

仲恺农业工程学院

# 毕 业 设 计

可扩展的程序在线评测系统研究与设计

姓 名 夏集球

院（系） 信息科学与技术学院

专业班级 计算机科学与技术 092

学 号 200910214219

指导教师 顾春琴

职 称 讲师（博士）

论文答辩日期 2013 年 5 月 20 日

仲恺农业工程学院教务处制

## 学生承诺书

本人郑重声明：所呈交的毕业论文《可扩展的程序在线评测系统研究与设计》，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学生签名：\_\_\_\_\_

日 期： 年 月 日

## 摘 要

程序语言课程是计算机相关专业的核心教学内容之一，要提高程序语言设计能力必须通过大量的实践练习与交流。在传统的学习过程中，往往通过人工方式对程序的源代码进行评测，遇到问题，不易于学习交流。本系统目的便是构建一个基于 B/S 的，能够方便扩展系统功能的在线评测系统。同时在基础框架之上扩展出几个模块，以增强程序学习者之间的学习交流，提供教学辅助，简化程序的评测过程，提高程序学习的效率。

本系统结合当前优秀的开源框架，构建一个相对易于扩展的 web 在线评测系统。系统总体框架实现和技术主要为 Spring, Hibernate, MySQL, Lucene 等主流技术。采用 Spring MVC 来控制主要的业务逻辑，Hibernate 实现数据的 ORM 映射，MySQL 实现系统数据的存储。前端使用 JSP 进行显示，用 jQuery 进行行为的控制，前后端之间 Ajax 传输 JSON 格式数据。

本论文阐释了如何可扩展系统，jQuery 插件开发，Ajax 服务的构建基于角色的权限控制等适应于本系统的基础应用。最后在这些技术的基础之上扩展了三个模块，分别是，基础管理模块，在线评测模块和论坛模块。

**关键字：**可扩展 SpringMVC Hibernate Ajax RBAC

# 目 录

1 前 言.....	1
1.1 研究目的.....	1
1.2 研究意义.....	1
1.2.1 用户在线学习程序设计语言.....	2
1.2.2 教学辅助.....	2
1.2.3 学员间交流.....	2
1.2.4 基于角色的权限控制.....	2
1.3 国内外研究概况.....	2
1.3.1 国内研究现状.....	3
1.3.2 国外研究现状.....	3
2 需求与可行性分析.....	4
2.1 项目必要性.....	4
2.2 系统需求.....	5
2.2.1 外部行为需求.....	5
2.2.2 内部特性需求.....	6
2.3 项目可行性.....	7
2.3.1 经济可行性.....	7
2.3.2 技术可行性.....	7
2.3.3 法律可行性.....	7
2.4 系统开发环境.....	7
2.4.1 客户端软件.....	7
2.4.2 服务端软件.....	7
2.4.3 开发工具.....	8
3 相关技术简介.....	8
3.1 Spring.....	8
3.1.1 简介.....	8
3.1.2 应用场景.....	10
3.1.3 价值.....	11

3.2	Hibernate.....	11
3.3	jQuery.....	12
3.4	Ajax.....	12
4	基础架构设计与实现.....	13
4.1	可扩展架构设计与实现.....	13
4.1.1	可扩展架构设计.....	13
4.1.2	可扩展架构实现.....	17
4.1.3	架构的不足.....	19
4.2	RBAC 权限模型设计与实现.....	19
4.2.1	web 中的权限模型概述.....	21
4.2.2	领域模型分析.....	22
4.2.3	基础数据操作.....	24
4.2.4	RBAC 实现.....	26
4.3	JSONLIB 设计与实现.....	28
4.3.1	背景.....	28
4.3.2	JSONLIB 设计.....	28
4.4	Ajax 基础服务设计与实现.....	31
4.4.1	背景.....	31
4.4.2	Ajax 服务设计.....	31
4.5	日志服务设计与实现.....	34
4.6	数据访问设计与实现.....	35
4.7	分页插件设计与实现.....	37
4.7.1	分页概述.....	37
4.7.2	关键算法实现.....	39
5	模块分析.....	41
5.1	模块划分.....	41
5.2	模块功能.....	41
5.2.1	基础应用扩展模块.....	41
5.2.2	在线评测扩展模块.....	42

5.2.3 在线论坛扩展模块.....	43
5.3 领域模型分析.....	43
5.3.1 系统参与者.....	43
5.3.2 参与者权限分配.....	44
6 模块详细设计与实现.....	46
6.1 系统体系架构.....	46
6.1.1 系统体系架构.....	46
6.1.2 职责分配.....	47
6.1.3 结合 SSH 的 MVC 模型.....	48
6.1.4 SSH 工作流程说明.....	50
6.2 基础应用扩展模块.....	50
6.2.1 领域模型分析.....	50
6.2.2 视图与控制层设计.....	51
6.2.3 业务组件设计.....	53
6.3 在线测评扩展模块.....	53
6.3.1 领域模型分析.....	53
6.3.2 视图与控制层设计.....	54
6.3.3 业务组件设计.....	56
6.4 在线论坛扩展模块.....	56
6.4.1 领域模型分析.....	56
6.4.2 视图与控制层设计.....	57
6.4.3 业务组件设计.....	58
7 系统测试.....	59
7.1 基础应用模块测试.....	60
7.2 在线评测模块测试.....	64
7.3 在线论坛模块测试.....	67
8 总结.....	76
参 考 文 献.....	77
Abstract.....	78

致	谢.....	79
---	--------	----

# 1 前 言

## 1.1 研究目的

程序语言课程是计算机相关专业的核心教学内容之一，要提高程序语言设计能力必须通过大量的实践练习和交流，在传统的教学过程中，往往通过人工方式对程序的源代码进行评测，这种人工评测方式存在效率不高，容易误判并且缺乏一个可以共套交流的平台。本毕业设计课题提出一个基于 B / S 模式的可扩展在线程序语言评测系统，对系统的设计和主要的实现技术进行分析和探讨。

程序设计类课程，具有实践性强的特点。它不但要求学生掌握基础的理论知识，更重要的是要求学生不断提高自身的编程实践能力。因此，在这类课程中，老师势必会布置大量的程序设计作业，传统通过电子邮件或手写提交作业的方式让老师感到批改作业的任务繁重，让学生感到费时费力，难以提高编程的实践能力。为方便教学，提高学生动手编程的实践能力，开发一个自动化，智能化的评测系统成为需要。随着现代信息化的发展，软硬件的不断升级，使得开发这样一个评测系统成为可能。

Online Judge System（以后简称 OJS），则是指一个在线的裁判系统，它可对程序源代码在线进行编译和执行，并通过预先设计的测试数据来检验程序源代码的正确性。

现阶段已经拥有许多优秀的程序语言在线测评系统，许多是随着 ACM 而生的。例如北京大学在线测评系统是为训练国际大学生程序设计竞赛北大代表队而设计开发的，在训练的过程中发现，该系统在大学生程序设计类课程的教学也能发挥重要作用于是将其逐步引入相关课程的教学，革新了程序设计类课程的教学手段和考试办法，提高了学生的学习积极性，同时也在一定程度上抑制了学生在编写程序时常见的抄袭现象。

## 1.2 研究意义

EPOJS（Extensible Program Online Judge System）是一个学习的好帮手，开发这样一个系统是具有一定的实际意义的，下面分点阐述。



### 1.2.1 用户在线学习程序设计语言

无论是什么样的用户，只要想学习 C 程序设计语言，就可以在本系统上面实现在线编辑程序，执行程序并查看执行结果，通过测试的情况反馈使用者的学习情况。

### 1.2.2 教学辅助

在在线模块中，主要角色有教师和学生，对于学生，可以参加具体的某个教师的课程，教师会在相应的课程上面进行作业的布置，学生就可以在线进行做作业并能立刻看到结果，系统将会记录相关的解题情况，尝试次数等。老师变可以通过这些做题的情况来了解同学们的学习情况，以便有针对性的对学生进行授课，提高教学质量。

### 1.2.3 学员间交流

现有的 OJ 系统，没有论坛系统功能，学员之间不利于沟通。因此，本系统也扩展了论坛系统功能，便于用户在论坛上面进行沟通。

### 1.2.4 基于角色的权限控制

使用 RBAC 来进行权限的管理与控制，使得权限分配更加集中和灵活。设计这样的权限控制模型，主要是利于扩展。对于最原始的系统来说，也许只有一种角色类型，就是普通用户。但是对于特定的模块，可能就需要新的角色，如在在线评测模块中，为了对教学进行支持，定义了学生和教师的角色。

## 1.3 国内外研究概况

在线测评系统发展到现在已经比较常见了，无论是其体系结构还是运行模式都已经相对成熟了。现行的许多大学校园中，都拥有自己的在线测评系统，但大都具有类似的工作模式，主要体现在以下几个方面：

第一方面，由于各大高校越来越重视 ACM 竞赛，很多高校变进行了各自的程序语言在线测评系统的开发和应用。主要目的就是要提供一个平台供学生们提

高程序设计水平。

第二方面，许多的程序语言在线测评系统并没有大胆的和教学结合在一起（当然也有，比如北京大学在线测评系统），应用面不够广。

ACM/ICPC 带动了演算算法程序设计的风气，世界上许多大专院校的咨询系所，仿照 ACM/ICPC 的比赛模式，纷纷自行开发出即时线上比赛系统，能够自动批改、评分、计时、统计。学生不必齐聚一堂，就可以相互切磋程式设计技巧。比赛结束之后，便将比赛题目列为题库，并开放线上批改程序的功能，供学生赛后练习。这样的系统大家一般称之为「Online Judge System」，或直接称为「Online Judge (OJ)」。

下面就国内和国外的 OJ 系统进行简介。

### 1.3.1 国内研究现状

#### （1）浙江大学 Online Judge (ZOJ)

国内最早也是最有名气的 OJ，有很多高手在上面做题。特点是数据比较刁钻，经常会有想不到的边界数据，很能考验思维的全面性。

#### （2）北京大学 Online Judge (POJ)

建立较晚，但题目加得很快，现在题数和 ZOJ 不相上下，特点是举行在线比赛比较多，数据比 ZOJ 上的要弱，有时候同样的题同样的程序，在 ZOJ 上 WA，在 POJ 上就能 AC。

#### （3）同济大学 Online Judge (TOJ)

这个 OJ 题数上不能与上两个相比，推荐这个 OJ 的原因是它是中文的，这对很多对英文不太感冒的兄弟是个好消息吧。它也因此吸引了众多高中的 OIer，毕竟他们的英文还差一些呵呵，上面的题目也更偏向高中的信息学竞赛一些。

### 1.3.2 国外研究现状

#### （1）西班牙 Valladolid 大学 Online Judge (UVA)

世界上最大最有名的 OJ，题目巨多而且巨杂，数据也很刁钻，全世界的顶尖高手都在上面。据说如果你能在 UVA 上 AC 一千道题以上，就尽管向 IBM、微软什么的发简历吧，绝对不会让你失望的。

## (2) 俄罗斯 Ural 立大学 Online Judge (URAL)

也是一个老牌的 OJ，题目不多，但题题经典。

## (3) 俄罗斯萨拉托夫国立大学(Saratov State University)(SGU)

SGU 是俄罗斯萨拉托夫国立大学(Saratov State University)用于培养 ACM 选手的训练网站。这个网站的建成时期较晚，但随着比赛的举行以及新题目的加入，这个题库的题目也日渐丰富。这个题库 的一大特点就是 Online Judge 功能强大，它不仅使你避开了多数据处理的繁琐操作，还能告诉你程序错在了第几个数据。这一点虽然与 ACM 的 Judge 有些出入，但是却方便了调试程序。与 UVA 相比，这里的题目 在时间空间上要求都比较严格，而且更多的考察选手对算法的掌握情况，所以特别推荐冲击 NOI 的选手也来做一做。

## (4) UsacoGate Online Judge (USACO)

全美计算机奥林匹克竞赛 (USACO) 的训练网站，特点是做完一关才能继续往下做,与前面的 OJ 不同的是测试数据可以看到，并且做对后可以看标准解答。

# 2 需求与可行性分析

## 2.1 项目必要性

本系统所面向的用户是广大的学生、教师和普通用户，为学生提供良好的学习平台，为老师提供一个功能完备的练习、测试及相关统计信息。

本系统是对可扩展系统的研究和设计，扩展的 EPOJS 为计算机相关专业的学生提供一个很好的学习和交流平台。其根本需求是提高学生学习程序设计的兴趣和学习的质量，提高教师的管理效能。对于传统的在线测评系统，并没有结合到教学中去，也没有相应的交流平台，不利于技术之间的交流。同时，所提供的统计信息不够丰富，针对性不强。

本系统旨在解决部分的需求，研究并设计一个可扩展的框架，对多种程序设计语言进行可扩展支持。而本次毕业设计将重点扩展 Java 语言的在线测评模块和论坛模块。论坛系统用于学生，教师以及其他使用本系统的用户进行技术交流。

## 2.2 系统需求

### 2.2.1 外部行为需求

本系统涉及到的用户角色有普通用户，学生，教师，超级管理员，超级管理员是具有所有权限的，也就是可以使用系统的所有功能。系统要求对于不同的用户角色，可以进行权限的分配。

从外部来看，本系统主要提供以下几个功能：

- 用户可以选题进行解答，在线进行评测，获得结果，查看相关的统计信息。
- 学生可以参加某个教师的课程，完成老师布置的作业，查询作业情况。
- 教师可以创建课程和题目，布置作业，查看学生的作业情况，统计信息。
- 用户可以使用内部论坛系统。

下面我们来看一看可扩展的在线测评系统顶层用例图：

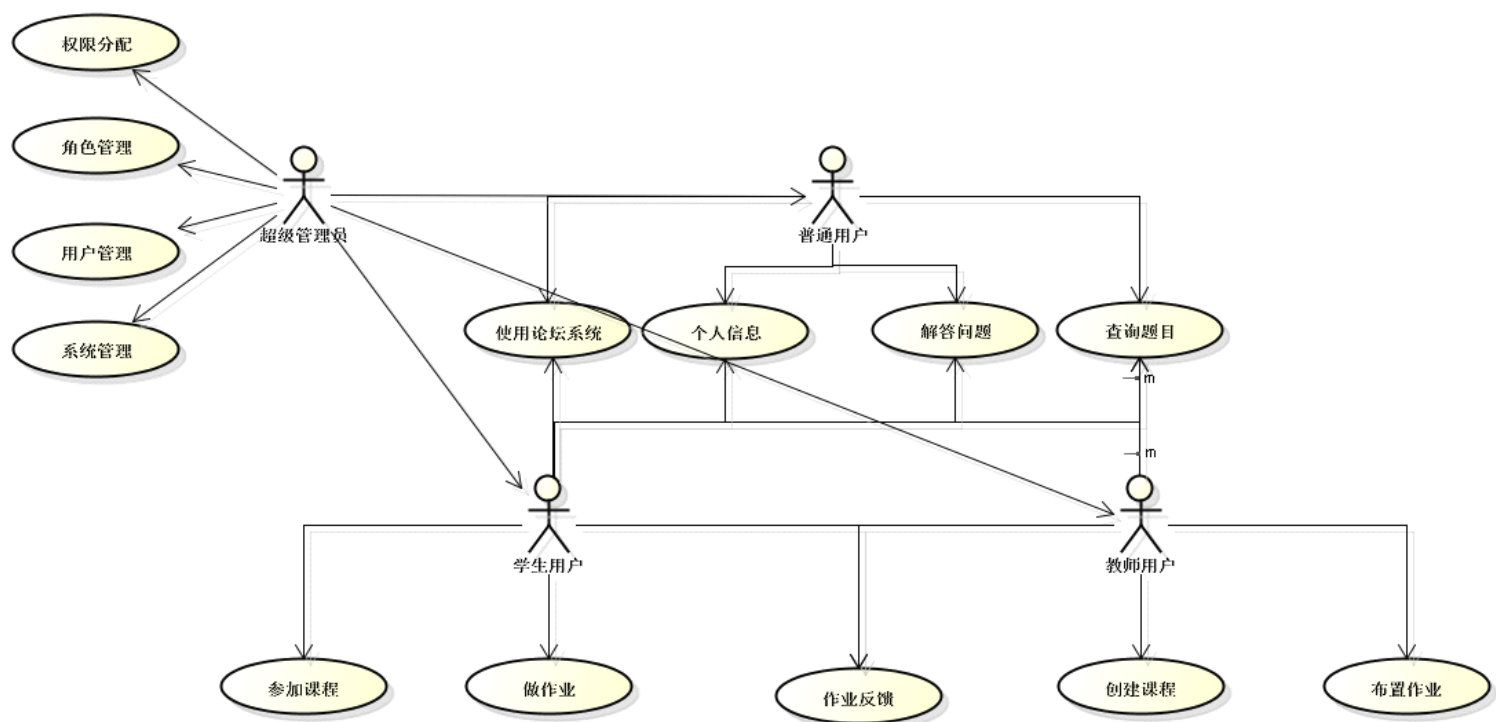


图 2-2-1 系统顶层用例图

上图描述了系统的主要功能，更进一步地，我们来定义系统的详细功能需求

如下（按照系统的角色来定义功能）：

表 2-2-1 角色功能表

角色	功能列表	公有功能
普通用户	NULL	1 个人信息查看与更新 2 论坛系统-板块列表 3 论坛系统-主题列表 4 论坛系统-显示主题回复列表 5 论坛系统-回复主题 6 论坛系统-编辑、发表主题 7 POJ-查看，查询题目列表 8 POJ-在线解题 9 POJ-重解问题 10 用户注册
学生	1 检索课程 2 参加课程 3 查看相应课程作业 4 做作业并提交 5 查看作业情况	
教师	1 我的课程 CRUD 2 查看对应课程学生信息 3 添加学生到我的指定课程 4 查看指定课程的学生作业 5 查看作业统计信息	
超级管理员	具备所有功能，列表如下： 1 用户的 CRUD 2 角色的 CRUD 3 权限的分配 4 论坛管理 5 测试题目管理	

### 2.2.2 内部特性需求

内部特性需求主要是作为开发者角度去看这个问题，为了得到更好的扩展性或是其他性能方面的要求，定义了以下需求：

- （1） 系统必须是能够比较方便的进行功能模块的扩展，如扩展在线聊天模块。
- （2） 必须能够实现基于角色的权限控制模式。
- （3） 系统可以运行在 Windows 和 Linux 上面，即能够跨平台。

## 2.3 项目可行性

### 2.3.1 经济可行性

从效益上面来说，这个项目并没有金钱的上的收益，但是作为一个毕业设计，所涉及到的知识比较全面，是对大学四年来学习知识的一个总结和应用，这将是一笔人生的宝贵财富。

### 2.3.2 技术可行性

现有的技术完全可以实现本系统，就系统本身而言，还是存在一些难点，如如何实现系统的扩展性，如何进行代码的在线编译和执行，如何将基于角色的权限控制模型无缝应用在 WEB 系统中等。建议使用 Spring，Hibernate，Lucene，Struts2 等主流的开源技术，技术上面本身没有太多的限制。

### 2.3.3 法律可行性

在本系统的开发过程中，所使用的都是开源的技术，绝对不会在社会上和政治上面引起任何的侵权行为，不会对任何个人或组织造成利益上面的冲突。

## 2.4 系统开发环境

### 2.4.1 客户端软件

- 操作系统： windows，Linux Desktop
- 浏览器： Chrome（推荐），Firefox， IE9 及以上版本

### 2.4.2 服务端软件

- 操作系统： windows server， Linux server
- WEB 服务器： Tomcat7.0
- 数据库： MySQL5.0
- JDK/JRE： jdk1.7

- C++编译器: g++3.4.5

### 2.4.3 开发工具

- 开发平台: Eclipse3.7
- 构建工具: Ant
- 版本控制: GitHub
- UML 建模: Astah
- 测试框架: JUnit4
- 应用框架: Spring, Hibenate, jQuery
- 浏览器: Chrome, Firefox, IE9 及以上版本

## 3 相关技术简介

在本次项目中，所使用的均是主流的开源技术，易于获取，网上的学习资源丰富，关于该类开源框架的论坛和社区也不少，并且在企业中，这些技术都应用得非常广泛，这也是本项目选用这些技术的根本原因。下面就来简单介绍一下这些开源的技术框架。

### 3.1 Spring

作为在 Java 领域最为成功的开源软件之一，Spring 在 Java EE 开发中，使用者众多。2002 年，伴随着 Rod Johnson 的《Expert One-on-One J2EE Design and Development》一书出版而发布的 Spring 框架（也就是当年的 interface21），经过几年的发展，现在已经逐渐成熟起来，Spring 带来的崭新开发理念，也早已伴随着它的广泛应用而“飞入寻常百姓家”。

#### 3.1.1 简介

简化 Java 企业应用开发是 Spring 的目标，可以这么说，Spring 为开发者提供了一站式的轻量级应用开发框架（或者说平台）。作为平台，Spring 抽象了我们在许多应用中遇到的共性问题；同时，作为一个轻量级的应用开发框架，

Spring 有其自身的特点,通过这些特点, Spring 充分体现了它的设计理念: 在 Java EE 的应用开发中, 支持 POJO (Plain Old Java Objects, 简单的 Java 对象) 和使用 JavaBean 的开发方式, 使应用米娜想接口开发, 充分支持 OO (面向对象) 的设计方法。

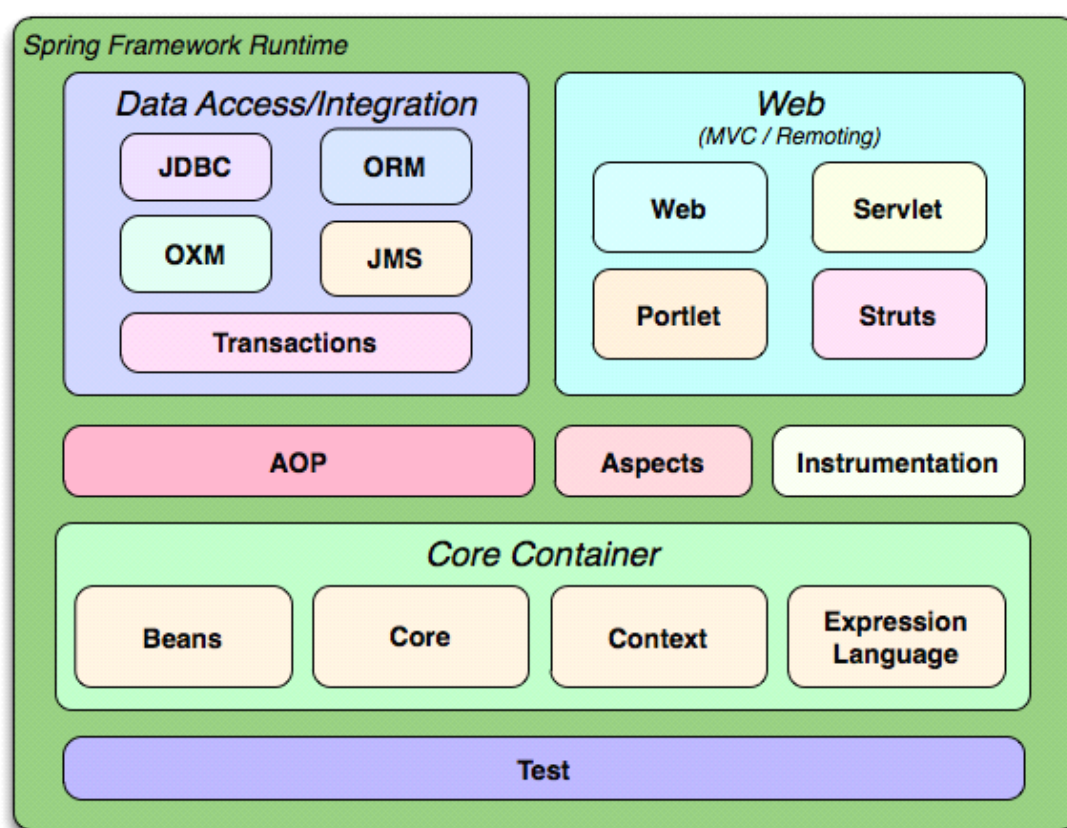


图 3-1 Spring 总体架构图

简单来说, Spring 是一个轻量级的控制反转 (IoC) 和面向切面 (AOP) 的容器框架。

- **轻量**——从大小与开销两方面而言 Spring 都是轻量的。完整的 Spring 框架可以在一个大小只有 1MB 多的 JAR 文件里发布。并且 Spring 所需的处理开销也是微不足道的。此外, Spring 是非侵入式的: 典型地, Spring 应用中的对象不依赖于 Spring 的特定类。
- **控制反转**——Spring 通过一种称作控制反转 (IoC) 的技术促进了松耦合。当应用了 IoC, 一个对象依赖的其它对象会通过被动的方式传递进来, 而不



是这个对象自己创建或者查找依赖对象。你可以认为 IoC 与 JNDI 相反——不是对象从容器中查找依赖，而是容器在对象初始化时不等对象请求就主动将依赖传递给它。

- **面向切面**——Spring 提供了面向切面编程的丰富支持，允许通过分离应用的业务逻辑与系统级服务（例如审计（auditing）和事务（transaction）管理）进行内聚性的开发。应用对象只实现它们应该做的——完成业务逻辑——仅此而已。它们并不负责（甚至是意识）其它的系统级关注点，例如日志或事务支持。
- **容器**——Spring 包含并管理应用对象的配置和生命周期，在这个意义上它是一种容器，你可以配置你的每个 bean 如何被创建——基于一个可配置原型（prototype），你的 bean 可以创建一个单独的实例或者每次需要时都生成一个新的实例——以及它们是如何相互关联的。然而，Spring 不应该被混同于传统的重量级的 EJB 容器，它们经常是庞大与笨重的，难以使用。
- **框架**——Spring 可以将简单的组件配置、组合成为复杂的应用。在 Spring 中，应用对象被声明式地组合，典型地是在一个 XML 文件里。Spring 也提供了很多基础功能（事务管理、持久化框架集成等等），将应用逻辑的开发留给了你。

所有 Spring 的这些特征使你能够编写更干净、更可管理、并且更易于测试的代码。它们也为 Spring 中的各种模块提供了基础支持。

### 3.1.2 应用场景

通过前面的介绍，我们了解到 Spring 是一个轻量级框架。在 Spring 这个一站式的应用平台或框架中，其中各个模块除了依赖 IoC 和 AOP 之外，相互之间并没有很强的耦合性，Spring 的最终目标是简化应用开发的编程模型。一方面，我们可以把 Spring 作为一个整体来使用，另一方面，也可以各取所需，把 Spring 的各个模块拿出来独立使用，这取决于我盟对 Spring 提供服务的具体需求。正因为如此，才大大地拓宽了 Spring 的应用场景。

在 JavaEE 企业应用的开发中，我们了解了使用 Spring 最为基本的应用场景，就是 SSH 架构来完成企业应用开发，取代传统的 EJB 开发模式，在 SSH 中，Struts

作为 Web UI， Spring 作为中间件平台， Hibernate 作为数据持久化工具（ORM-Object-Relation Mapping）来操作关系数据库。

在 Spring 的实现中，它的核心实现，如 IoC 的容器的实现，是直接依赖于 JVM 虚拟机的，也就是说在 Java 环境中， Spring IoC 容器是可以独立使用的。对于 Spring 而言，如果要在 .NET 环境中使用， Spring 团队也提供了 Spring .NET 的实现；在 Android 平台中也有支持。从这些应用场景来看， Spring 设计时的轻量级特性，以及推崇 POJO 开发，所以使用起来非常灵活。

### 3.1.3 价值

在 Spring 的应用中， Spring 团队为我们列举了 Spring 的核心价值，非常值得参考：

- Spring 是一个非侵入性（Non-invasive）框架，其目标是是应用程序代码对框架的依赖最小化，应用代码可以在没有 Spring 或者其它容器上面进行。
- Spring 提供了一个一致的编程模型，使应用可以直接使用 POJO 进行开发，从而与运行环境（如应用服务器）隔离开来。
- Spring 推动应用的设计风格向面向对象和面向接口编程转变，提高了代码的重用性和可测试性。
- Spring 改进了体系结构的选择，虽然作为应用平台， Spring 可以帮助我们选择不同的技术实现，比如从 Hibernate 切换大其他 ORM 工具（如 MyBatis），从 Struts 切换到 Spring MVC，尽管我们通常不会这么做，但是我盟在技术方案的选择使用 Spring 作为应用平台， Spring 至少为我们提供而来这种可能性，从而降低了平台锁定的风险。

## 3.2 Hibernate

Hibernate 是当前主流的开源 ORM（Object Relation Mapping-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。

Hibernate 不仅负责从 Java 类映射到数据库表（和从 Java 数据类型到 SQL 数据类型），但也提供了数据查询和检索。我们不用手工去编写繁琐的 SQL 语句和 JDBC，Hibernate 可以使用 HQL 来帮助我们完成这些更为底层的实现，从而快速开发。通过隐藏底层实现，作为一个开发人员，你可以使用纯面向对象的方式去思考问题，提高对问题空间的理解和抽象，Hibernate 允许你这么做。

### 3.3 jQuery

jQuery 是一个前端的轻量级 JavaScript 库，它以“Write Less, Do More”为设计理念。是继 prototype 之后又一个优秀的 JavaScript 框架。它兼容 CSS3，还兼容各种浏览器。jQuery 使用户能更方便地处理 HTML documents、events、实现动画效果，并且方便地为网站提供 AJAX 交互。jQuery 还有一个比较大的优势是，它的文档说明很全，而且各种应用也说得很详细，同时还有许多成熟的插件可供选择。jQuery 能够使用户的 html 页面保持代码和 html 内容分离，也就是说，不用再在 html 里面插入一堆 js 来调用命令了，只需定义 id 即可。

更为重要的是，jQuery 是开源免费的，对于热爱 javascript 的开发人员来说，jQuery 的源代码具有非常好的参考价值。简单的说，jQuery 具有以下一些特点：

- 动态特效
- AJAX
- 通过插件来扩展
- 方便的工具 - 例如浏览器版本判断
- 渐进增强
- 链式调用
- 多浏览器支持

### 3.4 Ajax

相信做过 web 开发的人员都不陌生，Ajax 就是“Asynchronous JavaScript and XML”（异步 JavaScript 和 XML）。在过去还是主要以 XML 作为数据传输格式，但是 XML 太过庞大和复杂，逐渐被 JSON 数据格式取代。现在大多数 Ajax 都是以 JSON 作为数据传输的格式。

Ajax 不是一种新的编程语言，而是一种用于创建更好更快以及交互性更强的 Web 应用程序的技术。通过 Ajax，可使用 JavaScript 的 XMLHttpRequest 对象来直接与服务器进行通信。通过这个对象，可在不重载页面的情况与 Web 服务器交换数据。Ajax 在浏览器与 Web 服务器之间使用异步数据传输（HTTP 请求），这样就可使网页从服务器请求少量的信息，而不是整个页面。

如前面提到的，Ajax 可以大大提高用户与 web 应用程序的交互速度，对于之更心一小部分显示文档的需求，Ajax 技术缩短了文档传送和文档呈现所需要的时间。Ajax 能够极大地改善用 Web 用户的体验。

## 4 基础架构设计与实现

### 4.1 可扩展架构设计与实现

#### 4.1.1 可扩展架构设计

##### （1）系统流程

在 Java web 中，WEB 具有一定的规范，应用服务器会先读取部署描述符 web.xml 进行系统的部署，突破口就在这里。

可扩展的架构设计方案如下：

- （1）应用服务器读取 web.xml，初始化监听器。
- （2）应用服务器通过反射调用监听器的初始化方法。
- （3）在监听器的初始化方法中，读取项目的模块配置文件，修改与 Spring 相关的配置并保存。
- （4）启动 Spring 在 web 环境中的容器，此时读取到的是最新的配置文件。
- （5）根据模块配置文件，刷新系统权限数据，并更新数据库。
- （6）执行每个模块监听器的初始化方法，初始化每个模块。
- （7）启动完毕。

下面是启动流程顺序图：

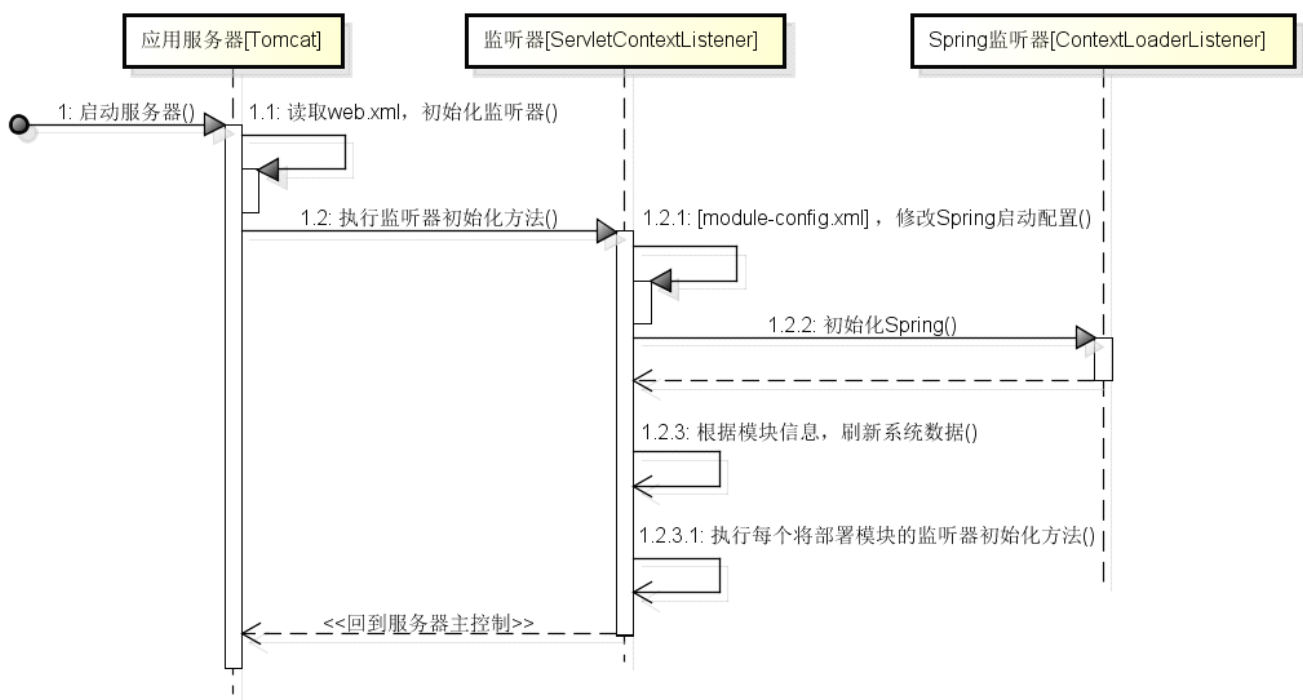


图 4-1-1 系统启动流程图

## (2) 扩展需要的资源

上面介绍了系统的启动流程，接下来介绍系统是如何进行扩展的。在进行扩展操作之前，必须先准备好扩展模块的资源，要扩展模块的资源包含下面几个方面：

- 模块编译好的 class 文件
- 页面显示的资源文件[jsp, js, css, images 等]
- 项目配置文件[spring,hibernate,privileges]

关于 spring, hibernate, privilege 配置文件的格式如下（不作强制要求）：

建议命名格式：

- ◆ Spring:      applicationContext-[模块名称].xml
- ◆ Hibernate:   hibernate.cfg-[模块名称].xml
- ◆ Privileges:   privilege-[模块名称].xml

### 1) applicationContext-[模块名称].xml 文件

该文件和普通的 Spring 配置文件一致，主要配置相关的 bean 以及搜索的包路径。可以作为 Spring 的配置文件导入到框架的 Spring 核心配置文件中，请看

下面一个 Spring 文件的示例：

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:p="http://www.springframework.org/schema/p"
  xmlns:context="http://www.springframework.org/schema/context"
  xmlns:mvc="http://www.springframework.org/schema/mvc"
  xsi:schemaLocation="
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-context-3.0.xsd
    http://www.springframework.org/schema/mvc
    http://www.springframework.org/schema/mvc/spring-mvc-3.0.xsd">

  <!-- 搜索的包 -->
  <context:component-scan base-package="edu.zhku"/>

  <!-- 导入hibernate配置文件 -->
  <import resource="frame/applicationContext-hibernate.xml"/>

  <!-- 导入web层相关的配置 -->
  <import resource="frame/applicationContext-web.xml"/>

  <mvc:annotation-driven/>
  <!-- 启用Spring注解 -->
  <context:annotation-config/>
</beans>
```

## 2) hibernate.cfg-[模块名称].xml

该配置文件主要是配置相关的模块实体类映射文件，样例文件如下：

```
<!DOCTYPE hibernate-configuration PUBLIC
  "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
  "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
  <session-factory>
    <!-- 配置映射文件 -->
    <mapping resource="edu/zhku/fr/domain/User.hbm.xml" />
    <mapping resource="edu/zhku/fr/domain/Privilege.hbm.xml" />
    <mapping resource="edu/zhku/fr/domain/Role.hbm.xml" />
  </session-factory>
</hibernate-configuration>
```

### 3) privilege-[模块名称].xml

该文件非常重要，它描述了在本模块中的所有功能，是实现权限控制的必要条件，每一个功能亦称之为一个权限。

```
<?xml version="1.0" encoding="UTF-8"?>
<privs>
  <!-- 一个模块作为一个权限组 -->
  <top-privilege name="系统管理" action="systemManage">
    <!-- 第二级别显示在下拉菜单中 -->
    <priv name="用户管理" action="user/userList">
      <!-- 第三级别的是在页面中出现的，一般只有三级 -->
      <sub-priv name="用户列表" action="user/userList" />
      . . . . .
    </priv>
    <priv name="角色管理" action="role/roleList">
      <sub-priv name="角色列表" action="role/roleList" />
      . . . . .
    </priv>
  </top-privilege>
</privs>
```

### (3) 开始扩展

准备好了上面的步骤之后，接下来就是把该模块添加到原有的系统中去了，只要做以下几个步骤即可。

#### 1) 修改 module-config.xml

示例（配置基础应用模块）：

```
<?xml version="1.0" encoding="UTF-8"?>
<!--说明
module: 表示一个模块
module[name]: 模块名称，这个是唯一的
module[deploy]: 表示是否要在安装系统的时候安装本模块
module[depends]: 表示本模块依赖于其他的哪些模块，分隔符可以是 ';' ',' ':' '-' ' '
module>>hibernate-cfg: 表示该模块对应的Hibernate配置文件路径，只能接收一个文件，
                        因为其他配置都可以通过import方式导入
module>>spring-cfg: 表示该模块对应的 Spring配置文件路径，只能接收一个文件，因为其他
                    配置都可以通过import方式导入
module>>privilege-cfg: 表示该模块对应的权限数据配置文件，只能接收一个文件，因为其他
                      配置都可以通过import方式导入
module>>listeners: 表示该模块下的监听器
属性:
class: 代表该监听器对应的类，该类必须要有一个public的空构造函数
deploy: 表示是否部署，即是否在web.xml中添加了该监听器，true表示添加了，那
```

么在模块初始化的时候就不会在执行该监听器了，否则就会反射创建对象，并执行`contextInitialized`方法来初始化本模块所需要的数据。

```
-->
<modules>
  <module name="base" deploy="true">
    <!-- 对应模块的配置文件位置 -->
    <hibernate-cfg>classpath:base/hibernate.cfg-base.xml</hibernate-cfg>
    <spring-cfg>classpath:base/applicationContext-base.xml</spring-cfg>
    <privilege-cfg>classpath:base/privileges-base.xml</privilege-cfg>
    <!-- 对应模块启动监听器，可以没有，看模块的需求 -->
    <listeners>
      <listener class="edu.zhku.base.Listener.BaseContextLoaderListener"
        deploy="false"/>
    </listeners>
  </module>
</modules>
```

## 2) copy 资源

将 classes 复制到主系统的 classes 下面，同时把配置文件也复制在下面，目录结构与 classes 目录保持一致。最后把页面相关资源[jsp, html, js, css, image 等]复制到 WebContent 下面，目录结构相对于 WebContent 要一致。

## 3) 重新启动系统即可

### 4.1.2 可扩展架构实现

前面介绍了架构的运行流程，接下来就是详细讨论如何实现这么一个架构了，本架构的实现主要依赖与 Spring 和 web 应用服务器。

#### (1) 可扩展架构核心类图



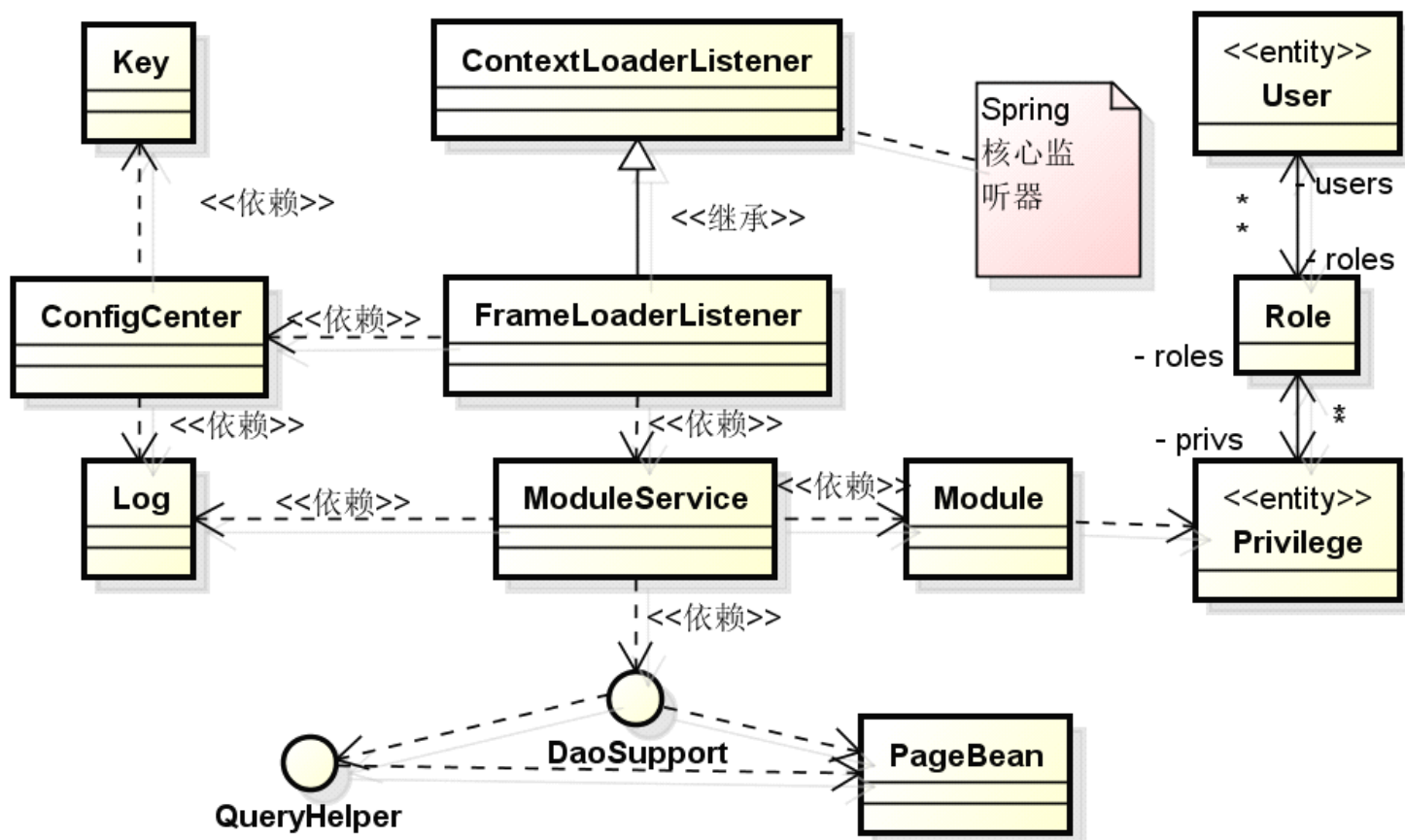


图 4-1-2 扩展架构核心类图

表 4-1-1 扩展架构核心组件描述

组件	组件职责
ContextLoaderListener	Spring 核心监听器，启动 SpringMVC。
FrameLoaderListener	系统核心监听器，模块加载，初始化相关配置，如初始化 ConfigCenter，Log，准备系统所需数据等。
ModuleService	这个是可扩展的核心组件，管理模块的部署和相关资源的清理和构建。系统启动的时候调用，主要有一下几个方法，init 进行模块初始化，initModuleContext 进行模块上下文初始化，clear 进行系统的资源清理，freshPrivilege 进行系统权限数据的更新，contextDestory 进行系统上细纹资源的清理。本组件是实现扩展的核心。
ConfigCenter	保存配置信息，以便其他组件获取相关的配置信息，系统启动时候初始化。

Key	为配置体系服务, 提供 key-value, 作为 ConfigCenter 的 key 值, 当系统没有指定相关的配置时, 使用 key 中的默认值。
Log	轻量级的日志服务组件, 若没有指定日志配置文件, 将自动加载默认配置文件 classpath:log.cf。
Module	是模块配置的一个抽象, 在 module-config.xml 中进行了配置, ModuleManager 会读取这个配置并转化成 Module 对象, 然后根据模块部署的情况进行系统初始化处理。
DaoSupport	数据访问组件接口, 实现对数据的访问, 使用的是 ORM 框架, 子类是使用 Hibernate 的实现类。
QueryHelper	查询帮助组件接口, 支持条件查询, 分页查询和排序。
PageBean	保存分页查询的结果 Bean, 其中包含结果信息, 总页数, 当前页, 总记录数目等分页相关的信息。
Privilege	权限类, 领域模型对象, 为实现 RBAC 而设计。
Role	角色类, 领域模型对象, 为实现 RBAC 而设计。
User	用户类, 领域模型对象, 为实现 RBAC 而设计。

## (2) 权限模型设计

这个主题在下一节中进行描述, 见 [4.2 RBAC 权限模型设计与实现](#)

### 4.1.3 架构的不足

由于本架构是基于 Spring 进行设计的, 对 Spring 的依赖很大。目前并没有设计成其他技术的扩展, 所以, 在进行系统的模块扩展的时候, 必须使用相同的架构实现, 这是本系统的一大缺陷。

另外, 本系统并没有继续开发自动部署的工具, 使得在进行模块导入的时候要花点时间。

## 4.2 RBAC 权限模型设计与实现

前面所提到的扩展架构其实要对权限相关数据进行初始化。那么本节就权限模型的设计和实现进行阐述。

不同职责的人员，对于系统操作的权限应该是不同的。可以对“组”进行权限分配。对于一个大企业的业务系统来说，如果要求管理员为其下员工逐一分配系统操作权限的话，是件耗时且不够方便的事情。所以，系统中就提出了对“组”进行操作的概念，将权限一致的人员编入同一组，然后对该组进行权限分配。权限管理系统应该是可扩展的。它应该可以加入到任何带有权限管理功能的系统中。就像是组件一样的可以被不断的重用，而不是每开发一套管理系统，就要针对权限管理部分进行重新开发。

满足业务系统中的功能权限。传统业务系统中，存在着两种权限管理，其一是功能权限的管理，而另外一种则是资源权限的管理，在不同系统之间，功能权限是可以重用的，而资源权限则不能。

那么本系统将设计以下权限控制模型：

权限 ----> 权限组 ---> 角色 ---> 用户组

下面对这个模型进行简单的介绍：

#### (1) 权限

在系统中，权限通过 *模块+动作* 来产生，模块就是整个系统中的一个子模块，可能对应一个菜单，动作也就是整个模块中（在 B/S 系统中也就是一个页面的所有操作，比如“浏览、添加、修改、删除”等）。将模块与之组合可以产生此模块下的所有权限。

#### (2) 权限组

为了更方便的权限的管理，另将一个模块下的所有权限组合一起，组成一个“权限组”，也就是一个模块管理权限，包括所有基本权限操作。比如一个权限组（用户管理），包括用户的浏览、添加、删除、修改、审核等操作权限，一个权限组也是一个权限。

#### (3) 角色

权限的集合，角色与角色之间属于平级关系，可以将基本权限或权限组添加到一个角色中，用于方便权限的分配。

#### (4) 用户组

将某一类型的人、具有相同特征人组合一起的集合体。通过对组授予权限（角色），快速使一类人具有相同的权限，来简化对用户授予权限的繁琐性、耗时性。

用户组的划分，可以按职位、项目或其它来实现。用户可以属于某一个组或多个组。

#### 4.2.1 web 中的权限模型概述

在 web 系统中，权限的控制有其特性。在 web 系统中，我们在 web 页面中进行的每一个动作对应于浏览器做出的反应都是一一对应的，如果结合后端的服务，那么用户在前端所做的动作，每次发出的请求都是与后端服务意义对应的。而请求本质上就是一个 URL 的访问。由于在一个单独的 web 系统中，URL 是唯一的，因此，设计 web 中的权限模型方案就可以以 URL 为突破口。本系统所使用的权限模型方案正是基于站内 URL 唯一的特性设计的。以某一个 URL 作为一个权限的代表。在 web 中进行权限的控制有两个方面的内容，一是权限的显示问题，另一个是权限的拦截问题。所谓权限的显示，就是说在 web 页面中，系统会根据用户的角色来判定你是否具备某个权限或不具备某个权限，如果具备，就会提供一个操作界面给你，如果不存在就不会提供这样的在操作界面。所谓权限的拦截，就是说，可能在 web 页面中没有显示，但是这个用户直接在浏览器上面输入一个自己没有权限访问的 URL，那么后台服务器应该能够对这个请求进行拦截，对没有权限的操作进行阻塞。

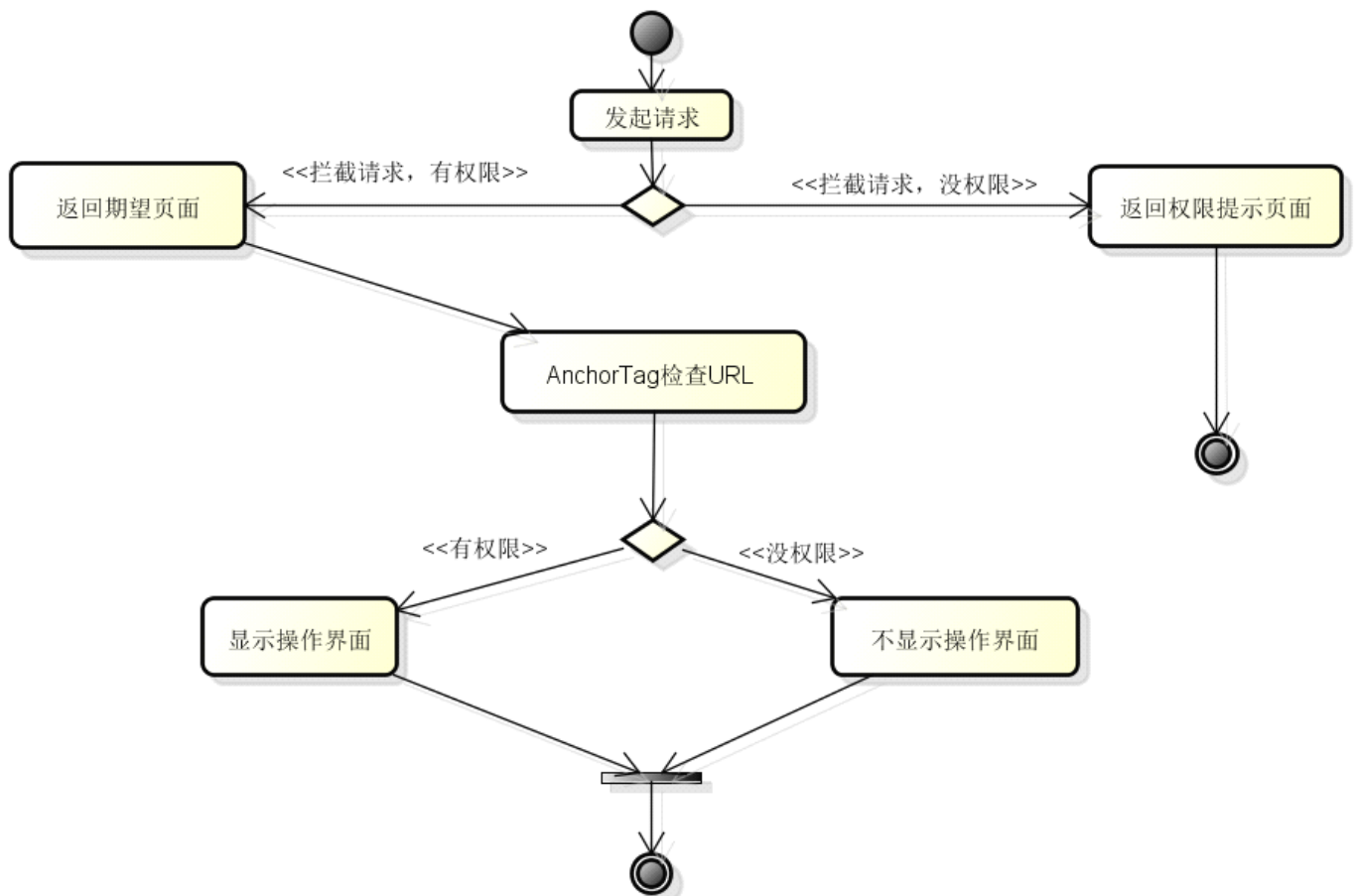


图 4-2-1 web 权限控制活动图

权限控制请求流程描述：

- (1) 用户使用 URL 发起请求。
- (2) 服务端使用 CheckPrivilegeInterceptor 拦截到该请求，解析该 URL，分析出权限，如果有权限访问就正常返回期望页面并跳到 (3)，如果没有权限就直接跳到 (4)。
- (3) 根据用户的角色信息，拿到用户的所有权限，对于链接，使用自定义的 a 标签，实现对权限的控制，一旦没有操作对应 URL 的权限就不显示该标签，最后返回一个正常页面。
- (4) 结束。

#### 4.2.2 领域模型分析

要设计基于角色的权限控制模型，用户便是使用权限的主体。这里的领域模型包括：用户，角色和权限。

用户可以有多个角色，一个角色也可以有多个用户，一个角色包含 0 个或多个权限，一个权限可以属于多个角色，下面是这三者之间的类图。

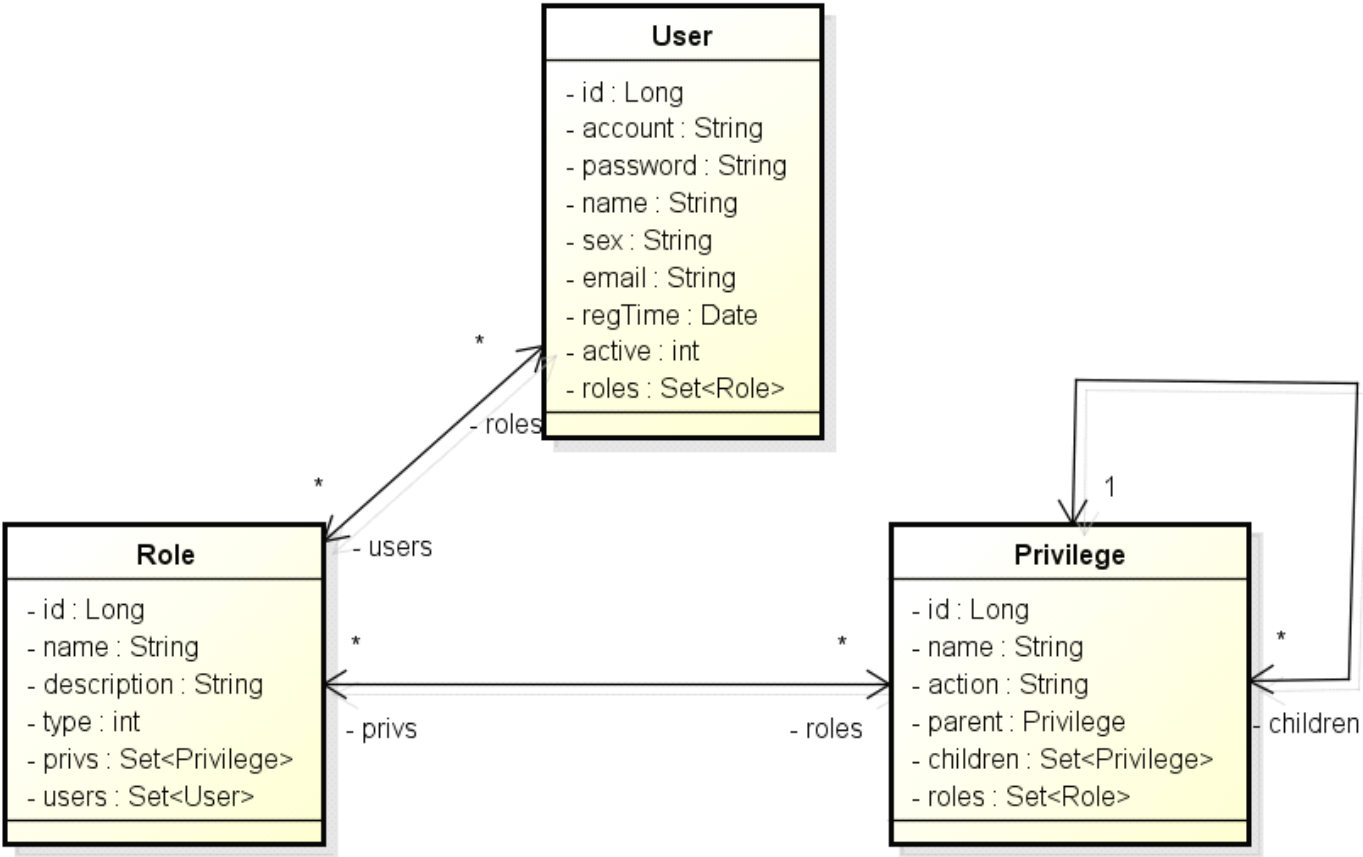


图 4-2-2 权限领域模型关系

从上面可以看到，用户并不直接与权限关联，而是通过角色划分，权限分配到角色上，角色具备一组权限。但这是粗粒度上面的划分，仅仅是控制到功能上面的权限控制，并没有控制到字段的权限控制。这与 Oracle 的权限不同，Oracle 权限可以控制到某行数据的某个字段，是细粒度的权限模型。

表 4-2-1 模型属性说明

模型	属性说明
User	<pre>private Long id;    // 用于数据库中唯一标识 private String account; // 帐号 private String password; // 密码 private String name; // 真实姓名 private String sex; // 性别 private String email; // 邮件 private Date regTime; // 注册时间</pre>

	<pre>// 邮件激活的状态，1表示已经激活，0表示未激活 private Integer active; // 包含的角色 private Set&lt;Role&gt; roles = new HashSet&lt;Role&gt;();</pre>
Role	<pre>private Long id;    // 数据库中唯一标识 private String name; // 角色的名称，必须是唯一的 private String description; // 角色相关的描述 // 1表示系统固有角色，不可删除，0表示扩展属性，可删除 private Integer type = TYPE_CAN_DEL; // 角色具有那些权限 private Set&lt;Privilege&gt; privs = new HashSet&lt;Privilege&gt;(); // 该角色下面有哪些用户 private Set&lt;User&gt; users = new HashSet&lt;User&gt;();</pre>
Privilege	<pre>private Long id;    // 数据库中唯一标识 private String name; // 权限名称，唯一 private String action; // 权限对应的URL动作，唯一 private Privilege parent; // 父权限 // 子权限 private Set&lt;Privilege&gt; children=new HashSet&lt;Privilege&gt;() // 包含的角色 private Set&lt;Role&gt; roles = new HashSet&lt;Role&gt;();</pre>

### 4.2.3 基础数据操作

要实现权限的分配和存储，就必须进行相关数据的存取，本系统使用 MySQL 数据库存储相关的信息，使用 Hibernate 实现数据的访问。组件之间的类图关系如下（关于 DAO 数据访问请参照 [4.5 数据访问设计与实现](#)）：

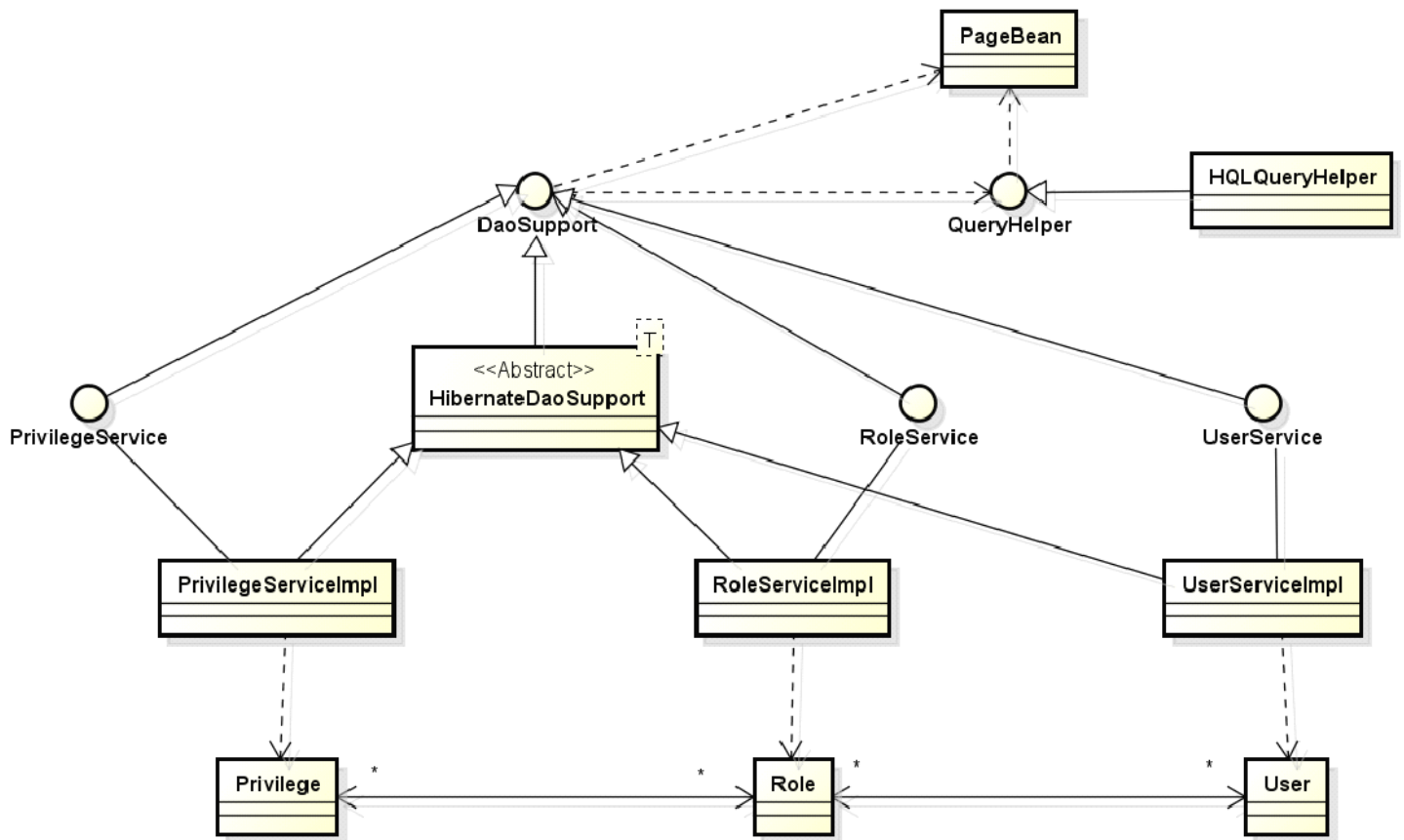


图 4-2-3 RBAC 数据访问类图

表 4-2-2 组件描述

组件	功能描述
DaoSupport<T>	数据访问接口组件，见 <a href="#">4.6</a>
HibernateDaoSupport<T>	数据访问 Hibernate 支持，见 <a href="#">4.6</a>
Privilege	领域模型，代表一个权限
PrivilegeService	权限模型数据访问服务接口
PrivilegeServiceImpl	权限模型数据访问服务接口默认实现
Role	领域模型，代表一个角色
RoleService	角色模型数据访问服务接口
RoleServiceImpl	角色模型数据访问服务接口默认实现
User	领域模型，代表用户
UserService	用户模型数据访问服务接口
ServiceImpl	用户模型数据访问服务接口实现



4.2.4 RBAC 实现

前面介绍了一些基础，知道了整个系统的权限控制流程，那么现在来具体谈谈前后端是怎么进行权限控制的。

正如[\[4.2.1 web 中的权限模型概述\]](#)所描述的那样，WEB 中的权限控制包含两个方面的内容。其一，控制权限的显示；其二，请求的拦截，并检测权限。对于其一，采用自定义标签以及自定义标签函数实现对权限的显示控制；对于其二，采用 Spring 的拦截器进行请求的拦截，检测权限以决定返回的页面。下面整体来看看这个类图设计。

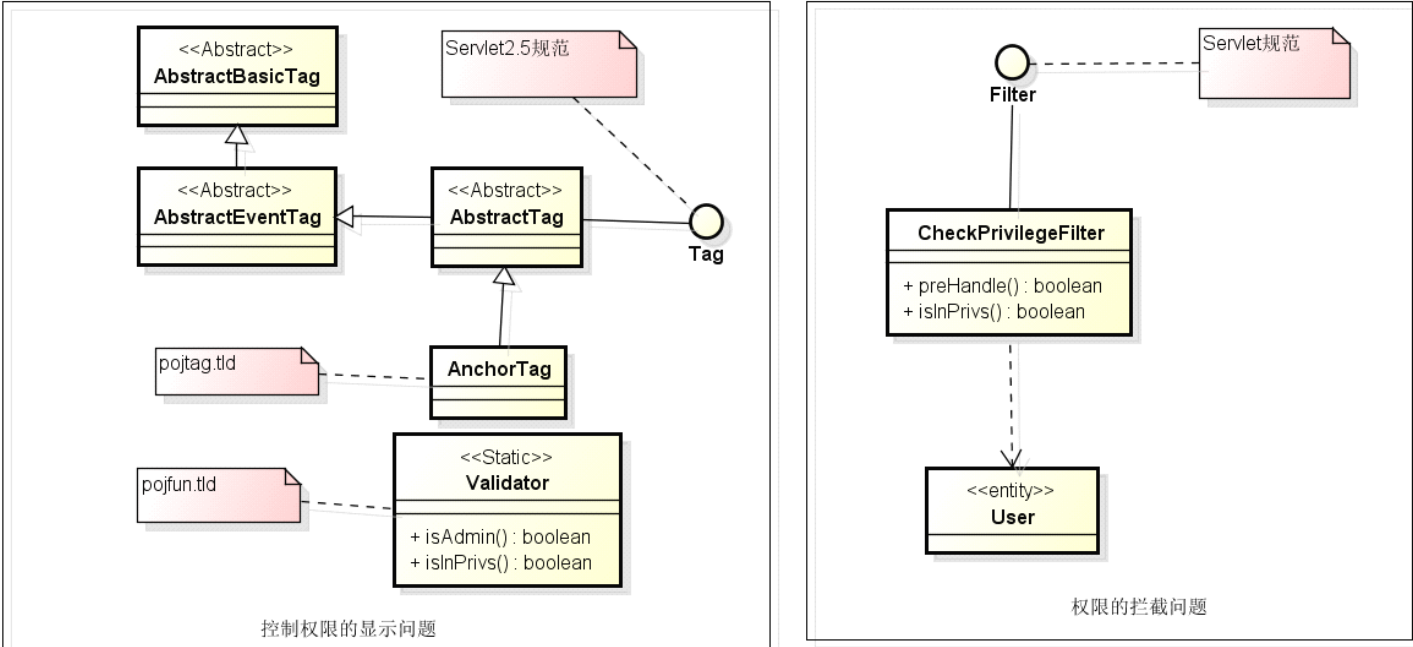


图 4-2-4 WEB 权限控制类图

表 4-2-3 WEB 权限控制组件描述

组件	功能描述
AbstractBasicTag	封装了标签的基本属性，用于继承
AbstractEventTag	封装了标签的基本事件属性，用于继承
AbstractTag	提供 Servlet 规范中，Tag 接口的默认实现，用于继承
AnchorTag	权限显示控制的核心标签
Validator	相关验证，如是否有权限，是否为管理员等
CheckPrivilegeFilter	请求过滤器，专用于权限的检测，后台的权限拦截就是靠这

	个类了，前端请求都会被这个类拦截，如果通过权限检测就返回对应页面，没有权限就重定向到没有权限提示页面
--	--

关于自定义标签以及自定义函数的使用方法在这里就不再赘述了。

权限控制流程如下：

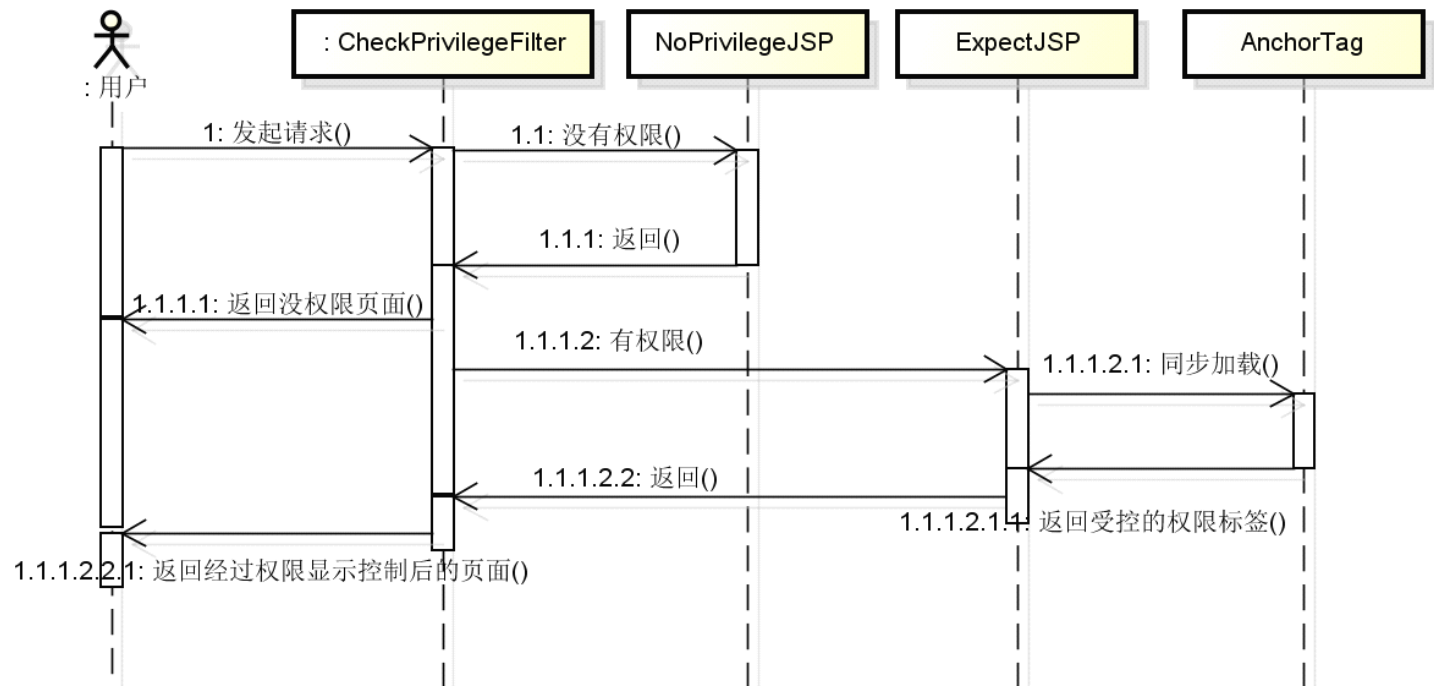


图 4-2-5 权限控制过程图

在上图中，CheckPrivilegeFilter 作为一个核心的权限控制器，起到了请求转发的作用。它会根据用户的情况进行权限检测以返回相应的页面，当返回正常页面时，便会同步加载所有的自定义标签，在自定义标签内部，会进行权限的检测，如果没有权限就不会显示该标签。如果该请求对应的用户没有操作的权限，就会返回一个没有权限的页面，告知用户没有操作权限，这个页面是可以进行配置的。具体的配置可以在 web.xml 中添加初始化参数。

## 4.3 JSONLIB 设计与实现

### 4.3.1 背景

在本系统中，所有的 Ajax 请求所使用的数据格式都是 JSON，当前确实有很多 JSON 相关的开源工具，如 json-lib，Gson，Jackson（Spring 就是使用这个）。

但是，这些开源工具都功能太过强大，就本项目而言，只需要最基本的功能即可，第三包中很多功能特性根本不需要，为了减少第三方包的依赖，决定自行设计一个简单的，适合于本项目的 jsonlib。

### 4.3.2 JSONLIB 设计

在本项目中，主要的任务是要把 java 对象转化成 javascript 可识别的 json 字符串。对于从字符串转换成 java 对象的需求并不是很大，因此在设计的时候更多精力是放在了 java 转 json 字符串的问题上面（关于本项目可在 github 上面浏览：<https://github.com/cianfree/cdesign/tree/master/jsonlib>）。

对于本 jsonlib 的设计需求，应该能够自定义解析的方式，提供一定的灵活性，以下是 jsonlib 的设计类图。

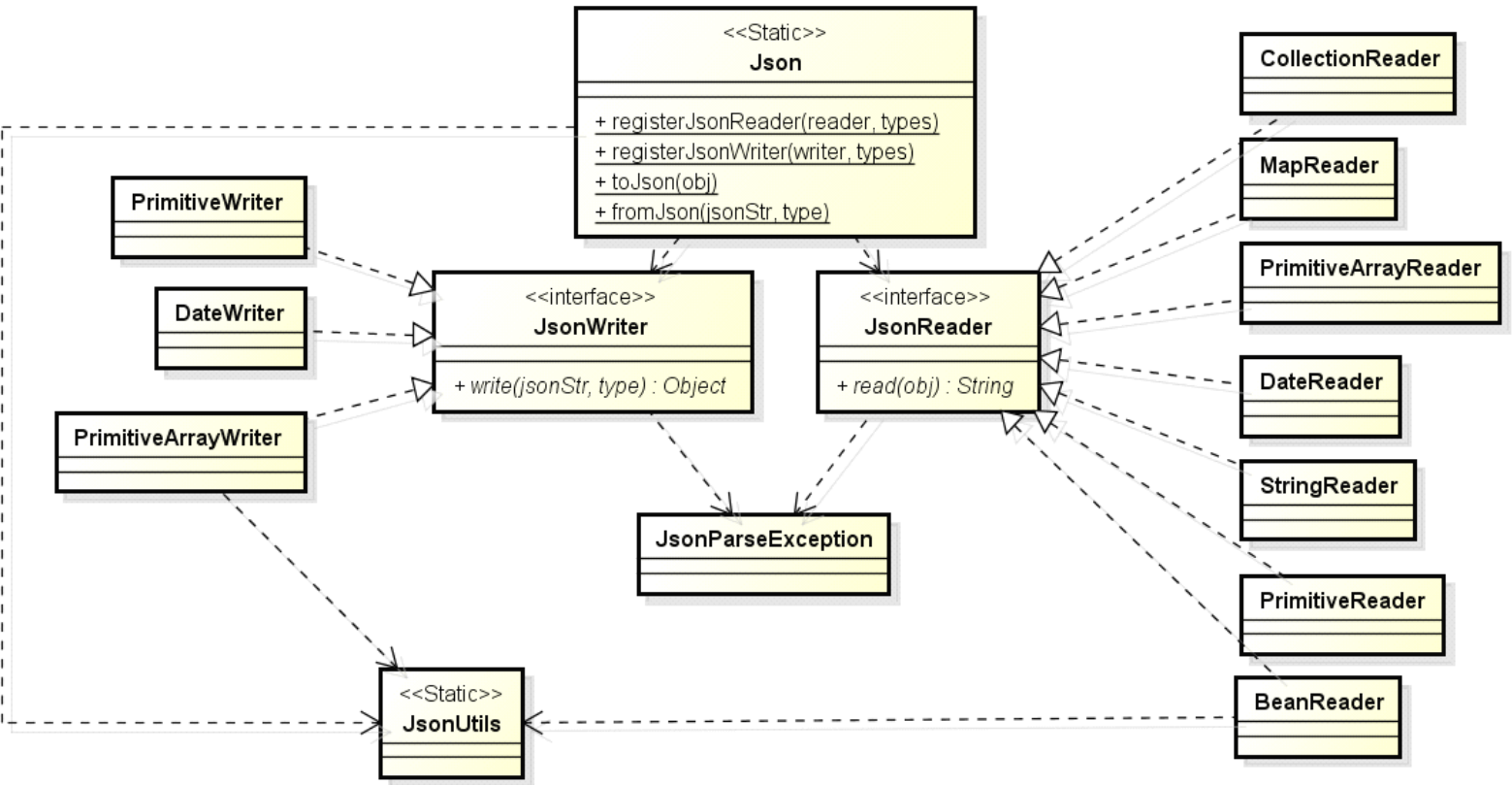


图 4-3-1 JSONLIB 类图

从上面的类图中我们可以看到，以 **Json** 为核心类，**JsonWriter** 和 **JsonReader** 为核心接口。下面对这些组件进行详细描述：

表 4-3-1 jsonlib 组件描述

组件	功能描述
Json	进行 json 数据格式转换，可以进行 <b>JsonWriter</b> 和 <b>JsonReader</b> 的注册，也就是允许开发者自定义 java 对象的 Reader 和 Writer，提供了扩展性。
JsonReader	把一个 java 对象转成一个 json 字符串，接口，开发者可以自定义 Reader，以便实现特殊 java 对象的转换。
JsonWriter	把一个 json 字符串转成一个 java 对象，接口，开发者可以自定义 Writer，实现特殊对象的转换。
JsonUtils	Json 相关的工具类，提供基础数据的处理。

JSONLIB 系统内部处理流程：

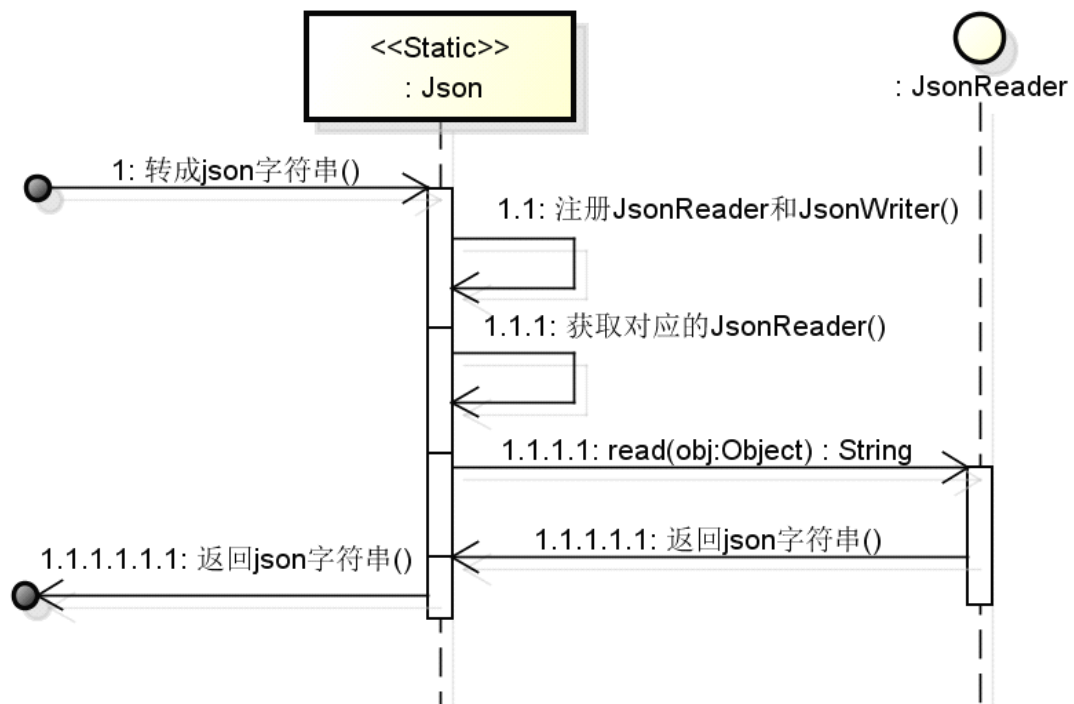


图 4-3-2 java 对象转 json 字符串

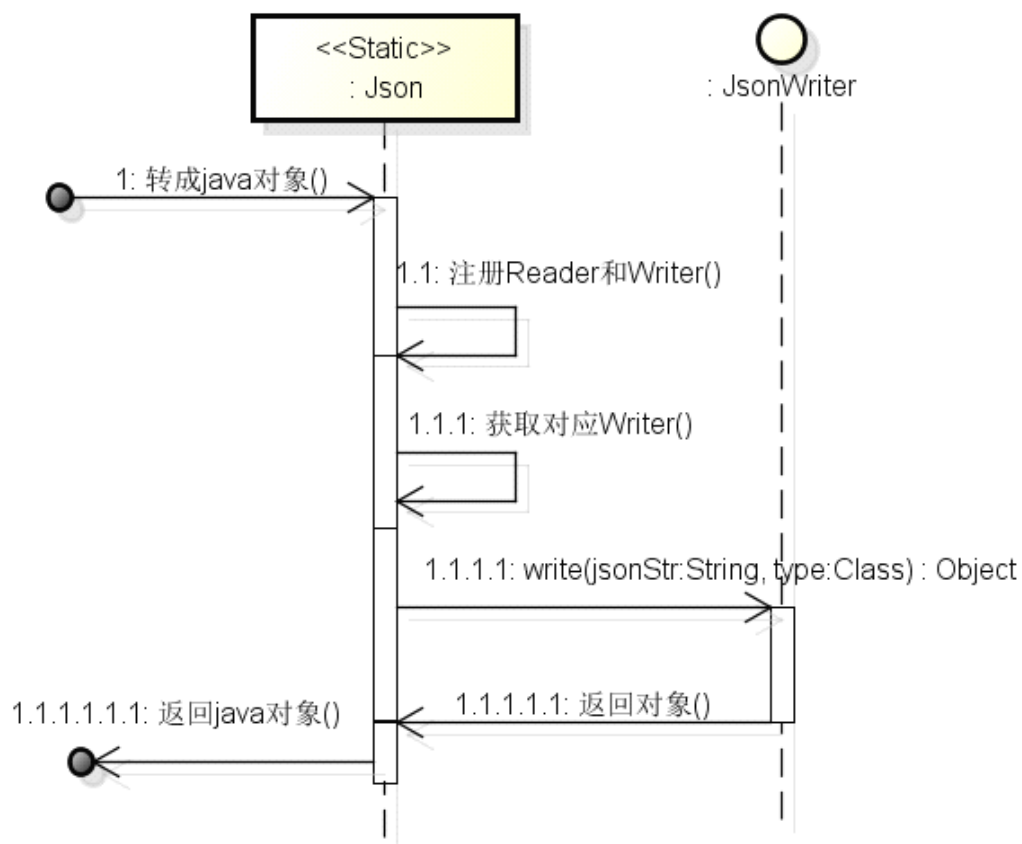


图 4-3-3 json 字符串转 java 对象

可以看到, jsonlib 被设计得非常简单, 与此同时也给了开发者很大的自由度, 开发者可以灵活的进行扩展 JsonLib 的功能, 这已经能够满足本项目的需求。

## 4.4 Ajax 基础服务设计与实现

### 4.4.1 背景

为了将 Ajax 单独抽象出来, 提高系统的模块化程度, 提供系统的可维护性和扩展性, 也是为了最小化满足系统的 Ajax 需求, 设计了适合于本系统的 Ajax 服务。本服务是以模块作为服务划分的, 要访问本服务, 就必须指定模块和对应的服务名称。系统在初始化的时候会注册相应的模块及其有关 Ajax 的服务。

### 4.4.2 Ajax 服务设计

本次设计的方案是使用一个核心的 Servlet 作为 Ajax 请求的分发器, 将前端发出的请求经过解析分发给特定的模块中具体的服务进行处理, 最终返回的是 json 字符串数据。也就是说本项目是依赖于前面 Jsonlib 项目的。(关于本项目可在 github 上面浏览: <https://github.com/cianfree/cdesign/tree/master/cdajax>)。

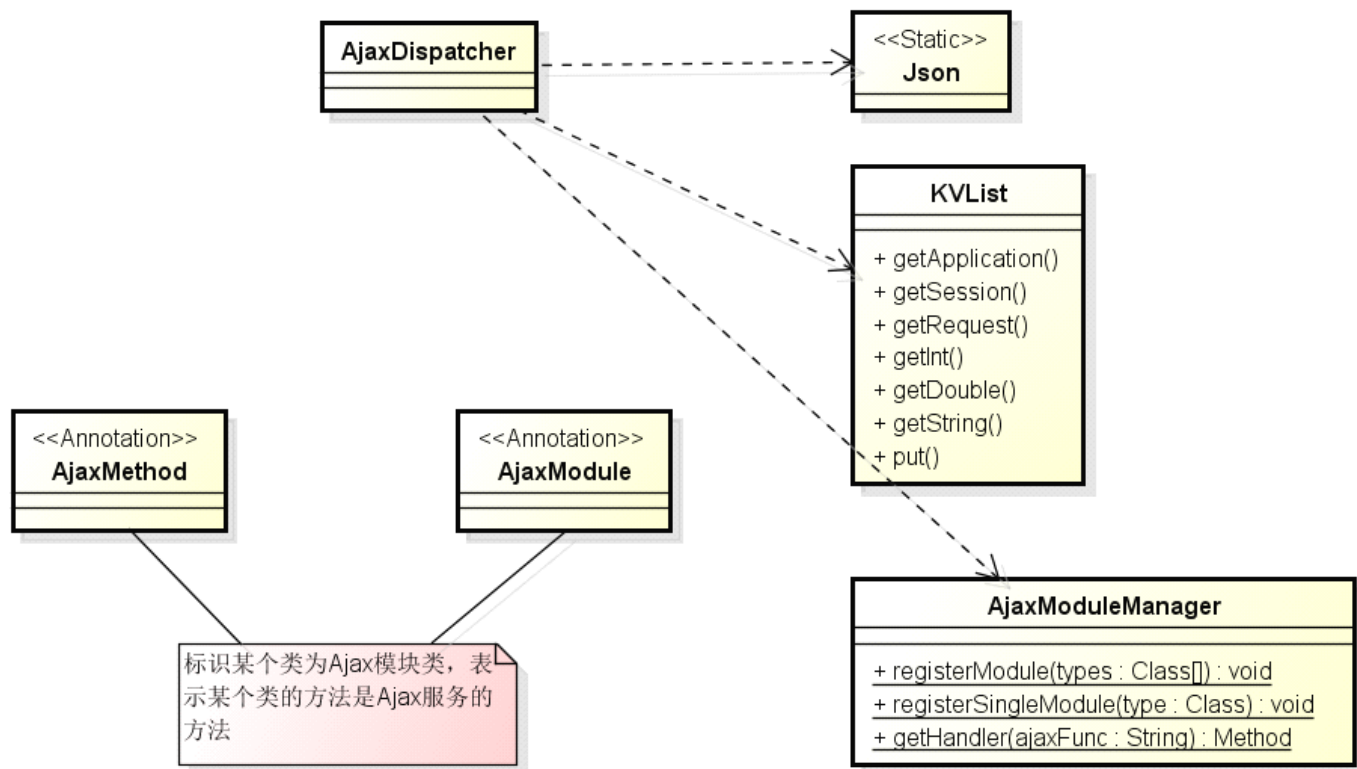


图 4-4-1 Ajax 服务类设计

表 4-4-1 Ajax 组件职责说明

组件	功能描述
AjaxDispatcher	Ajax 核心分发器，根据请求解析出处理该 ajax 服务的模块和对应模块内的服务
AjaxModuleManager	注册系统 ajax 模块和获取相应的 ajax 服务
KVLlist	存储请求相关的信息，key-value，request，Session，Applciation；但不包含文件相关的信息。
AjaxModule	标识一个类是 Ajax 组件类，是一个 Annotation
AjaxMethod	表示一个方法是 Ajax 服务方法，是一个 Annotation

具体代码实现就不说了，可以参考 GitHub 上面的项目源代码，下面介绍 Ajax 的启动流程和处理流程。

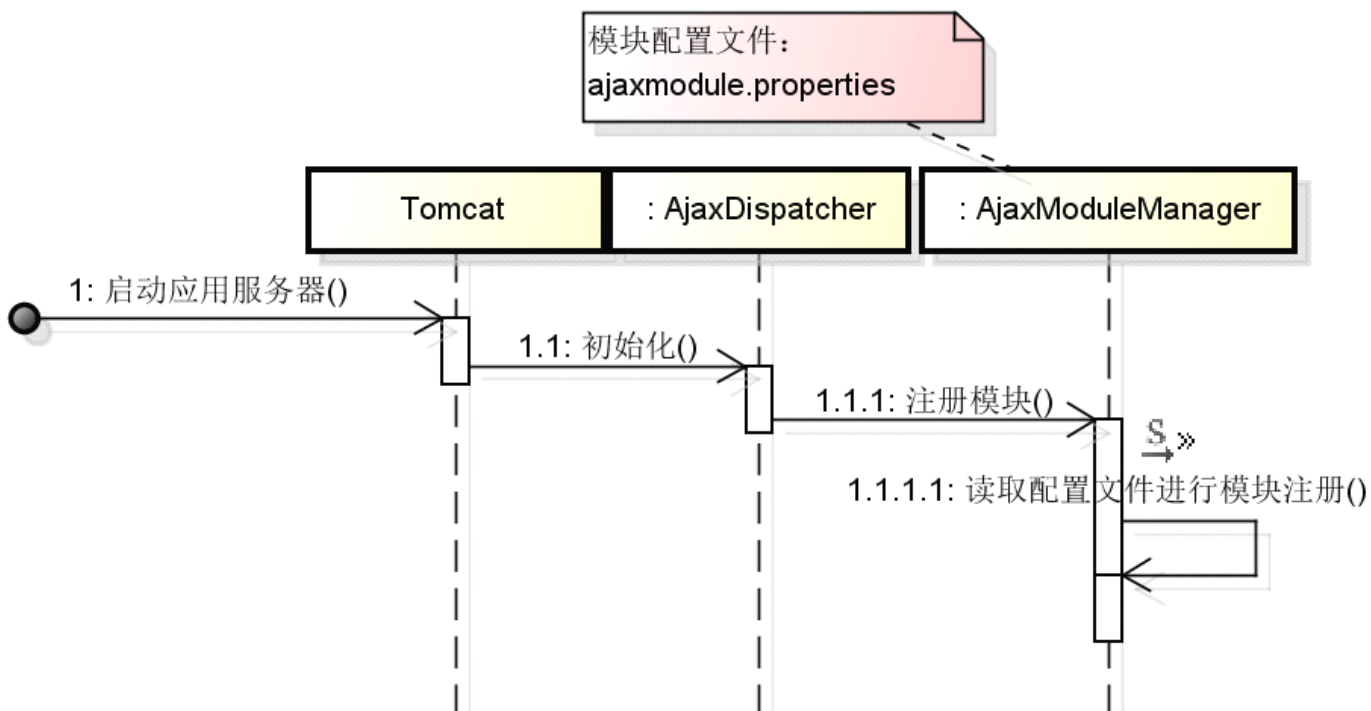


图 4-4-2 Ajax 服务启动流程

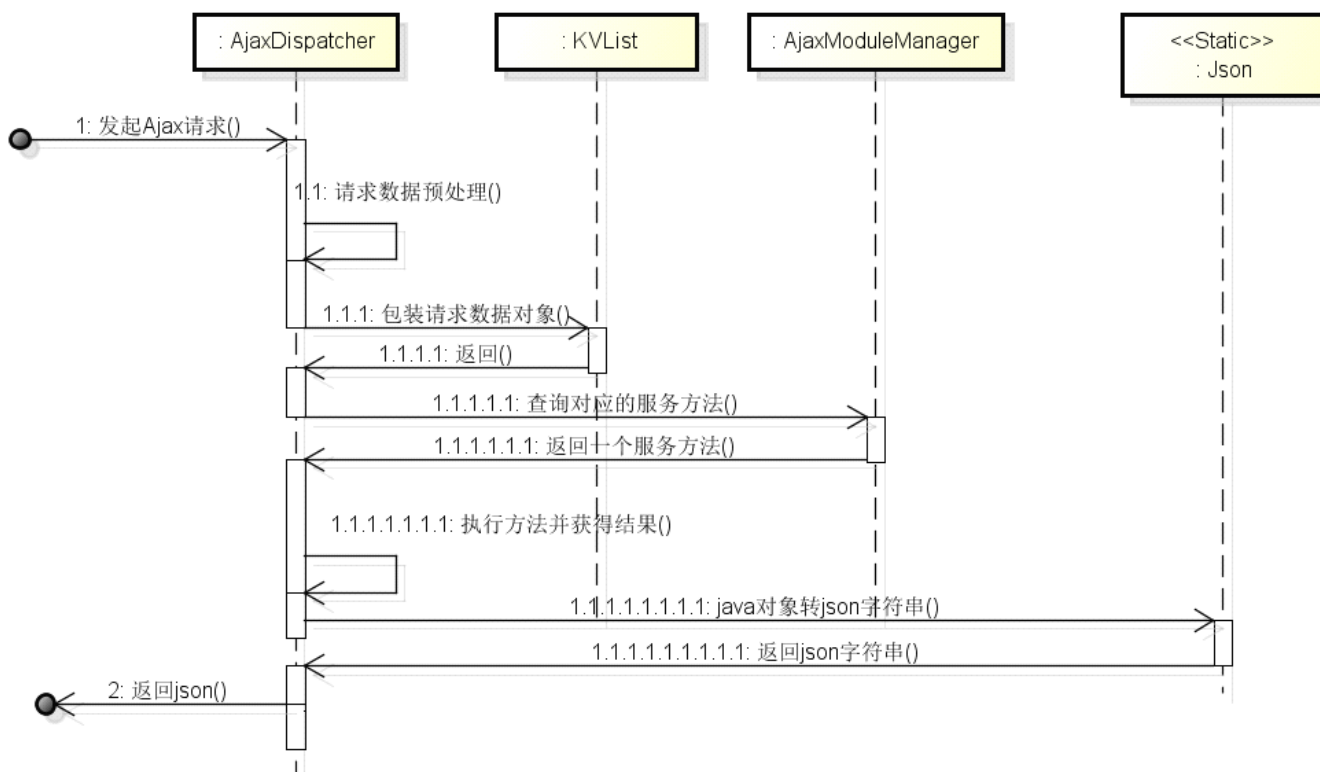


图 4-4-3 Ajax 请求处理流程



## 4.5 日志服务设计与实现

本系统并不打算使用第三方的日志工具，以自己设计了一个简单的日志记录工具，可以通过配置文件进行简单的配置（关于本项目可在 [github](https://github.com/cianfree/cdesign/tree/master/cdlog) 上面浏览：<https://github.com/cianfree/cdesign/tree/master/cdlog>）。

可配置的选项如下：

表 4-5-1 日志配置项

配置项	描述
log.level	日志输出级别，五个级别[debug, info, warn, error, fatal]，默认是 fatal，即把 fatal 以下的日志全部输出。
log.target	日志输出目标，可选项[console, dir]，即可以输出到控制台或文件中，默认是 console
log.target.dir	日志输出目录，可选项：当前应用目录[user.dir]，使用绝对的物理路径[如：/home/arvin/logs]
log.date.pattern	日期格式化，默认是 yyyy-MM-dd HH:mm:ss

日志工具总体类图，如图 4-5-1 所示：

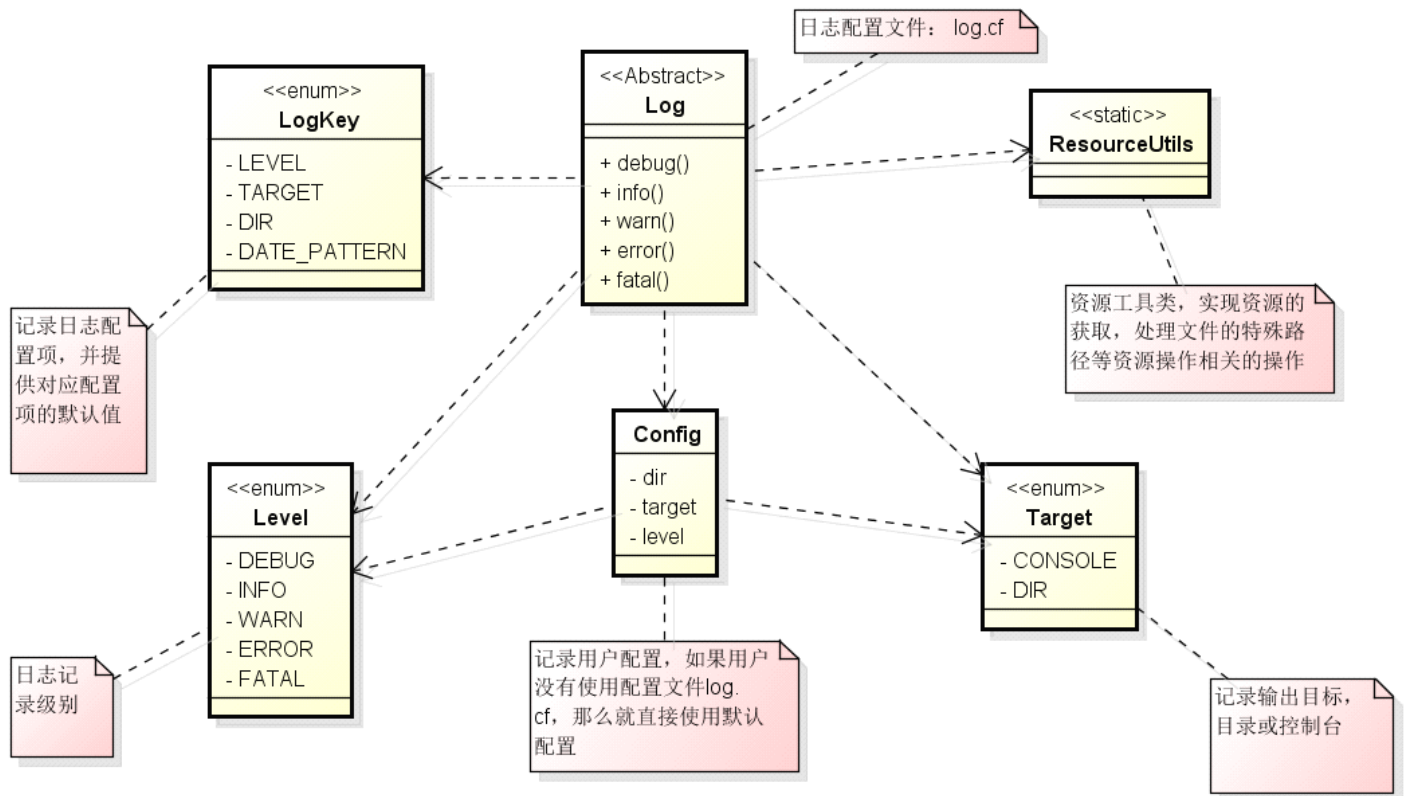


图 4-5-1 日志工具类图

配置文件示例：log.cf （classpath 路径下才可生效）

```

# 日志输出级别，五个级别[debug, info, warn, error, fatal]
log.level=fatal
# 日志输出目标，可选项[console, dir]
log.target=console
# 日志输出目录，可选项：当前应用根目录[user.dir]，自定义绝对路径[如D:/poj/log]
log.target.dir=user.dir
# 日期格式化
log.date.pattern=yyyy-MM-dd HH:mm:ss
  
```

#### 4.6 数据访问设计与实现

在本系统中，选择了 Hibernate 作为数据持久化方案，关于 Hibernate 的相关技术这里不再赘述，请查看 [3.2 Hibernate](#)。

对于数据的持久化，最基本的莫过于增删改查了，也就是平常所说的 CRUD。另外，分页功能也是非常的常用，如果对每个实体类都相应的设计一个 Dao 接口，不仅费时费力，而且使用起来也非常不方便，扩展性和重用性都很差。基于

此，在本系统中，对数据操作进行进一步的封装，使得更加方便扩展和使用。以下便是 DAO 的抽象设计：

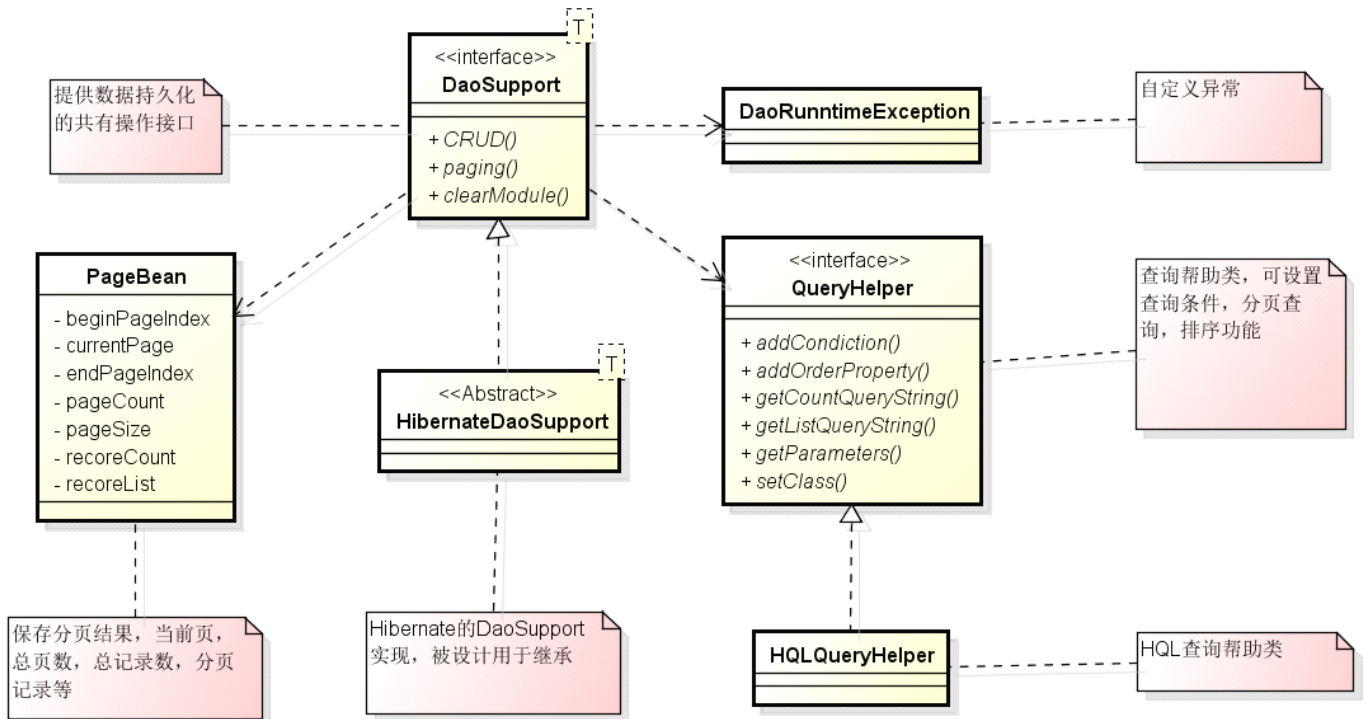


图 4-6-1 DAO 类图

经过这样的设计之后，关于 DAO 的实现就变得非常简单，而且扩展性能也好，重用性非常高。由于逻辑简单在本系统中没有特别使用 Dao 层，直接使用了 Service 层，即没有设 DAO，例如我们进行 User 类的数据库访问，只需设计一个对应的 UserService 及其实现类 UserServiceImpl 即可。经过上面的设计之后，UserService 和 UserServiceImpl 就变得非常简单，请看下面的例子。

UserService:

```

public interface UserService extends DaoSupport<User> {
    // 这里可以定义关于User的特殊业务处理方法
}

```

ServiceImpl:

```

public class UserServiceImpl
    extends HibernateDaoSupport<User>
    implements UserService {
    // 这里给定特殊实现
}

```

}

在本系统中，DaoSupport 的继承体系如下：

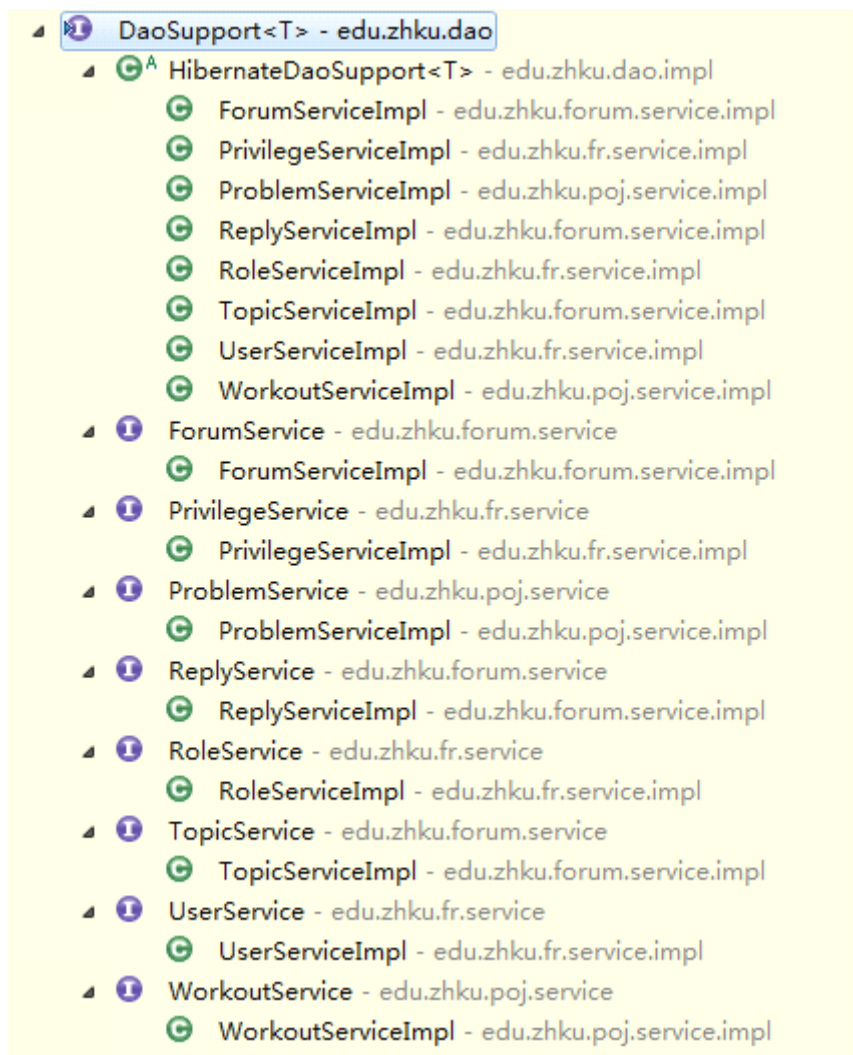


图 4-6-2 DAO 继承体系

## 4.7 分页插件设计与实现

### 4.7.1 分页概述

在前端，在许多场景中都需要对数据进行分页，在网上也有很多 javascript 分页的库。有些是纯 javascript 的，有些是基于 jQuery 的，有免费的也有收费的。但是，网上这些分页工具大都功能太强大，在本系统中，根本不需要那么多的功能，在本系统中只需要进行简单的分页就行，甚至不需要什么 Ajax 分页，基于这个目的，于是在 jQuery 的基础上设计了一个分页的插件（具体实现可以

在 GitHub 上面找到：<https://github.com/cianfree/cdesign/tree/master/pagingplugin>）。

目录结构：

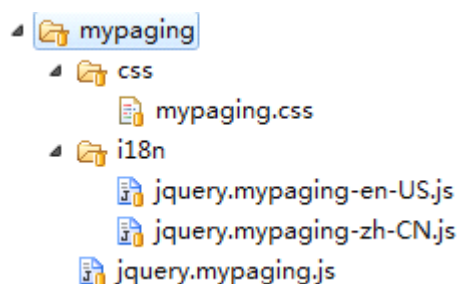


图 4-7-1 分页插件目录结构

先从调用形式说起：

```
<html>
<head>
<script type="text/javascript" src="jquery.js"/>
<!-- 引入自定义的分页插件 -->
<link href="css/mypaging.css" type="text/css" rel="stylesheet"/>
<script type="text/javascript" src="jquery.mypaging.js"/>
<script type="text/javascript">
    $(function() {
        jQuery("#pagingDiv").myPaging({
            currentPage: 5, // 当前页
            pageCount: 30, // 总页数
            pageSize: 10, // 每页显示条数
            totalRecord: 300, // 总记录数目
            showSize: 10, // 分页栏显示的页码个数
            callback: function(currentPage, pageSize) {
                // 回调函数，当点击页码的时候就会执行这个函数
            }
        });
    })
</script>
</head>
<body>
<div id="pagingDiv"></div>
</body>
</html>
```

效果图：

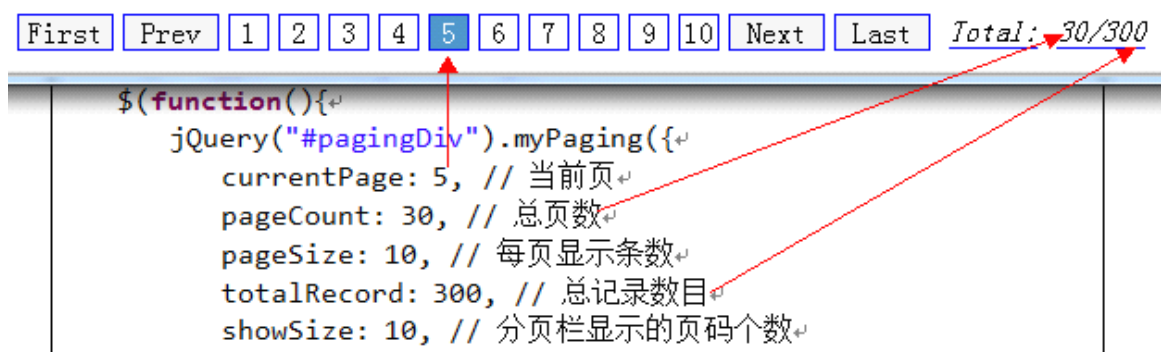


图 4-7-2 分页插件演示

### 4.7.2 关键算法实现

设计该分页插件的具体实现思路就是，在给定的 div 上面根据分页相关参数添加页码按钮，显示当前页码，添加页面按钮的响应事件，如上一页，第一页，首页，末页。当点击的时候就会返回当前页和每页显示的距离数目。

在显示分页信息的时候，有以下几种情况：

表 4-7-1 分页显示情况表

Current Page	Show Size	期望显示
当 pageCount=20, pageSize=10, totalRecord=200 时候		
-1 [ $\leq 0$ ]	8	<a href="#">First</a> <a href="#">Prev</a> <a href="#">1</a> <a href="#">2</a> <a href="#">3</a> <a href="#">4</a> <a href="#">5</a> <a href="#">6</a> <a href="#">7</a> <a href="#">8</a> <a href="#">Next</a> <a href="#">Last</a> <a href="#">Total: 20/200</a>
21 [ $> 20$ ]	8	<a href="#">First</a> <a href="#">Prev</a> <a href="#">13</a> <a href="#">14</a> <a href="#">15</a> <a href="#">16</a> <a href="#">17</a> <a href="#">18</a> <a href="#">19</a> <a href="#">20</a> <a href="#">Next</a> <a href="#">Last</a> <a href="#">Total: 20/200</a>
10	8	<a href="#">First</a> <a href="#">Prev</a> <a href="#">7</a> <a href="#">8</a> <a href="#">9</a> <a href="#">10</a> <a href="#">11</a> <a href="#">12</a> <a href="#">13</a> <a href="#">14</a> <a href="#">Next</a> <a href="#">Last</a> <a href="#">Total: 20/200</a>
当 pageCount=5, pageSize=10, totalRecord=50 时候		
-1 [ $\leq 0$ ]	8[ $> 5$ ]	<a href="#">First</a> <a href="#">Prev</a> <a href="#">1</a> <a href="#">2</a> <a href="#">3</a> <a href="#">4</a> <a href="#">5</a> <a href="#">Next</a> <a href="#">Last</a> <a href="#">Total: 5/50</a>
6[ $> 5$ ]	8[ $> 5$ ]	<a href="#">First</a> <a href="#">Prev</a> <a href="#">1</a> <a href="#">2</a> <a href="#">3</a> <a href="#">4</a> <a href="#">5</a> <a href="#">Next</a> <a href="#">Last</a> <a href="#">Total: 5/50</a>
2	8[ $> 5$ ]	<a href="#">First</a> <a href="#">Prev</a> <a href="#">1</a> <a href="#">2</a> <a href="#">3</a> <a href="#">4</a> <a href="#">5</a> <a href="#">Next</a> <a href="#">Last</a> <a href="#">Total: 5/50</a>

经过仔细分析得知，该分页插件的难点在于如何计算显示页码的开始和结束

下标，以下是结算开始和结束下标的算法：

```
// 计算begIndex和endIndex
var indexAdapterFlag = false;
function indexAdapter() {
    // showSize=10: 偶数, 左4=10/2-1, 中间选中, 右5=10/2
    // showSize=5: 奇数, 左2=(5-1)/2, 中间一个, 右2=(5-1)/2

    indexAdapterFlag = true;
    // 计算showSize
    var showSize = getShowSize();
    var leftSize,    // 在当前页左边显示几个页码
        rightSize;  // 在当前页右边应该显示几个页码
    if(showSize % 2 == 0) {
        leftSize = showSize/2 - 1;
        rightSize = leftSize + 1;
    } else {
        leftSize = (showSize - 1) / 2;
        rightSize = leftSize;
    }

    if(getCurrentPage() - leftSize <= 0) {
        begIndex = 1;
        rightSize += (leftSize - getCurrentPage() + 1);
    } else {
        begIndex = getCurrentPage() - leftSize;
    }

    if(getCurrentPage() + rightSize > getPageCount()) {
        endIndex = getPageCount();
        begIndex -= (getCurrentPage() + rightSize - getPageCount());
    } else {
        endIndex = getCurrentPage() + rightSize;
    }
    if(begIndex <= 0) begIndex = 1;
    if(endIndex > getPageCount()) endIndex = getPageCount();
}
```

源代码位于 GitHub 上面：

<https://github.com/cianfree/cdesign/tree/master/pagingplugin>

## 5 模块分析

### 5.1 模块划分

在本次项目开发中，本系统将构建一个基础应用模块和扩展的两个模块。如下图：

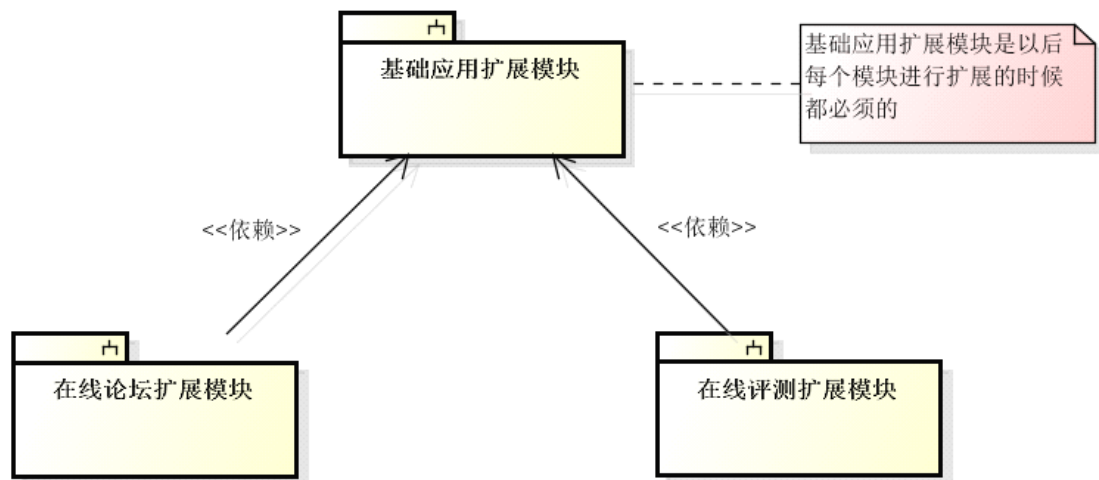


图 5-1-1 系统模块图

### 5.2 模块功能

在第四节中讨论了系统基础架构设计，现在可以进一步设计系统所需要的功能的，经过前面的分析，本次开发需要扩展出三个模块，接下来对这三个扩展模块进行详细说明。

#### 5.2.1 基础应用扩展模块

本模块是所有其他扩展模块的基础，是本系统框架的第一个扩展模块。提供了系统的相关通用的管理功能，如用户管理，角色管理，个人信息管理。下面对该模块的功能进行详细的描述。



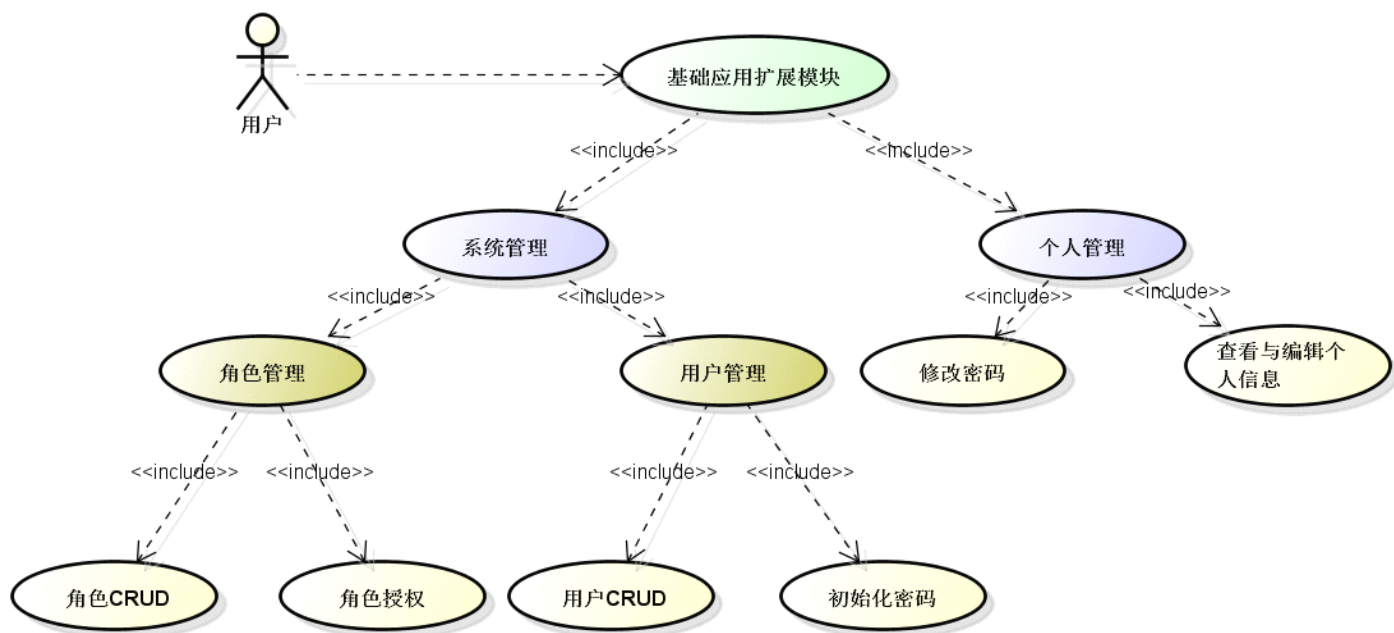


图 5-2-1 基础应用扩展模块功能

### 5.2.2 在线评测扩展模块

在线评测系统扩展模块是本次系统开发的核心功能模块之一，功能稍微复杂一些，主要提供在线评测和辅助教学功能。

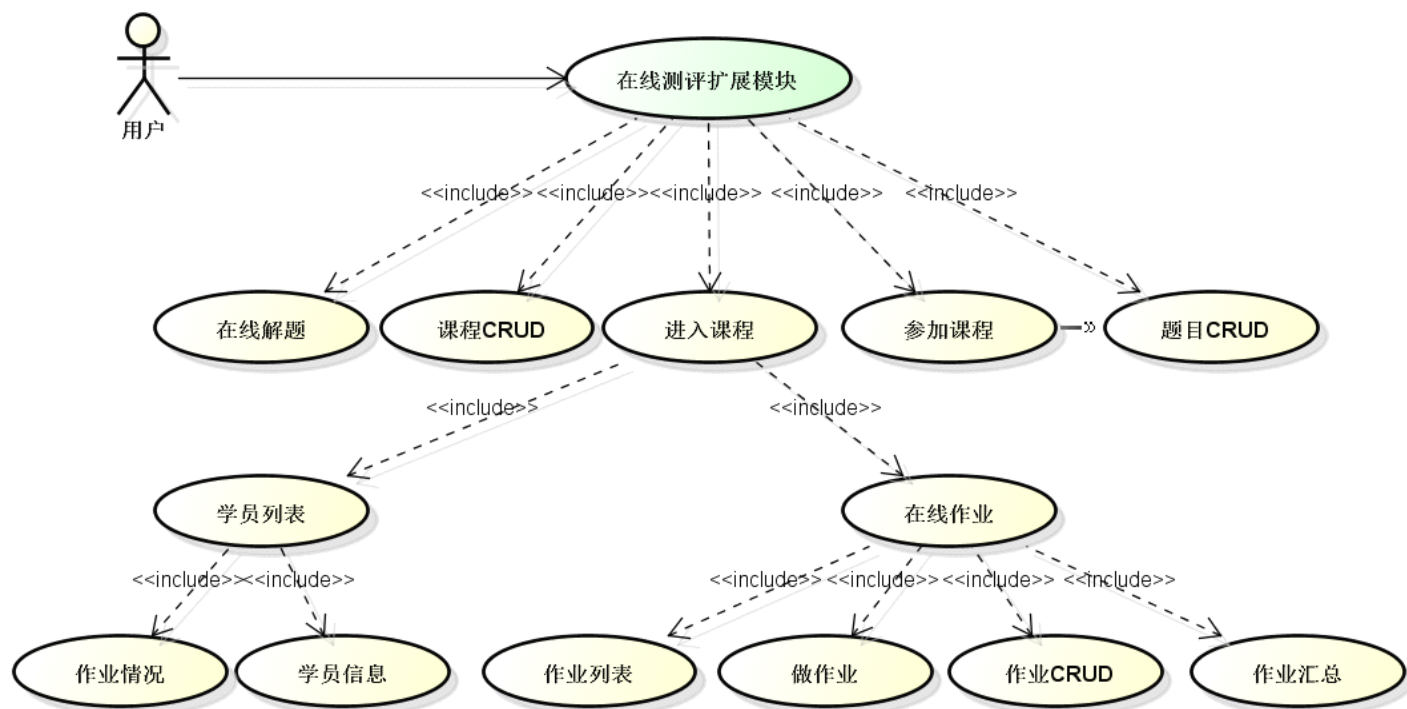


图 5-2-2 在线评测模块功能

5.2.3 在线论坛扩展模块

在线论坛扩展模块是为了提供一个平台让大家可以相互之间进行交流，论坛的功能和当前网络上面的论坛没有太多的区别，相对比较简单，请看本系统论坛的功能明细：

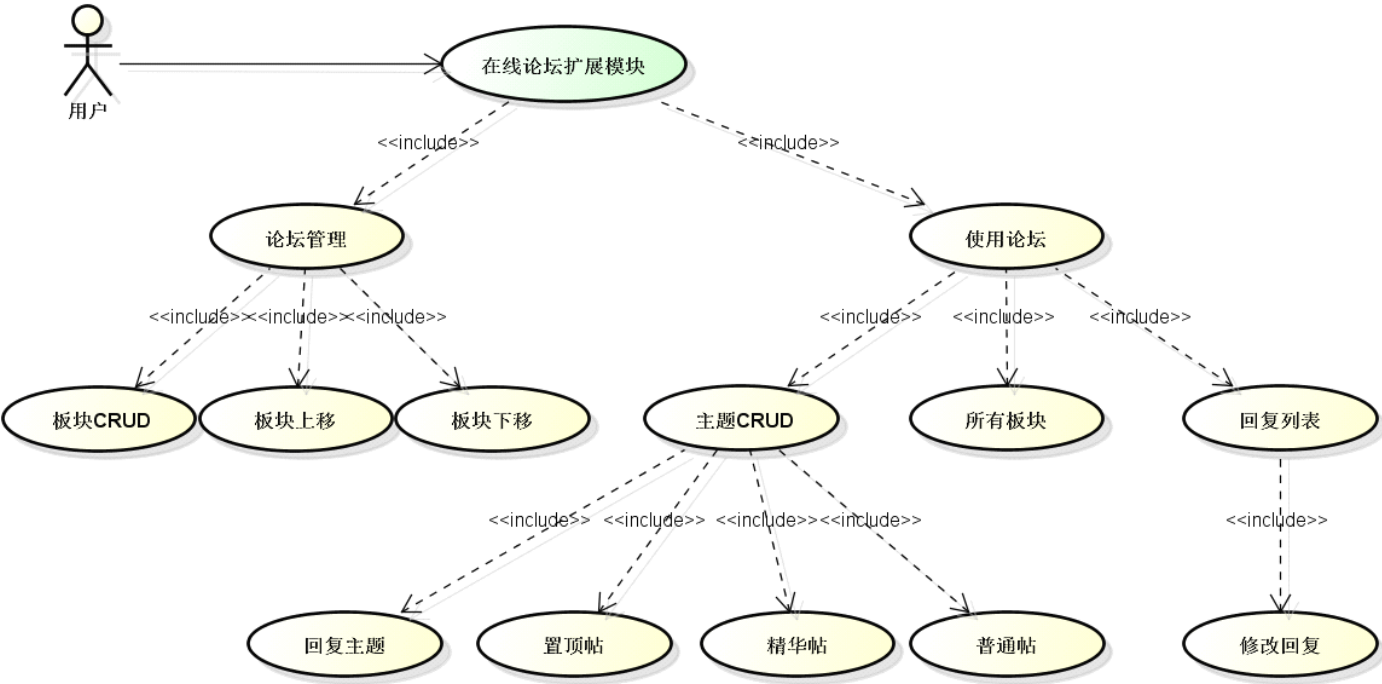


图 5-2-3 在线扩展模块功能

5.3 领域模型分析

5.3.1 系统参与者

本系统为可扩展的在线测评系统，在可扩展的框架之上，本次扩展了三个模块，分别是基础管理模块，在线评测模块和论坛模块。系统外部参与者主要有：

- 超级管理员
- 普通用户

- 学生
- 教师

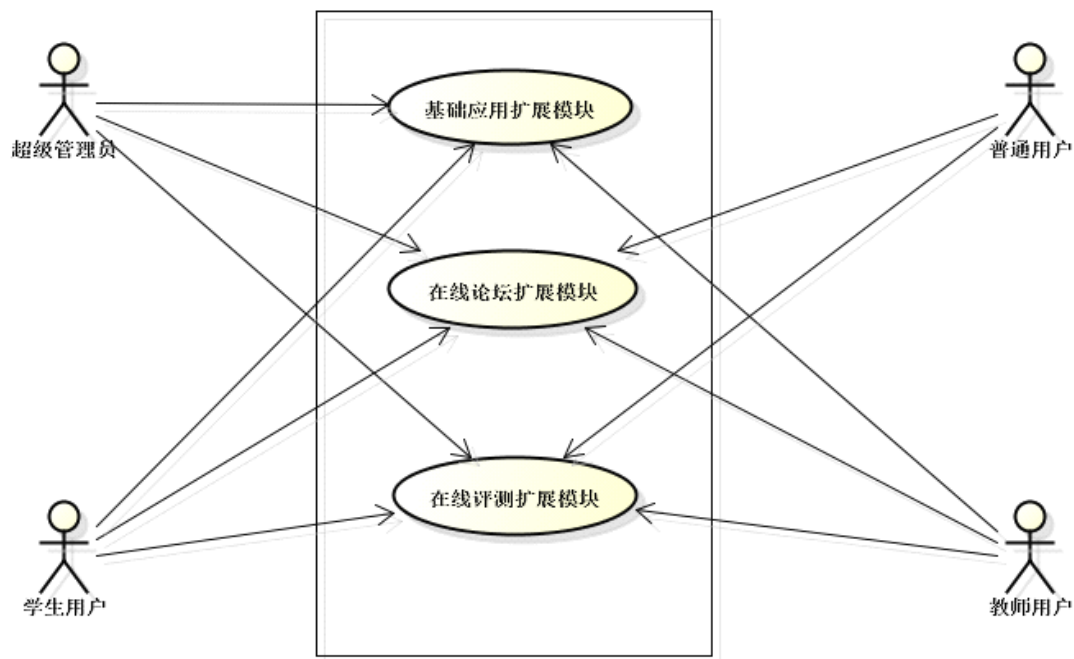


图 5-3-1 顶层用例

### 5.3.2 参与者权限分配

在本系统中，所谓的参与者，其实从根本上来说还是用户，因为使用的是基于角色的权限控制，所以在本系统中只有用户的概念，其余的都是定义的角色。本系统中超级管理员是没有权限限制的。学生，教师是系统自定义的角色，超级管理员可以给角色分配权限。下面就不同的角色进行权限的分配。

#### (1) 普通用户权限初定义

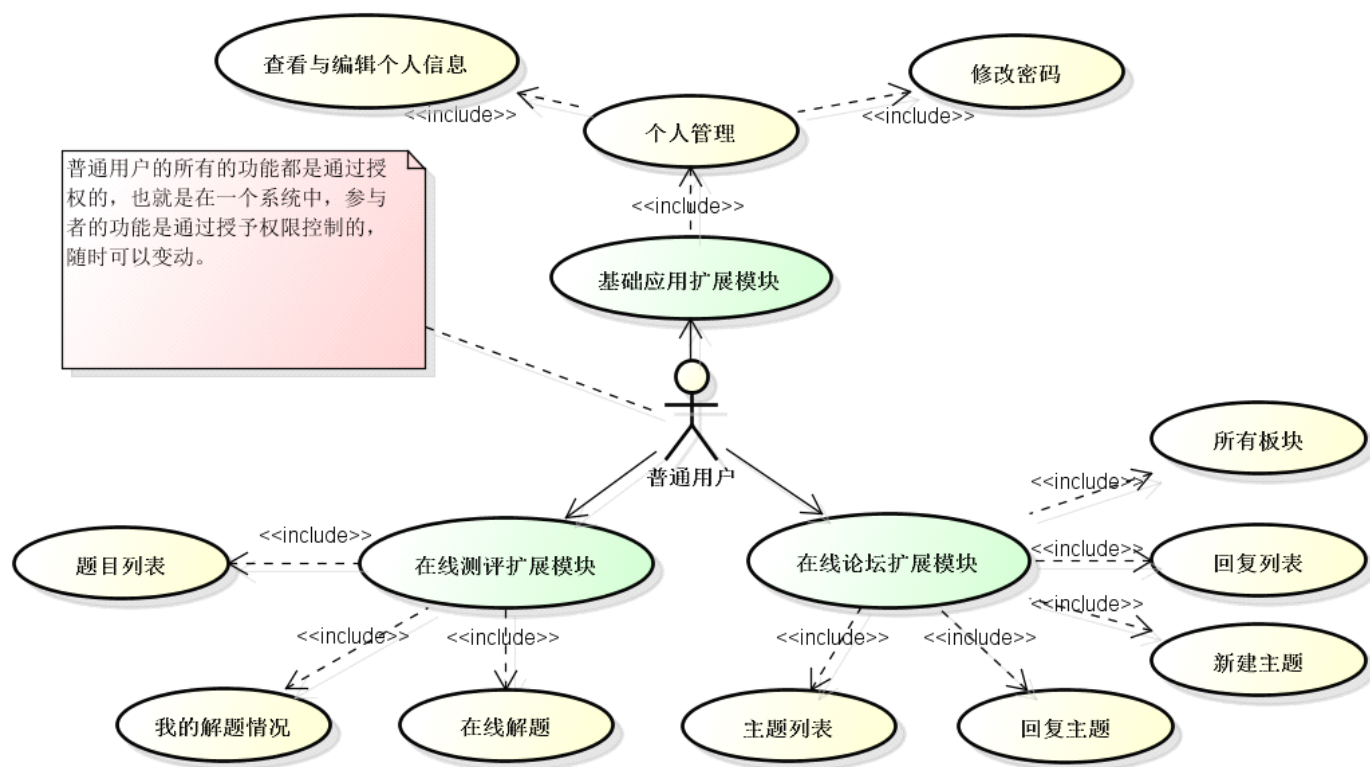


图 5-3-2 普通用户功能

## (2) 学生用户权限初定义

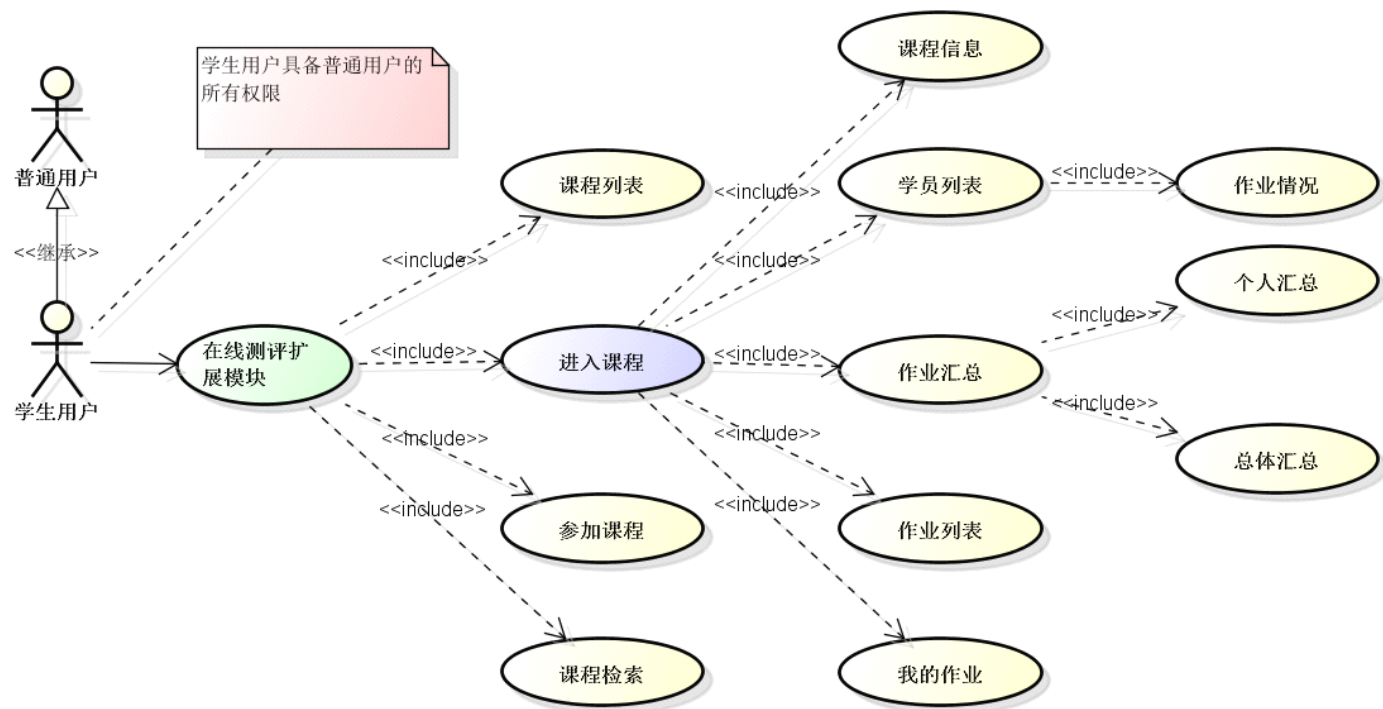


图 5-3-3 学生用户功能

### (3) 教师用户权限初定义

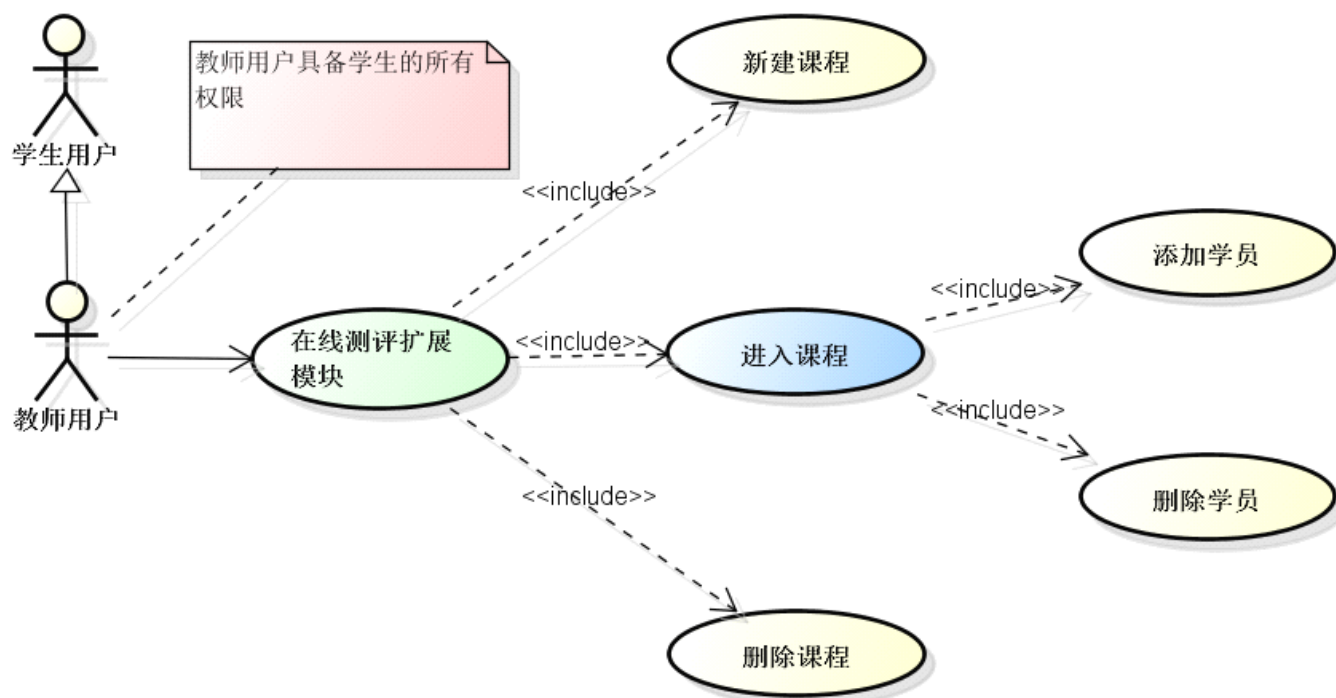


图 5-3-4 教师用户功能

## 6 模块详细设计与实现

### 6.1 系统体系架构

#### 6.1.1 系统体系架构

本项目的扩展模块均使用 SSH（SpringMVC+Spring+Hibernate）体系架构进行设计和实现。在进行 web 系统开发的时候，使用一个良好的体系结构能提高 web 系统的扩展性和可维护性，同时能提高系统开发的效率，利于项目任务划分。本项目就 web 系统开发所面临的问题，结合当前流行的 web 开源框架 Spring，Hibernate，将 SSH 应用到项目中来。

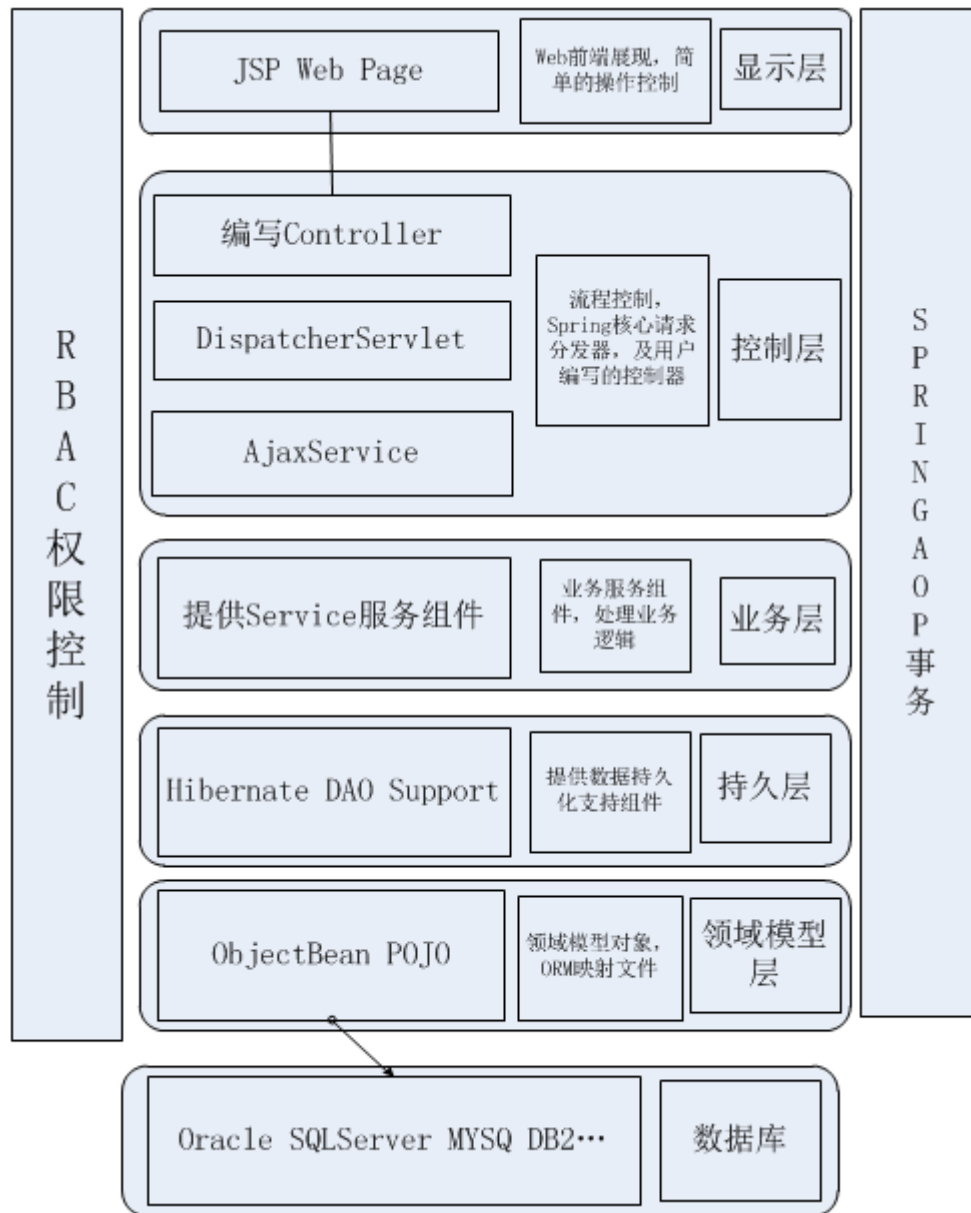


图 6-1-1 SSH 应用到项目系统架构图

## 6.1.2 职责分配

### (1) SpringMVC 负责 WEB 层

SpringMVC 的核心分发器接收到前端的表单提交的请求，然后通过映射关系，分发给 Controller 进行请求的处理，所有的 Controller 都使用 Annotation 进行标识，要处理请求的方法也经过 Annotation 进行标识，SpringMVC 启动的时候就会处理这些映射关系。

## (2) Spring 负责业务层的管理

Service 提供了统一的调用接口，封装了数据的持久化操作，实现业务逻辑，并集成了 Hibernate，利用 Spring 的声明式事务处理来统一管理 Hibernate 的事务，对数据的 ACID 属性进行有效控制。

## (3) Hibernate 负责数据持久化

Hibernate 通过一组 xxx.hbm.xml 文件和 POJO 对象，建立起对象-关系映射，使得与数据库中的表一一对应，并且能方便的屏蔽掉数据库之间的差异，提供一个 HQL 进行统一查询，方便数据库的迁移。本项目更是在此基础之上，构建了更为灵活的数据访问，见 [4.6 数据访问设计与实现](#)。

### 6.1.3 结合 SSH 的 MVC 模型

MVC 模式在 UI 设计中使用得非常普遍，这个模式的特点是：分离了模型，视图，控制器三种角色，将业务从 UI 设计中独立出来，封装到模型和控制器的设计上面去，使得它们之间相互解耦，可以独立扩展而不彼此依赖。

从整体上看，在使用 SpringMVC 的时候，需要在部署描述符（web.xml）中配置 DispatcherServlet，这可以看成是一个前端控制器的具体实现，另外需要在 Bean 中配置 Web 请求和 Controller（控制器）的关系，可以使用 xml 和 Annotation 两种方式进行关联，最后还要处理各种视图的展现方式。在具体使用 Controller 的时候，会看到 ModelAndView 数据的生成，还能看到 ModelAndView 把数据交给相应的 View（视图）来进行呈现。

在 SSH 的基础上，为了更好的扩展，设计了更为灵活的数据访问，现在就来看看扩展的 SSH 是如何实现 MVC 这种设计模式的。

首先，来看 SpringMVC 的处理流程是如何的：



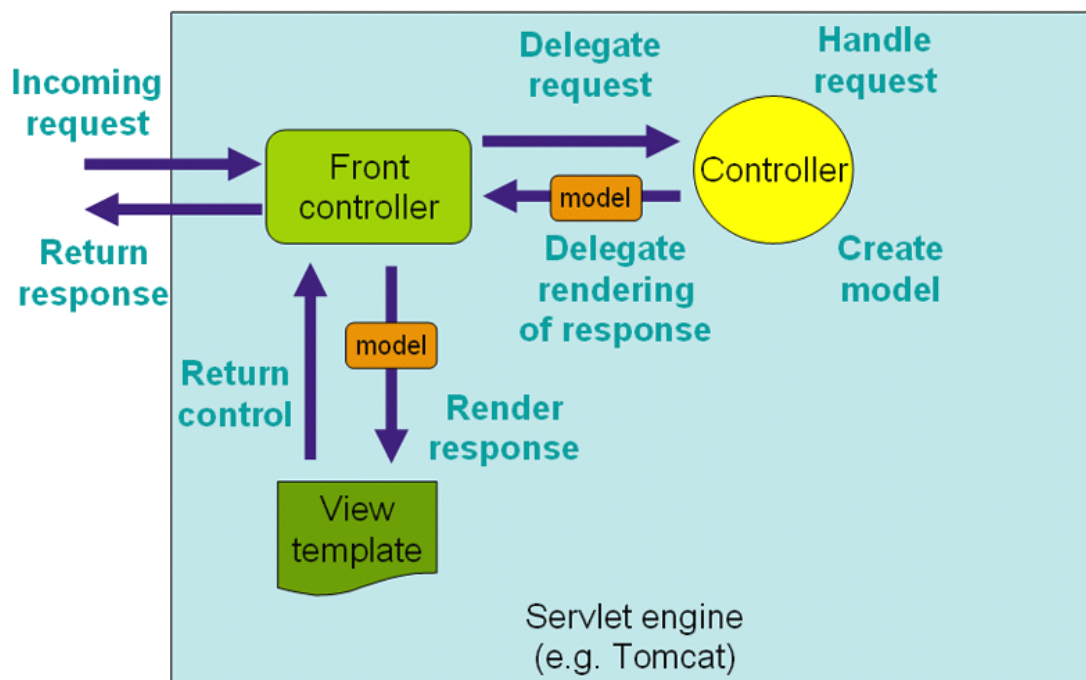


图 6-1-2 SpringMVC 工作流处理

经过前面的介绍，大致了解了 SpringMVC，接下来看看 SSH 的一个整体工作流程是如何的。

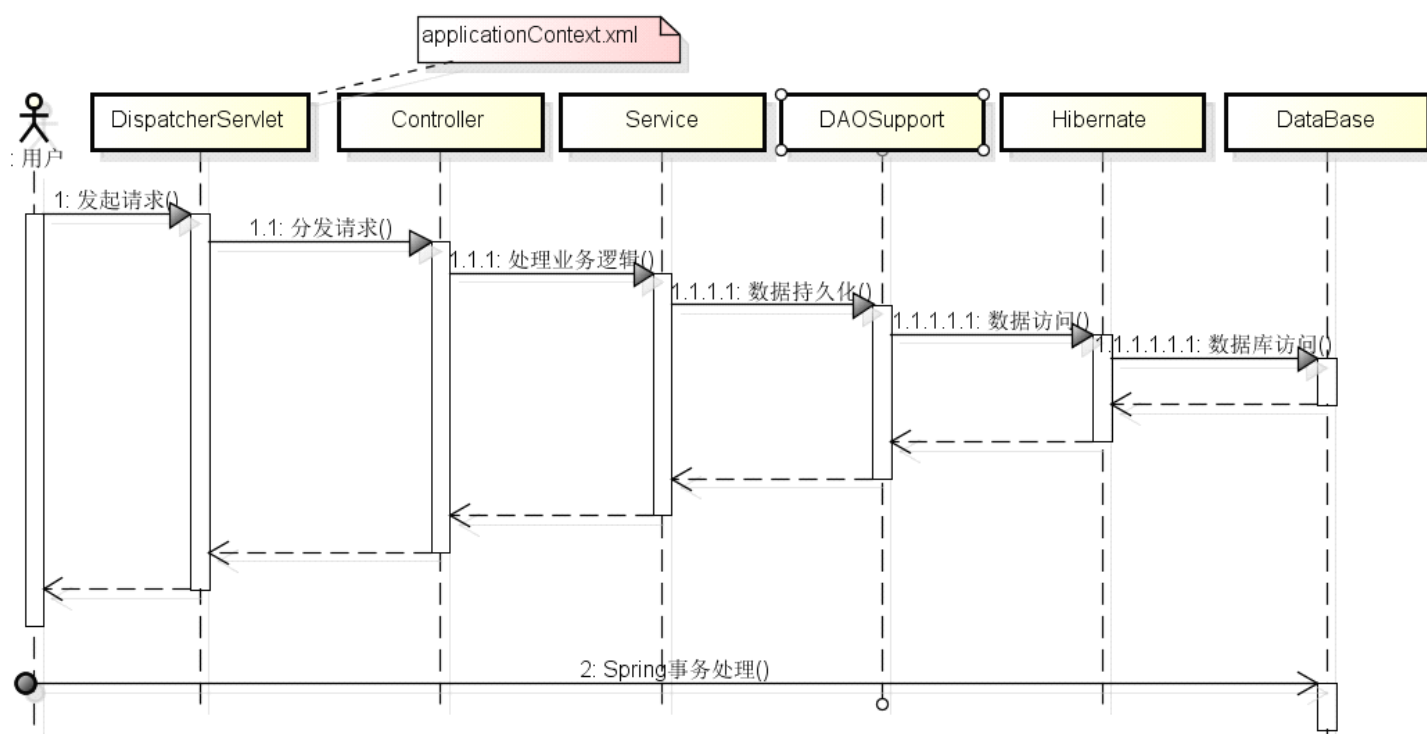


图 6-1-3 MVC 序列图



#### 6.1.4 SSH 工作流程说明

应用服务器（如 Tomcat）启动时，应用服务器会去读取部署描述符 web.xml，初始化相关资源，如初始化 Spring 核心容器 DispatcherServlet，DispatcherServlet 就会读取 applicationContext.xml 来初始化 SpringMVC 环境。记录请求与 Controller 的对应信息等。

当 Web 浏览器向服务器发起请求的时候，DispatcherServlet 就会根据请求信息找到相应的 Controller，执行 Controller 的对应处理方法处理本次请求，此时 Controller 就是去调用业务层的组件 Service，Service 通过 Hibernate 访问数据库，对数据进行相关的业务处理，然后把请求的结果数据封装到一个 ModelAndView 的类中。ModuleAndView 就会把数据提交给 Spring，让 Spring 找到相应的视图解析器去展现数据给用户，本系统使用的就是 jsp 来显示。JSP 动态进行数据的渲染，最后把结果返回 Web 浏览器。

## 6.2 基础应用扩展模块

### 6.2.1 领域模型分析

基础应用扩展模块的功能就好像基础设施一般，没有它不行，所有的其他扩展模块都要依赖于本模块。本模块主要是做一些管理的工作，如用户的管理，角色的管理，角色的授权，个人信息的管理，这些都是基础性服务。

关于本模块的功能，详见 [5.2.2 基础应用扩展模块](#)。

从上面的分析可知，本模块涉及到的领域模型有：用户，角色，权限，以下是这三者的类图。

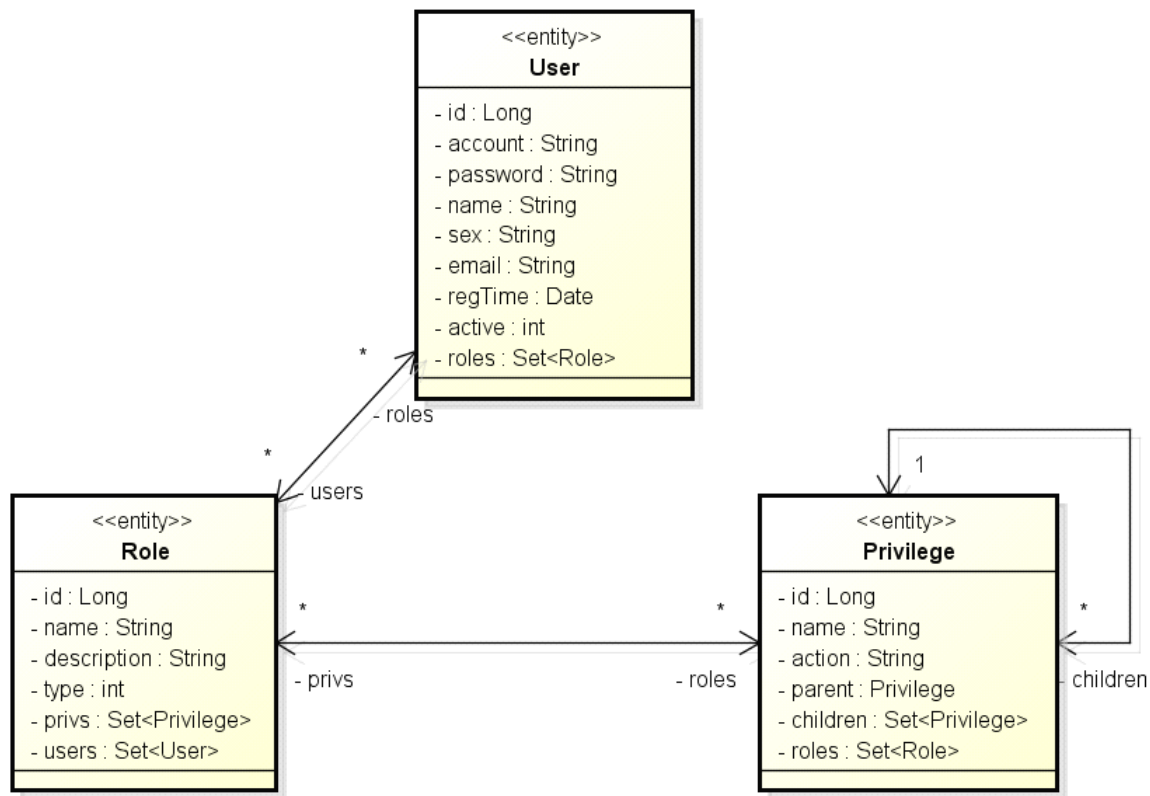


图 6-2-1 领域模型类图

## 6.2.2 视图与控制层设计

视图通常是与某个 Controller 联系在一起的，用户发起请求，后台经过数据处理返回用户一个页面。从 [5.2.2 基础应用扩展模块](#) 分析可以进行以下的界面设计。所有的 Controller 都继承与 BaseController，BaseController 提供公共方法以及注入主要的组件，以便提供子类使用，简化代码，提高重用。

本模块有四个核心 Controller，分别是 HomeController，PersonalController，RoleController 和 UserController。它们的职责如下：

表 6-2-1 基础应用扩展模块控制器职责分配

控制器	职责
HomeController	主页面相关跳转控制，主页，登录，注销，没有权限页面，简介页面
PersonalController	个人设置相关控制器，如更新个人信息，修改密码

RoleController	角色信息管理控制器，如角色 CRUD 以及角色授权
UserController	用户相关控制器，如用户的 CRUD，用户注册，发送激活邮件，初始化密码等

控制器是由 Spring 的 IoC 容器进行管理的，因此使用起来非常的方便，你可以直接在方法中声明相关的参数，如 `HttpServletRequest`, `HttpSession`, `ModuleMap` 等，Spring 就会自动帮你注入这些参数，比起 Struts 来说这可要灵活许多。

表 6-2-2 基础应用扩展模块 Controller 映射

控制器	动作	RequestMapping	出口
HomeController	主页面	index.html home.html	home.jsp
	无权限	noPrivilege.html	noPrivipege.jsp
	简介	introduction.html	introduction.jsp
	注销	logout.html	logout.jsp
PersonalController	查看个人信息	personalInfo.html	editUserInfoUI.jsp
	更新个人信息	updateUserInfo.html	personalInfo.html
	修改密码页面	personalPassword.html	personalInfo.html
	修改密码	updateUserPassword.html	updateUserPwdSuccess.jsp editUserPasswordUI.jsp
RoleController	角色列表	roleList.html	roleList.jsp
	编辑角色	editRoleUI.html	editRoleUI.jsp
	保存角色	saveRole.html	roleList.html
	更新角色	updateRole.html	roleList.html
	角色授权	privTreeUI.html	roleList.html
UserController	用户登录界面	loginUI.html	loginUI.jsp
	登录	login.html	home.jsp
	注册页面	regUI.html	regUI.jsp
	注册	reg.html	regSuccess.jsp

			regUI.jsp
	用户激活	activate.html	home.jsp resendmail.jsp
	用户列表	userList.html	userList.jsp
	保存用户	saveUser.html	userList.html
	更新用户	updateUser.html	userList.html

### 6.2.3 业务组件设计

在 SSH 系统架构中，模型组件负责处理系统的业务逻辑，这些 Service 被 Spring 的 IoC 容器管理着，方便的处理依赖关系，并且所有的业务组件都是以接口方式出现，是完全面向接口编程，扩展性好。

表 6-2-3 组件职责

业务组件接口	组件默认实现	职责描述
PrivilegeService	PrivilegeServiceImpl	权限相关的业务处理
RoleService	RoleServiceImpl	角色相关业务处理
UserService	UserServiceImpl	用户相关业务处理
MailService	MailServiceImpl	邮件操作组件，主要是发送激活邮件

## 6.3 在线测评扩展模块

### 6.3.1 领域模型分析

本扩展模块是本系统的重要功能之一，详细的功能分析见 [5.2.2 在线测评扩展模块](#)。

从上面的功能分析中可以抽象出该模块具有以下几个领域模型：题目，课程，解题结果，作业，作业结果。他们之间的关联关系如图 6-3-1：

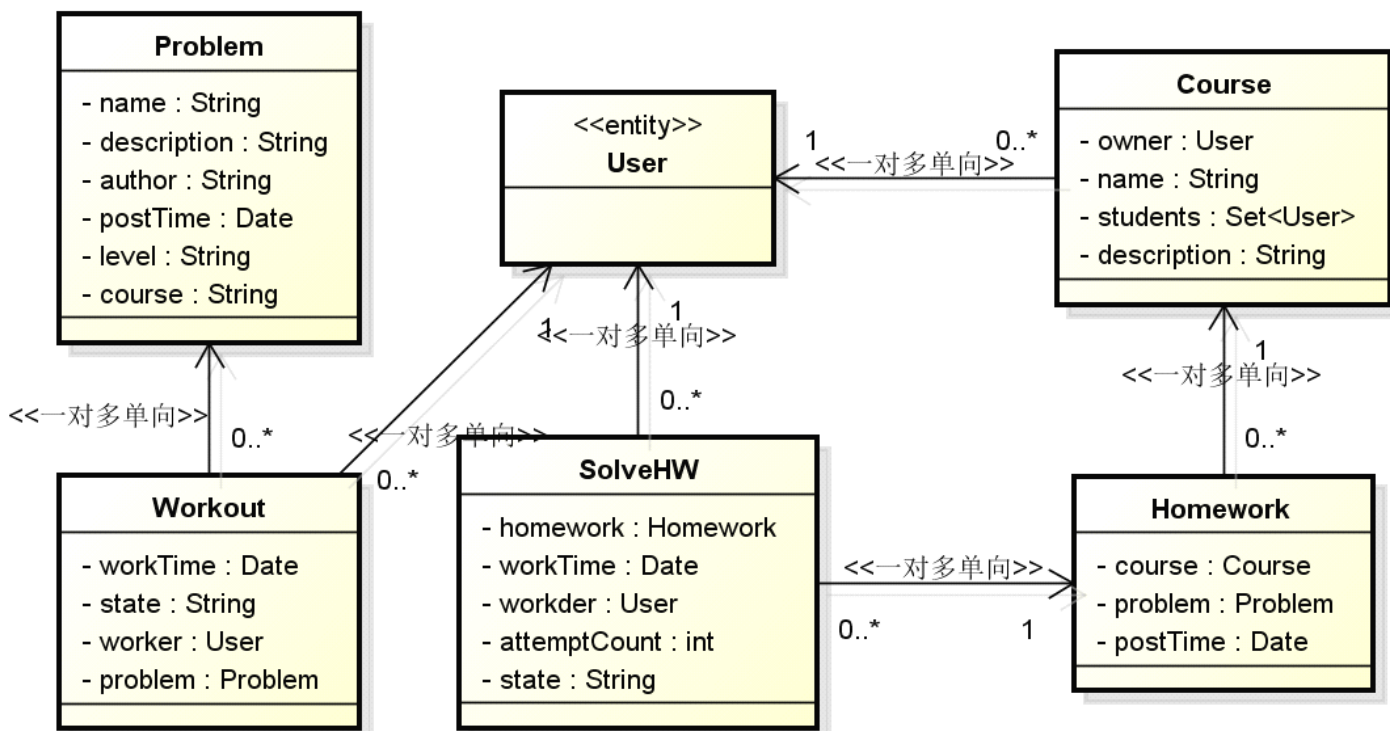


图 6-3-1 在线评测领域模型

### 6.3.2 视图与控制层设计

与前面一样，本模块同样还是使用 SSH 架构，在本模块中包含以下几个 Controller：ProblemController，WorkoutController，HomeworkController，CourseController。各个控制器的职责分配如下：

表 6-3-1 Controller 职责描述

控制器	职责
ProblemController	题目管理控制器
HomeworkController	作业管理控制器
CourseController	课程管理控制器

根据 SpringMVC 处理流程，可以确定 Controller 和页面之间的调用关系，如表 6-3-2：

表 6-3-2 在线评测模块 Controller 映射

Controller	动作	RequestMapping	出口
ProblemController	题目列表	problemList.html	problemList.jsp
	编辑题目	editProblemUI.html	editProblemUI.jsp
	保存题目	saveProblem.html	problemList.html
	更新题目	updateProblem.html	problemList.html
	删除题目	deleteProblem.html	problemList.html
	解题页面	solveProblemUI.html	solveProblemUI.jsp problemList.html
CourseController	课程列表	courseList.html	courseList.jsp
	课程信息	viewCourse.html	viewCourse.jsp
	编辑课程	editCourseUI.html	editCourseUI.jsp
	保存课程	saveCourse.html	courseList.html
	更新课程	updateCourse.html	courseList.html
	删除课程	removeCourse.html	courseList.html
	参加课程	joinCourse.html	viewCourse.html
	退出课程	quitCourse.html	viewCourse.html
	课程学生列表	courseStudentList.html	courseStudentList.jsp
	删除某课程学生	deleteCourseStudent.html	courseStudentList.html
	学生用户列表（全站）	studentList.html	studentList.jsp
	添加学生到课程	addStudentToCourse.html	courseStudentList.html
HomeworkController	作业列表	courseHomeworkList.html	courseHomeworkList.jsp
	添加作业	addCourseHomework.html	courseHomeworkList.html

	删除作业	removeCourseHomework.html	courseHomeworkList.html
	某道作业 作答情况	studentSolveList.html	studentSolveList.jsp
	某学生的 作业情况	studentHomeworkList.html	studentHomeworkList.jsp
	做作业	solveHWUI.html	solveHWUI.jsp

### 6.3.3 业务组件设计

本模块除了基本的 Service 组件之外,还有 Ajax 组件,Ajax 组件是不受 Spring 控制的,是一个独立的 Ajax 服务。详见 [4.4 Ajax 基础服务设计与实现](#)。下面介绍各大组件的职责,见表 6-3-3:

表 6-3-3 在线评测组件职责

组件接口	组件默认实现	组件职责
ProblemService	ProblemServiceImpl	题目业务处理
WorkoutService	WorkoutServiceImpl	普通在线评测业务处理
CourseService	CourseServiceImpl	课程业务处理
HomeworkService	HomeworkServiceImpl	作业业务处理
SolveHWSERVICE	SolveHWSERVICEImpl	作业的业务处理
无	POJAjaxModule	ajax 模块服务组件

## 6.4 在线论坛扩展模块

### 6.4.1 领域模型分析

在线论坛扩展模块是为了提供用户交流用的,关于本模块的功能,详见 [5.2.3 在线论坛扩展模块](#)。

从模块功能上面进行分析,可以总结出以下几个领域模型:板块,主题,回

复，用户。图 6-4-1 是本模块的领域模型类图。

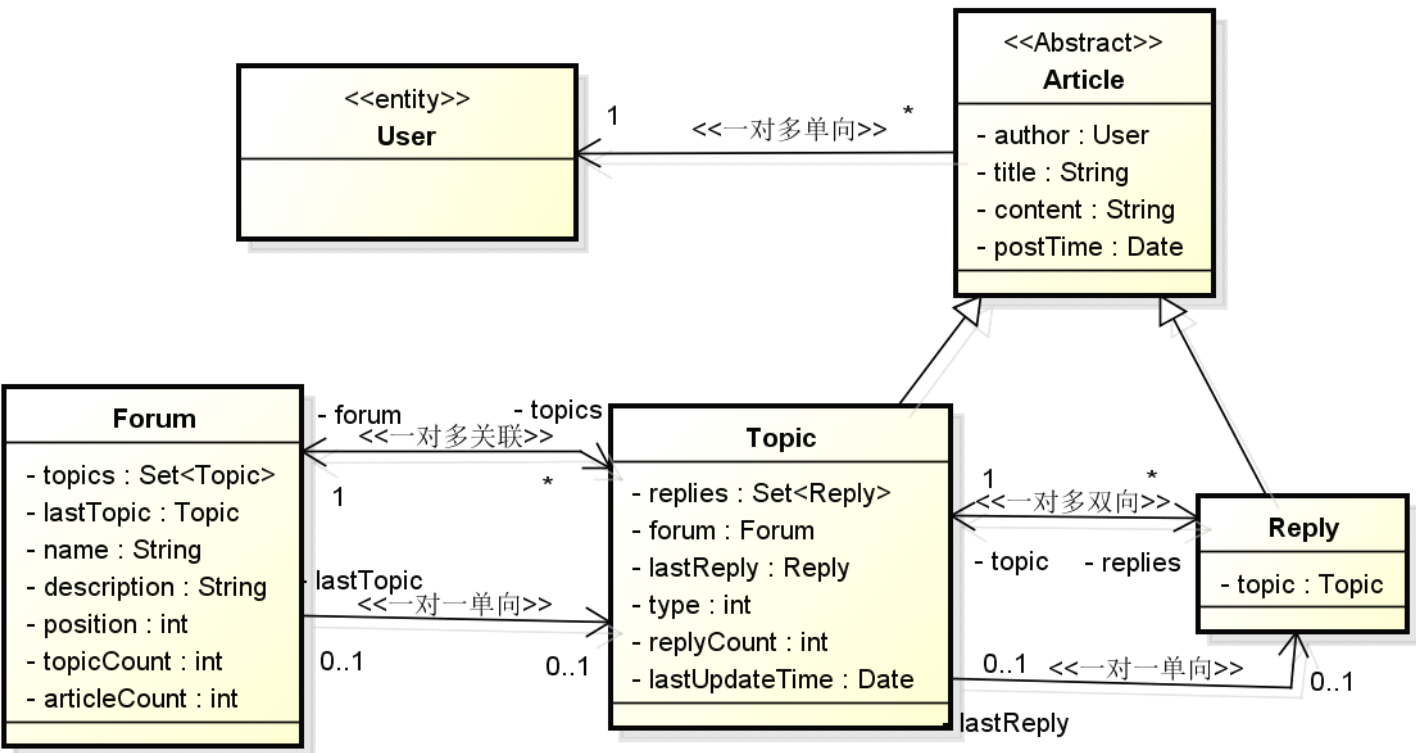


图 6-4-1 论坛模块领域模型

6.4.2 视图与控制层设计

本模块采用的还是 SSH，所以处理流程都是一样的，在本模块中包含以下两个核心的 Controller：ForumController 和 ForumManagerController，他们的职责如表 6-4-1：

表 6-4-1 Controller 职责描述

控制器	职责
ForumController	用户使用论坛控制器，主要从用户角度看
ForumManagerController	论坛管理控制器，从论坛管理角度看

根据 SpringMVC 处理流程，可以确定 Controller 和页面之间的调用关系，如



表 6-4-2:

表 6-4-2 论坛模块 Controller 映射

控制器	动作	RequestMapping	出口
ForumManagerController	板块列表	forumList.html	forumList.jsp
	编辑板块	editForumUI.html	editForumUI.jsp
	保存板块	saveForum.html	forumList.html
	更新板块	updateForum.html	forumList.html
	板块上移	moveUp.html	forumList.html
	板块下移	moveDown.html	forumList.html
	删除板块	deleteForum.html	forumList.html
ForumController	主题展示	forumShow.html	forumShow.jsp
	编辑主题	editTopicUI.html	editTopicUI.jsp
	保存主题	saveTopic.html	forumShow.html
	删除主题	deleteTopic.html	forumShow.html
	回复列表	topicShow.html	topicShow.jsp
	回复页面	replyTopicUI.html	replyTopicUI.jsp
	回复主题	replyTopic.html	topicShow.html
	移动主题	moveTopic.html	forumShow.html
	转普通帖	ordinaryTopic.html	forumShow.html
	转置顶帖	topTopic.html	forumShow.html
	转精华帖	creamTopic.html	forumShow.html

### 6.4.3 业务组件设计

和前面的一致，对于 Controller 来说，所有的组件都是接口类型，在本模块中包含下面的组件，如表 6-4-3:

表 6-4-3 论坛模块组件表

业务组件接口	组件默认实现	职责描述
ForumService	ForumServiceImpl	板块相关的业务处理
TopicService	TopicServiceImpl	主题相关业务处理
ReplyService	ReplyServiceImpl	回复相关业务处理

## 7 系统测试

系统的登录界面是用户使用本系统的唯一入口点，无论是具备什么角色的用户都通过该界面登录系统。



图 7-1 系统登录主界面

作为普通用户也能够进行注册，但是注册只能默认是普通用户，不能够选择，如果要转换自己的角色，需要向系统管理员沟通。



图 7-2 系统注册页面

## 7.1 基础应用模块测试

基础应用模块是系统的基础功能，首先使用超级管理员登录，登录后的界面如下：



图 7-1-1 Home 界面

鼠标移至系统管理→ 用户管理，见出现以下界面：

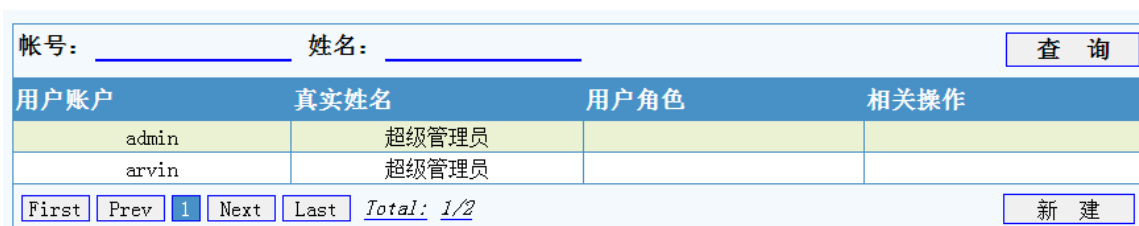


图 7-1-2 用户列表

超级管理员可以查看用户的列表信息，但是超级管理员不能删除自己和其他超级管理员，管理员具备非超级用户的增删改查。点击新建按钮出现创建用户的界面如下：

创建用户——填写用户基本信息

用户帐号：

cianfree

邮箱：

xiajiqiu1990@163.com

真实姓名：

张三

性别：

☒男

☐女

用户角色：

普通用户

学生

教师

提交

重置

返回

图 7-1-3 创建用户

接着点击提交，就会返回用户列表页面，并可以看到对应的操作选项，如下图：

帐号：姓名：

查 询

用户账户	真实姓名	用户角色	相关操作
admin	超级管理员		
arvin	超级管理员		
cianfree	张三	普通用户	<div>删除 修改 初始化密码</div>

First

Prev

1

Next

Last

Total: 1/3

新 建

图 7-1-4 添加用户后用户列表

点击初始化密码，将会给一个提示框，如下图：

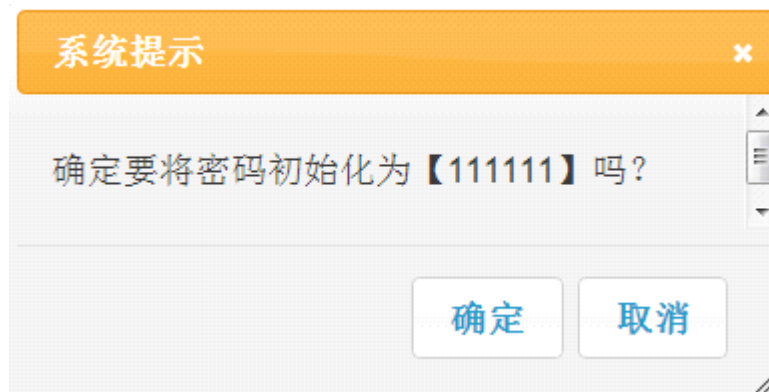


图 7-1-5 初始化密码提示

先来检查普通用户所具有的权限，将鼠标移至系统管理→角色管理，得到角色列表如下图：

角色名称	角色描述	相关操作
普通用户	用户在注册时候就是以普通用户的身份	修改 权限设置
学生	提供教学辅助的学生	删除 修改 权限设置
教师	提供教学辅助	删除 修改 权限设置

First Prev 1 Next Last Total: 1/3

新建

图 7-1-6 角色列表

可以看到，由于普通用户是系统固有的权限，不能删除，不过此时并没有配置相关的权限，单击权限设计进入普通用户的权限设置页面如下：

正在为[普通用户]角色配置权限

☐ 全选

☐ 论坛系统

☐ 系统管理

☐ 个人设置

☐ 在线测评

提交 返回

图 7-1-7 普通用户权限树

可以看到，当前普通用户没有任何权限，那么现在单击退出系统，使用张三这个用户登录系统，由于张三是普通用户，并且普通用户这个角色当前还没有配

置任何可用的权限，所以张三登录之后应该是没有什么可以操作的，如下图：



图 7-1-8 普通用户可操作的选项

可以看到，他没有在线评测和在线论坛的使用权限，但是有实用工具的权限，因为这个是本系统设定的逻辑，实用工具并没有纳入权限的控制之中。那么现在重新使用超级管理员登录，并授予普通用户以下权限：

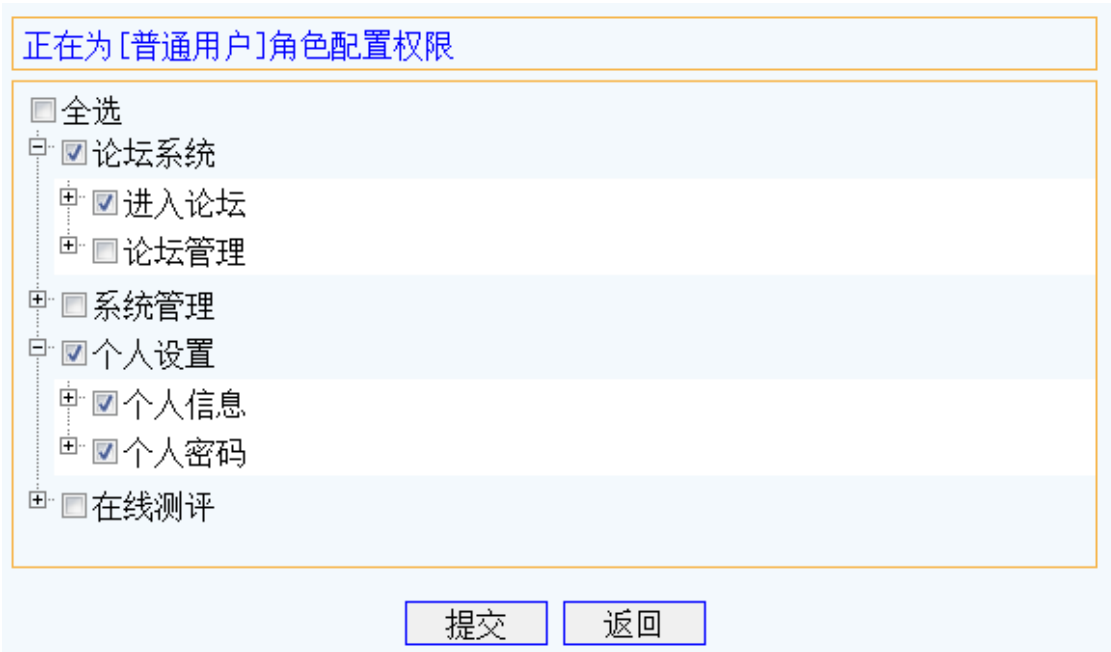


图 7-1-9 授权

本次演示授予普通用户使用论坛和个人信息设计的权限，点击提交并退出系统，再次使用张三这个用户登录系统，此时再来看：



图 7-1-10 新的权限

7.2 在线评测模块测试

在线评测模块是本系统中一个比较重要的扩展模块之一，先来测试普通用户情况下最简单的在线评测使用。使用超级用户登录，并部署 POJ 模块。登录系统之后，将鼠标滑至【在线评测】，在显示的下拉列表中选中【题目列表】，得到的结果如下：

编号:  关键字:  难度: ☐ 高级 ☐ 中级 ☒ 初级 排序规则: 按日期升序

编号	名称	创建日期	出题人	等级	操作
<div>新建</div>					

图 7-2-1 问题列表

由于当前并没有任何题目，所以列表为空，现在点击右下角的新建按钮，出现下面的画面。

题目名称

出题人

难度等级 ☒ 初级 ☐ 中级 ☐ 高级

题目描述 

向控制台输出HelloWorld

输入示例

输出示例 

HelloWorld

图 7-2-2 新建题目

上面的数据是要添加的数据，单击保存，就会跳转到题目列表页面，如下图，会显示刚添加的 HelloWorld 题目。

编号:	<input type="text"/>	关键字:	<input type="text"/>	难度:	<input checked="" type="radio"/> 高级	<input type="radio"/> 中级	<input type="radio"/> 初级	排序规则:	<input type="text" value="按日期升序"/>	<input type="button" value="查 询"/>
编号	名称	创建日期	出题人	等级	操作					
1	HelloWorld	2013-04-29 02:43:21.0	XJQ	初级	删除 修改 做题					
<div><input type="button" value="首页"/> <input type="button" value="上一页"/> <input checked="" type="button" value="1"/> <input type="button" value="下一页"/> <input type="button" value="末页"/> 总共: 1/1 <input type="button" value="新 建"/></div>										

图 7-2-3 显示添加的题目

在右边的操作中，我们可以看到可以进行删除，修改和做题操作，删除和修改这边就不再进行演示了，单击【做题】按钮，将出现以下界面。

<div><div>题目说明</div><div><div>题目名称</div><div>HelloWorld</div></div><div><div>题目描述</div><div>向控制台输出 HelloWorld</div></div><div><div>输入示例</div><div></div></div><div><div>输出示例</div><div>HelloWorld</div></div><div>这里描述题目的基本情况</div></div>	<div><div>编辑源代码</div><div>语言: <input checked="" type="radio"/> Java <input type="radio"/> C++</div><div><pre>public class Main {     public static void main(String[] args) {         System.out.println("HelloWorld");     } }</pre></div><div>这里是编辑代码的地方，系统会提供一个基本的代码结构给你单击其中的一种语言就会Ajax请求获取代码模版。</div><div><input type="button" value="提交代码"/> <input type="button" value="清空代码"/></div></div>
<div><div>运行情况</div><div>未运行任何代码.....</div><div>这里显示执行的结果</div></div>	

图 7-2-4 做题页面

左上角的面板是用来显示题目的基本信息，左下角的面板是用来显示代码的执行情况，右边的面板是用来编辑源代码的。本例是要在控制台中显示 HelloWorld，编辑好源代码之后，单击提交代码：





图 7-2-5 系统提示是否提交

系统会提交是否要进行提交，点击确定之后，就会出现下面的界面。



图 7-2-6 等待后台执行结果

这里其实是使用 Ajax 将代码发送到后台进行执行，然后把结果返回给客户端，实现无刷新的更新数据，最后把结果显示在左下角的面板中，如下图。



图 7-2-7 运行结果

当我们做完题目之后，就可以查看自己的做题结果了，把鼠标移动到在线测评，在下拉列表中选择我的成绩，就会显示我的做题列表。

编号	名称	解题时间	题目难度	答题状态	操作
1	HelloWorld	2013-04-29 02:49:42.0	初级	Accpeted	重做
首页 上一页 1 下一页 末页 总共: 1/1					

图 7-2-8 我的成绩列表

其中还有重做的操作选项，如果开始的时候做错了，你可以选择继续做。

### 7.3 在线论坛模块测试

在线论坛模块旨在提供一个站内用户的交流平台，提高程序学习的气氛和效率。为了便于演示所有的功能，我们使用超级管理员登录系统。登录系统之后，将鼠标移动到论坛系统，单击论坛管理，如下：



图 7-3-1 板块列表

系统刚上线的时候，还没有设置任何的板块，现在点击新建按钮，页面展示如下：

版块名称: JavaEE \*

版块说明: JavaEE相关技术讨论区

保存 返回

图 7-3-2 添加板块

输入相关信息，然后点击保存，系统会跳转到板块列表页面，再来看时就能看到刚添加的板块信息了，如下图：

版块名称	版块说明	相关操作
JavaEE	JavaEE相关技术讨论区	删除 修改 上移 下移

新建 首页 上一页 1 下一页 末页 总共: 1/1

图 7-3-3 显示添加板块

我们可以看到右边的操作中，有上移和下移的操作，这个是用来控制板块显示的位置的，先添加几个演示数据，再来看。

版块名称	版块说明	相关操作
JavaEE	JavaEE相关技术讨论区	删除 修改 上移 下移
C/C++	C/C++讨论专区	删除 修改 上移 下移
数据库	Oracle, MySQL, SQLServer等数据库技术专区	删除 修改 上移 下移

新建 首页 上一页 1 下一页 末页 总共: 1/3

图 7-3-4 板块列表

现在，我点击 C/C++操作栏中的[上移]，就会得到下面的结果，如图 7-3-5。

版块名称	版块说明	相关操作
C/C++	C/C++讨论专区	删除 修改 上移 下移
JavaEE	JavaEE相关技术讨论区	删除 修改 上移 下移
数据库	Oracle, MySQL, SQLServer等数据库技术专区	删除 修改 上移 下移
<input type="button" value="新建"/>		<input type="button" value="首页"/> <input type="button" value="上一页"/> <input type="button" value="1"/> <input type="button" value="下一页"/> <input type="button" value="末页"/> 总共: 1/3

图 7-3-5 板块上移操作

现在点击 JavaEE 板块对应操作栏中的下移操作，就会得到如下结果。

版块名称	版块说明	相关操作
C/C++	C/C++讨论专区	删除 修改 上移 下移
数据库	Oracle, MySQL, SQLServer等数据库技术专区	删除 修改 上移 <u>下移</u>
JavaEE	JavaEE相关技术讨论区	删除 修改 上移 下移
<input type="button" value="新建"/>		<input type="button" value="首页"/> <input type="button" value="上一页"/> <input type="button" value="1"/> <input type="button" value="下一页"/> <input type="button" value="末页"/> 总共: 1/3

图 7-3-6 JavaEE 板块下移

接下来测试分页的功能，现在继续添加板块，然后得到下面的结果，总共添加了 14 个板块，每页显示 10 条板块。

版块名称	版块说明	相关操作
C/C++	C/C++讨论专区	删除 修改 上移 下移
数据库	Oracle, MySQL, SQLServer等数据库技术专区	删除 修改 上移 下移
JavaEE	JavaEE相关技术讨论区	删除 修改 上移 下移
领域驱动设计		删除 修改 上移 下移
UML建模		删除 修改 上移 下移
设计模式	这些中文是可以替换的	删除 修改 上移 下移
HTML5		删除 修改 上移 下移
CSS		删除 修改 上移 下移
操作系统		删除 修改 上移 下移
Ubuntu Linux		删除 修改 上移 下移
<input type="button" value="新建"/>		<input type="button" value="首页"/> <input type="button" value="上一页"/> <input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="下一页"/> <input type="button" value="末页"/> 总共: 2/14

图 7-3-7 分页

点击按钮【2】或者是点击下一页，就会得到下面的结果。

版块名称	版块说明	相关操作
Android应用开发	现在显示的是第二页	删除 修改 上移 下移
JavaScript		删除 修改 上移 下移
SSH		删除 修改 上移 下移
IOS		删除 修改 上移 下移
<div>新建</div> <div>首页 上一页 1 2 下一页 末页 2/14</div>		

图 7-3-8 分页

以上测试主要演示了板块的管理功能，接下来要演示的是论坛的使用，把鼠标移到论坛系统上面，然后点击进入论坛，就会得到下面的结果。

版块	主题数	文章数	最后发表的主题
 <b>C/C++</b> C/C++讨论专区	0	0	主题: 作者: 时间:
 <b>数据库</b> Oracle, MySQL, SQLServer等 数据库技术专区	0	0	主题: 作者: 时间:
 <b>JavaEE</b> JavaEE相关技术讨论区	0	0	主题: 作者: 时间:
 <b>领域驱动设计</b>	0	0	主题: 作者: 时间:

图 7-3-9 板块及其相关信息列表

单击其中一个板块，如数据库，进入该板块的主题列表如下：

> 论坛 > 数据库		<a href="#">发新帖</a>	
主题	作者	回复数	最后回复
<a href="#">全部主题</a> 默认排序 (按最后更新时间排序, 但所有置顶帖都在前面)           降序 <a href="#">提交</a>			

图 7-3-10 板块的主题列表

由于该板块刚创建，因此没有主题，那么现在就可以单击【发新帖】按钮，













来发表新的主题，单击按钮，结果如下：


[论坛](#) > [数据库](#) >> [发表新主题](#)

标题

这个是新主题

内容

 这是新主题，大家有什么要发表的吗？欢迎来抢沙发

body p

提交

返回

图 7-3-11 发表新主题

填写好主题的内容之后，就可以单击提交按钮进行提交，提交之后会跳转到主题的列表页面，默认是普通主题帖，请看下图。

<a href="#">论坛</a> > <a href="#">数据库</a>		<a href="#">发新帖</a>	
主题	作者	回复数	最后回复
 这个是新主题	超级管理员 2013-04-29 02:06:52.0	0	2013-04-29 02:06:52.0

图 7-3-12 查看主题列表

单击该新主题，进入到该主题下的回复列表中，当然现在是没有任何回复内容的，因为是刚创建的，那么我们可以进行主题的回复。


论坛 > 数据库 >> 帖子阅读

发新帖

本帖主题：这个是新主题


[回复](#)
[移动到其他版块](#)
[精华](#)
[置顶](#)
[普通](#)

编辑



超级管理员

这个是新主题

 这是新主题，大家有什么要发表的吗？欢迎来抢沙发

[楼主]


2013-04-29 02:06:52.0


快速回复

标题

回复：这个是新主题

内容



这里可以进行快捷回复哦，

body p

提交

图 7-3-13 主题展示

单击提交按钮，就会刷新当前页面，并把刚才的回复内容显示出来。

论坛 > 数据库 >> 帖子阅读

发新帖

本帖主题：这个是新主题

[回复](#)
[移动到其他版块](#)
[精华](#)
[置顶](#)
[普通](#)

编辑



超级管理员

这个是新主题

 这是新主题，大家有什么要发表的吗？欢迎来抢沙发

[楼主]

2013-04-29 02:06:52.0



reply.author.name

回复：这个是新主题

这里可以进行快捷回复哦，

body p

[1楼]

2013-04-29 02:10:41.0

[首页](#)
[上一页](#)
[1](#)
[下一页](#)
[末页](#)
[总共： 1/1](#)

72

图 7-3-14 回复列表

相信你也注意到了右上角的一些按钮，这个按钮的功能也很简单，首先来看【回复】按钮，这个按钮就是回复当前主题，和快速回复一样，只不过是使用独立的回复页面，如下图。

> 论坛 > 数据库 >> 帖子回复	
帖子主题	这个是新主题
标题	<input type="text" value="回复：这个是新主题"/>
内容	<div><div>这个是独立的回复页面 😊</div><div>body p</div></div>
<div>提交 返回</div>	

图 7-3-15 独立的回复页面

再来看第二个按钮【移动到其它板块】，从左上角的导航中可以看到，当前主题实在【数据库】这个板块中的，现在我单击【移动到其它板块】按钮，将会出现下图：





图 7-3-16 移动板块页面

选中 JavaEE，然后单击提交按钮，就把该主题移动到 JavaEE 这个板块中去了，如下图：

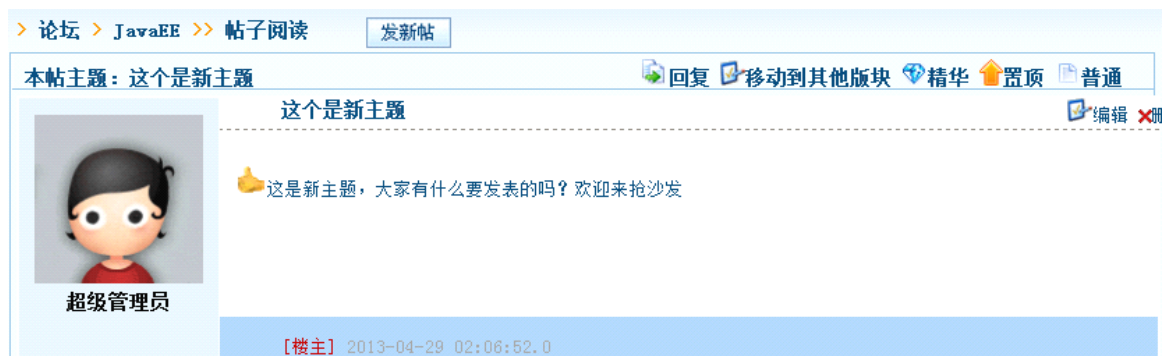


图 7-3-17 移动到其他板块的结果

可以看到左上角变了，变成了 JavaEE，也就是板块移动成功了。接下来是精华，置顶和普通三个按钮，这三个按钮的作用分别是把当前帖子变成精华帖显示，变成置顶帖显示和普通帖显示，默认情况下，帖子是普通帖，单击置顶按钮，可以看到下面的结果。


> 论坛 > JavaEE		发新帖	
主题	作者	回复数	最后回复
 这个是新主题	超级管理员 2013-04-29 02:06:52.0	2	超级管理员 2013-04-29 02:14:42.0
全部主题		默认排序 (按最后更新时间排序, 但所有置顶帖都在前面)	
		降序	提交
		首页	上一页 1 下一页 末页 总共: 1/1

图 7-3-18 置顶帖子

可以看到已经把该帖子变成了置顶帖子，另外两个按钮我就不再演示了，大同小异。我们在创建几个新的主题，看看置顶帖的效果，一般情况下，月先创建的帖子会放在最后面，也就是，默认情况下按照时间排序。

> 论坛 > JavaEE		发新帖	
主题	作者	回复数	最后回复
 这个是新主题	超级管理员 2013-04-29 02:06:52.0	2	超级管理员 2013-04-29 02:14:42.0
 这是第二个新帖	超级管理员 2013-04-29 02:22:39.0	0	2013-04-29 02:22:39.0
 这是第一个新帖	超级管理员 2013-04-29 02:22:20.0	0	2013-04-29 02:22:20.0

图 7-3-19 帖子的顺序

可以看到，默认情况下是按照时间的降序进行排列的，而且，置顶帖永远是在最上面的，即便是时间比较早创建也不受影响。点击进入第二个新帖，然后把它转成精华帖，精华帖同样会受到时间的排序。

> 论坛 > JavaEE		发新帖	
主题	作者	回复数	最后回复
 这个是新主题	超级管理员 2013-04-29 02:06:52.0	2	超级管理员 2013-04-29 02:14:42.0
 这是第二个新帖	超级管理员 2013-04-29 02:22:39.0	0	2013-04-29 02:22:39.0
 这是第一个新帖	超级管理员 2013-04-29 02:22:20.0	0	2013-04-29 02:22:20.0

图 7-3-20 转成精华帖

## 8 总结

经过一个多学期的艰苦奋斗，最终将毕业设计之可扩展的在线评测系统顺利完成，系统基本上已经达到了预期的效果。从最初学习和总结大学一直以来的知识，到问题领域的分析，技术的选型，架构的设计，代码的重构，架构的重构，模块开发，最后进行系统测试。这么一步一步走过来，经历了很多的心酸，但也获益匪浅。

本次毕业设计综合了很多知识，无论是开源框架，还是自定义的开发架构都很好运用到了本次系统的开发中来。是我感受最深的还是自己设计的一些组件和基础架构，如 Ajax 后端服务，这个是我参考了公司所开发的架构进行的一种简化。也许在别人眼中不值一提，但这是我经过努力思考的结果，在其中我成长了很多。

在实际的开发中，我总会不断的出现一些新的想法，不断对自己现有的架构进行调整和重构。一开始使用的是 Struts2 Spring Hibernate，后来换成了 SpringMVC Spring Hibernate 这是比较大的变动。

经过这次的毕业设计，我最兴奋的并不是能不能最终实现这个系统，我所在意的是，我能否把自己的想法体现到程序中来，否则我自己不会设计那么多的基础架构，网上完全有这种东西，我自己设计也是不断参考现有的，以便进行简化，开发最低限度适合系统的工具，同时降低对第三方框架和工具的依赖。

系统虽然能够实现基础期望功能，但仍然有缺陷。我自己设计的基础架构我也知道有很多的缺陷在里面，但是鉴于时间关系，并没有继续在自设计架构和工具上进一步优化设计，我想在以后的学习和工作中会进一步优化现在的工具，为我所用。

最后，总结一下本次毕业设计，几个月以来，虽然也会忙于工作，但总会抽出一些时间做毕业设计，遇到问题变先自己充分进行研究，想不明白的时候就直接去看开源框架的源代码，在不明白就去问百度，谷歌，如果还是不明白就问老师了，不过这种情况还真是少见。实习的时候，曾经我的一个技术主管对我说，我这种学习路线固然扎实，但是花的时间太多了，很多东西没必要刨根问底。虽然他说得有理，但是也许他不知道这就是我的乐趣吧。几个月来，终于看到了自己努力的成果。

## 参 考 文 献

- [1]计文柯. Spring 技术内幕 [M] 北京: 机械工业出版社, 2012.
- [2][美]David Flanagan. JavaScript: The Definitive Guide [M] O' REILLY, 2011.
- [3][美]Ross Harmes Dustin Diaz. Pro JavaScript Design Patern [M].TELECOM PRESS, 2009.
- [4]陶国荣. jQuery 权威指南 [M] 北京: 机械工业出版社, 2012.
- [5][美]Joshua Bloch. Effective Java [M] 北京: 机械工业出版社, 2011.
- [6] 南 阳 理 工 学 院 在 线 评 测 系 统 [EB/OL].  
<http://acm.nyist.net/JudgeOnline/problemset.php>.
- [7]宜宾大学在线评测系统[EB/LO]. <http://quanblog.com/oj>.
- [8] 基 于 RBAC 模 型 的 权 限 管 理 系 统 的 设 计 和 实 现 [EB/OL].  
<http://hi.baidu.com/injava/item/137586d196bfe6e7b2f77775>.
- [9]郭理, 秦怀斌, 梁斌. 基于 RBAC 的高校 Web 服务平台权限设计 [J]. 微计算机信息, 2011, (2).
- [10]jQuery Pagination Plugin [EB/LO].  
<http://archive.plugins.jquery.com/project/pagination>.
- [11]Spring Documentation[EB/LO]. <http://www.springsource.org/documentation>.
- [12][美]Erich Gamma Richard Helm Ralph Johnson[M]. 设计模式 北京: 机械工业出版社, 2011.
- [13] 高 性 能 WEB 应 用 开 发 指 南 [EB/LO].  
<http://developer.51cto.com/art/201104/257581.htm>.
- [14]jQuery 最佳实战[EB/LO]. <http://www.open-open.com/bbs/view/1318473226718>.
- [15]Java                      Object                      Memory                      Structure[EB/LO].  
<http://www.codeinstructions.com/2008/12/java-objects-memory-structure.html>.

# **The Research Of Extensible Program Online Judge System**

Xia Jiqu

(College of Information Science and Technology, Zhongkai University of Agriculture  
and Engineering, Guangzhou 510225, China)

## **Abstract**

Course of programming design is the mainly teaching contents of computer relative majors. Only take more practice and active communications can improve the programming ability. When in traditional learning process, people do the code judge work by their eyes. Meanwhile, it is not convenient to communicative with each other. The purpose of this system is to build and Program Online Judge which is B/S based and easy extensible. Several modules will be developed to promote the communication beyond learners. What's more, teaching assistant, easily judge processing and more effective programming learning will be provided.

This system is based on fluency and excellent open source framework. To building a WEB online judge system which is easy to extend. The implementation of system's framework and the technology mainly for SPRING, HIBERNATE, MYSQL and so on. Spring MVC is used to control the main business logic, while Hibernate implements the Object-Relation Mapping, and the MYSQL will store the system data. In the front page, JSP is ready to display. Meanwhile, web action will be controlled by the JQUERY. AJAX will be using to the transfer of data between front and back side without fresh, always the JSON formatted data.

This paper is ready to showing basic application's and framework's design of extensible system, the development of JQUERY plugin, AJAX service building and the Role-Based Access Control (RBAC) system. All of above is the basic applications which suitable for the core system. Finally, will extend three modules, respectively is the basic management module and program online judge module and BBS module.

**Key words:** Extensible; Spring MVC; Hibernate; Ajax; RBAC

## 致 谢

当写到这里的时候，心中突然有一种说不出的感觉，写到这里，我终于明白，我真的是要毕业了，回想起大学四年的种种经历，有太多的感慨，自己也将鼓起勇气迈向新的旅程。

在本次毕业设计中，首先应该感谢的是我的导师顾春琴老师，在我为毕业论文题目举棋不定的时候，她给予了我很多的建议，最终我也采纳了她的建议，在老师的指导下最终完成了毕业论文的写作。

其次，非常感谢公司的同事，在公司的时候，通过学习他们的总体框架，我感悟很多，并最终将他们的一些技术灵活运用到自己的项目中来。

同时，非常感谢我的亲朋好友，特别是我的父母和姐姐，他们经常打电话询问我的情况，却从不问我学习怎么样，而是关心我的生活，在后面默默得给予我支持，我才能毫无顾虑的向前进。

我想最要感谢的是我自己了，感谢自己在毕业设计开题到现在一直孜孜不倦的努力着，回想起这个过程，自己一次又一次的进行项目的重构，代码的重构和优化等等，现在想起来还是历历在目啊。

论文即将完成之际，非常激动，从论文开题到顺利完成的过程中，我感谢我的师长，亲人，同学和我的同事，他们在百忙之中给予了我慷慨的帮助。请接受我最诚挚的谢意。

最后，再次对关心、帮助我的老师、同学、同事和我的亲人表示衷心地感谢！