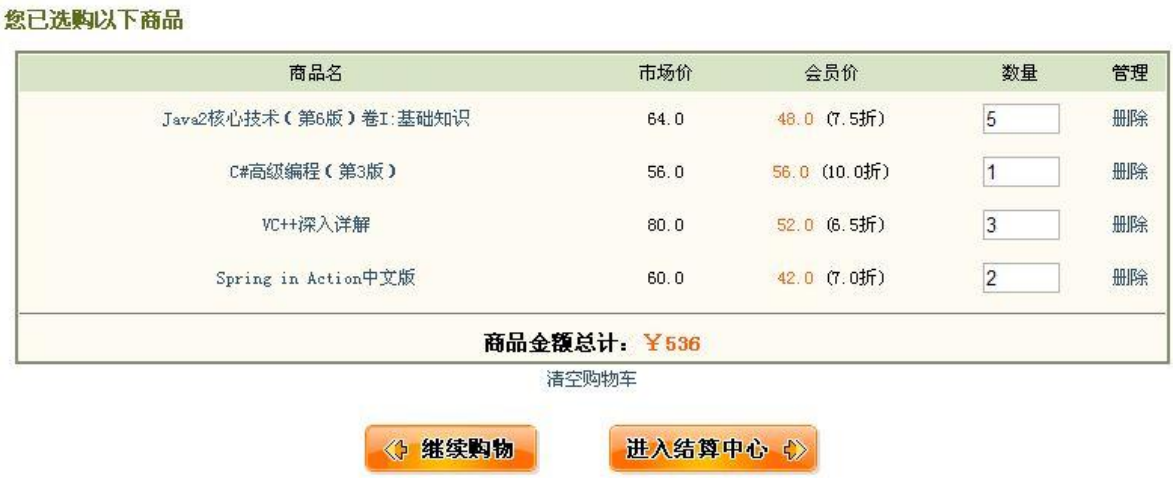

图 5-6 购物车

(6) 提交订单。当用户已经选定已经满意的商品时，即可提交订单。

图 5-7 提交订单

(7) 管理员界面

图 5-8 管理员界面

(8) 添加图书

图 5-9 添加图书

(9) 管理员查看订单。管理员可以对用户订单进行处理(如删除，更改状态等)。

订单号	收货人	订单总金额	下单时间	处理时间	管理
8	徐顺	￥59.25	2009-05-20 13:29:09	2009-05-20 16:15:14	编辑 删除
9	徐顺	￥470.0	2009-05-20 16:12:48	尚未处理	编辑 删除
10	shopping	￥57.0	2009-05-20 16:17:09	2009-05-20 16:17:59	编辑 删除
11	shopping	￥171.0	2009-05-20 16:17:21	2009-05-20 16:19:30	编辑 删除
12	徐顺	￥224.0	2009-05-20 16:18:49	尚未处理	编辑 删除

1 | 2 | 下一页 | 最后页

图 5-10 管理员查看订单

第 6 章 结论与展望

6.1 结论

本文给出了一种基于Struts2+Spring+Hibernate多层结构的Web应用系统框架模型，通过对框架模型主要层次的功能分析和相关开发技术的比较，提供了一种实现多层Web应用系统开发的解决方案。这个系统是基于B/S架构开发的网上书店系统，在程序源代码上实现了模块化，使得每一个功能既能相互独立有能相互关联，方便了日后的维护以及修改。使用此系统，可以大大地方便用户购买图书，真正做到足不出户就能买到自己需要的图书。

6.2 不足之处及未来展望

完成这次设计任务总共用了八个星期时间，前三个星期用来收集资料、学习要用到的各项开发技术、进行网上书店系统的分析，中间两个星期用来设计系统，后一个星期用来测试及修改。论文的撰写一直贯穿其中。

为本系统总共设计 38 个 JSP 网页，一万一千多行代码，8 个数据表。通过这次毕业设计，我从中学到了许多新的知识，而且通过这次毕业设计，培养了我综合多门学科中的知识、迅速规划并开发出目标系统的能力，以及编程能力也有了很大的提高。另外也有许多心得体会，所谓系统开发如人生百味，酸甜苦辣皆有之。

严格按软件工程的方法来设计系统相当重要，需要充分利用时间来做系统分析，再进入系统编码阶段。若想保证质量，把系统做得更可靠更有效率功能越强，应该考虑的方方面面就越多越复杂。系统分析过于简单，系统定义过于抽象，则在系统设计与编码阶段遇到的困难就越多，特别是其中若不得不做一些功能性甚至系统结构性方面的变动，将面对许多重复性的工作。在系统开发过程中重复工作过多，将会极大地影响系统开发的积极性，进而影响整个系统的质量。在这一点上，我体会尤深，我花了三个星期，即接近系统开发一半的时间用在系统分析与系统定义上，也就是在开始浪费了一周多的时间，即边编码边分析，边分析边编码，后来越来越乱越来越复杂，不得不重新考虑系统开发计划的合理性。总之，在这一点上，我的体会是，系统分析越充分，系统定义越具体，那么后续的系统设计与开发工作就越有效率，且系统的质量也越有保障。

本系统是一个最基本的基于 WEB 的网上书店系统，可扩展性很大，科学的开发过程也极有利于系统的扩充与扩展。系统现在采用的是 MySQL 数据库，视需要可以移植到 MS SQL Server 或者其它大中型数据库系统环境下，只需改动 Hibernate 配置文件。系统的分析与定义都结合了现在流行的面向对象方法以及传统的结构分析与设计方法。

回顾这一个半月的系统开发工作，总结起来那就是，软件的开发是相当辛苦的，但成功以后的喜悦也是非常美妙的，而且从中发现，其中投入的心血越多，成功以后你所获得的快乐与充实感也更多更强！

参考文献

[1] 李刚.Struts2 权威指南——基于 WebWork 核心的 MVC 开发[M].北京：电子工业出版社，2007

[2] 林信良. Spring2.0 技术手册.北京：电子工业出版社[M]，2008

[3] 夏昕，曹晓钢，唐勇. 深入浅出 Hibernate[M].北京：电子工业出版社，2007

[4] Bruce Eckel. Java 编程思想(第四版) [M].北京：机械工业出版社，2007

[5] Joshua Bloch. Effective Java[M].北京：机械工业出版社，2003

[6] Cay S.Horstmann .Java 核心技术 卷 1[M].北京：机械工业出版社，2007

[7] Craig Walls，Ryan Breidenbach. Spring in Action. [M]，Manning Publications，2005

[8] Christian Bauer, Gavin King，Hibernate in Action[M]，Manning Publications，2004

[9] （美）弗里曼，Head First 设计模式[M]，中国电力出版社，2007

[10]（美）梅特斯克，（美）韦克，Java 设计模式[M]，人民邮电出版社，2007

[11] 克拉恩，帕斯卡雷洛，杰姆斯. AJAX 实战[M] . 人民邮电出版社. 2006.

致　谢

在论文的最后，我要向所有帮助过我的人致以诚挚的感谢!

首先向我的导师蒋敏副教授致以最诚挚的谢意。她严谨的治学态度、精益求精的工作作风和诲人不倦的为师风范，让我受益匪浅，并将激励我在今后的学习和工作中不断进取。在整个学习阶段都得到了蒋老师细心的指导和热情的关怀，在论文的写作过程中，她一直对我悉心指导、严格要求，并在百忙之中为我审阅论文。在此谨向蒋老师表示衷心的感谢和真诚的敬意。

感谢与我朝夕相处的朋友、同学，四年的学习生活离不开他们的关心与帮助。

最后，向所有帮助我的人表示衷心的感谢，并诚挚地感谢为评阅本论文而付出辛勤劳动的各位老师！