|  |  |
| --- | --- |
| 学院标志(改) 副本 | 惠州学院  HUIZHOU UNIVERSITY |

毕 业 论 文（设 计）

**中文题目：­­­­­­­­ 课室自动点名系统的设计与开发\_\_\_**

**英文题目：** **Automatic attendance system**

**姓 名\_\_\_\_方争由\_\_\_\_\_\_\_\_**

**学 号\_\_\_\_081403104\_\_\_\_\_**

**专业班级\_\_\_\_软件工程1班\_\_**

**指导教师\_\_\_\_彭刚\_\_\_\_\_\_\_\_\_\_**

**提交日期\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

目录

[摘 要 4](#_Toc325572480)

[前 言 6](#_Toc325572481)

[第一章 绪论 7](#_Toc325572482)

[1.1 研究背景 7](#_Toc325572483)

[1.2 设计目标 7](#_Toc325572484)

[1.3 本文结构 7](#_Toc325572485)

[第二章 团队协作开发方式 9](#_Toc325572486)

[2.1 团队协作概述 9](#_Toc325572487)

[2.1.1 团队协作的重要性 9](#_Toc325572488)

[2.1.2 团队协作的主要思想 9](#_Toc325572489)

[2.2 我们的协作方式 10](#_Toc325572490)

[2.2.1 开会讨论 10](#_Toc325572491)

[2.2.2 功能列表文档记录 13](#_Toc325572492)

[2.2.3 每日工作汇报 15](#_Toc325572493)

[2.3 团队开发流程 17](#_Toc325572494)

[第三章 系统开发环境与技术 23](#_Toc325572495)

[3.1 系统开发环境 23](#_Toc325572496)

[3.1.1 MyEclipse插件介绍 23](#_Toc325572497)

[3.1.2 Tomcat服务器介绍 23](#_Toc325572498)

[3.2 系统开发技术 24](#_Toc325572499)

[3.2.1 JSP与Servlet技术 24](#_Toc325572500)

[3.2.2 JavaScript简介 27](#_Toc325572501)

[3.2.3 MVC模式 28](#_Toc325572502)

[3.2.4 Struts框架 29](#_Toc325572503)

[3.2.5 Spring框架 31](#_Toc325572504)

[3.2.6 Hibernate框架 33](#_Toc325572505)

[第四章 系统需求分析与前台设计 35](#_Toc325572506)

[4.1 需求分析 35](#_Toc325572507)

[4.1.1 系统超级管理员操作平台设计概述 35](#_Toc325572508)

[4.1.2 系统用例图 36](#_Toc325572509)

[4.2 系统设计 39](#_Toc325572510)

[4.2.1 系统层次划分 39](#_Toc325572511)

[4.2.2 数据库设计 40](#_Toc325572512)

[4.2.3 点名系统时序图 44](#_Toc325572513)

[第五章 系统详细设计与功能实现 47](#_Toc325572514)

[5.1 系统项目的文件夹结构 47](#_Toc325572515)

[5.2 系统运行过程细化分析 48](#_Toc325572516)

[5.2.1 项目中的JSP 48](#_Toc325572517)

[5.2.2项目中的DAO 49](#_Toc325572518)

[5.2.3项目中的JDBC 50](#_Toc325572519)

[5.3 视频管理器模块的具体实现 53](#_Toc325572520)

[5.3.1 拍照照片查询功能的实现 53](#_Toc325572521)

[5.3.2 照片删除功能的实现 55](#_Toc325572522)

[5.3.3 拍照时间设置功能的实现 57](#_Toc325572523)

[5.4 人员权限控制模块的具体实现 59](#_Toc325572524)

[5.4.1 角色操作功能设置的功能实现 59](#_Toc325572525)

[5.4.2 人员角色设置功能的实现 60](#_Toc325572526)

[5.5 修改密码模块的具体实现 62](#_Toc325572527)

[第六章 总结与展望 64](#_Toc325572528)

[6.1 课题总结 64](#_Toc325572529)

[6.2 进一步开发的展望 64](#_Toc325572530)

[参考文献 65](#_Toc325572531)

[致 谢 66](#_Toc325572532)

[附录A 外文原文 67](#_Toc325572533)

[附录B 外文译文 74](#_Toc325572534)

# 摘 要

课室自动点名系统(已申请国家专利)是对目前学校的考勤实际情况而进行设计、开发的，目的是为了用一种智能的自动化的方式实现课室的考勤任务。利用JSP技术和SSH框架以及相应的数据库访问技术实现了基于Web的系统。该框架可以减少模块之间的耦合性，让开发人员减轻重新建立解决复杂问题方案的负担，并且可以被扩展以进行内部的定制化。通过使用JSP技术建设动态网站，充分发挥了Java语言所独有的易用性、跨平台性和安全性，从而构建了一个运行高效、安全可靠、适用性广的管理系统，实现了课室考勤信息的管理与交互，满足自动点名的需求。

论文首先简要介绍了课室自动点名系统的一些研究与应用背景，其次介绍了该系统所采用的开发工具、平台以及开发环境。在此基础上，论文详尽描述了课室自动点名系统情况。

关键词：**JSP****，SSH框架，课室自动点名系统**

作者：方争由

指导老师：彭 刚

**Abstract**

Automatic attendance system (has applied for the national patent applications) is about to solve school's attendance satuation,which is developed with an intelligent, automated way to achieve the task of classroom attendance. The JavaEE technology and Orcale of database access technology are used to achieve the system which is based on B / S structure. The system uses the SSH framework to reduce re-establish of burden and try to solve complex problems, the program enhanced system scalability while easing of customization of internal coupling between modules..Develop by the unique of JSP technology to build dynamic websites can give full play to the ease of use by the Java language, cross-platform and security to build a run-effective, safe, reliable, and wide applicability management system, as well as to meet the classroom attendance information's management and interaction and the demand for auto-named. The thesis first introduces the research and application of background classrooms automatic naming system, followed by the introduction of the system development tools, platform and development environment. On this basis, the thesis has description of the classroom automatically attendance system situation with circumstance.

Keywords： **JSP，attendance system，Orcale**

Written by Zhengyou Fang

Supervised by Larry

# 前 言

随着科技的快速发展，我们身边的一切改变得如此之快，把固定通信变改为移动通信，把机械按钮改变为触摸按钮，IBM甚至提出在5年内生产出用人的意念控制的设备, 人们在追求更为便捷的生活。大学的课堂里，老师往往需要花费许多时间去点名，处理请假、旷课、迟到、早退，最终记录学生的出勤率，给予评价，事情是如此的繁琐。那么我们是否能研发一种系统，能够完全替代老师所要做的学生出勤率统计工作，让老师的工作更为轻松，便捷呢？这就是我们的课室自动点名系统。

本文的结构如下：

第一章是绪论部分，叙述了本系统的开发背景、开发目标、开发意义以及国内外的一些比较。

第二章介绍了团队合作开发的重要性和具体展示我们的团队合作方法。

第三章介绍开发的环境以及用到的相关技术的介绍。

第四章是系统需求分析与前台设计，阐述了在系统开发前的分析工作、系统的整体设计以及数据库设计。

第五章是系统的详细设计与功能实现，里面具体到对代码的详细解析。

第六章总结和展望，对整个项目的过程和感悟进行总结，并对项目未来的开发做出展望。

# 第一章 绪论

## 1.1 研究背景

当今时代信息技术飞速发展，人们已经处于数字化的生存空间之中。科技已经从根本上改变了人们的生活方式和工作方式，也改变了课堂上的各种传统。本系统通过对课室在有人和没人的时候获取图像，经过算法进行对比得到出勤信息，再通过管理系统进行管理。

课室自动点名系统是为是实现课室点名、学生出勤信息统计处理的系统，它可以智能的对课室的出勤情况进行分析，并统计好学生的请假、旷课、迟到、早退情况，并记录在学生的出勤信息库里，老师可根据信息库显示学生出勤状况，给予合理的评价。

我们生活在一个科技日新月异的时代，传统而繁琐的事情需要利用科技去改变，追求更为便捷、轻松的生活。

## 1.2 设计目标

开发设计本系统的任务是完成智能考勤功能，通过该系统能够自动获取出勤信息，并用户能对考勤相关的信息（如课程、人员等）进行管理。

开发系统的目标是实现点名自动化及对考勤信息进行各种管理的功能，满足的要求具体可包括如下几个方面：

1. 完成智能考勤功能。
2. 对考勤信息进行统计及考勤相关信息的管理。
3. 系统运行安全稳定。
4. 要求编写完善的文档，方便后期程序设计工作。

## 1.3 本文结构

本文的章节安排如下：

第一章绪论部分，说明本文的研究背景和设计目标。为接下来的系统后台的分析与设计做基础准备。

第二章团队合作开发，从思想上阐述其重要性，以及用实例介绍我们的开发小组的团队合作的具体方法。

第三章背景技术，简单介绍系统实现所用到的平台与技术。如MyEclipse、Tomcat服务器、JSP与Servlet原理、JavaScript、SSH（Struts-Spring-Hibernate）框架等。

第四章系统分析与设计，对系统后台进行需求分析，根据需求分析设计相应的功能与表结构。

第五章系统实现，给出系统后台的具体实现，包括实现效果和核心代码说明。

第六章总结与展望，对本文所做的工作进行总结，并针对目前工作中的不足对未来工作的展望。

# 第二章 团队协作开发方式

## 2.1 团队协作概述

### 2.1.1 团队协作的重要性

软件在不断的演进中变得越来越复杂，智力高度集中的软件开发也需要从个人英雄主义进入紧密配合的团队协作时代。IBM高级技术经理王守慧从事软件开发已经超过10年，软件开发的生命周期管理之痛是她这十年来一直思索着的问题。王守慧认为：“在传统的软件开发流程中，项目组中的每个成员在理解自己所完成的部分在整个软件中所起的作用时，是非常抽象的，因为新产品在整体交付之前，无论是对客户还是对它的建造者来说，几乎是不可见的”

软件开发应该是一种高度协作的过程，每个人不但要完成自己这部分的代码，更重要的是需要和项目中的各个模块完美的结合在一起。就像是一支乐队在集体演奏，每一个音乐家不但要集中精力演奏自己的部分，同时还要与乐队保持合拍与同步，否则表演就会失败。过去，每个人都在寻找适合自己的工具，导入导出着各种信息数据，用来提高软件开发过程中的协作、效率和透明度，现在，我们需要一个平台，它能够将整个软件开发生命周期中的各个环节都无缝的管理起来。

### 2.1.2 团队协作的主要思想

对于一个团队协作是否能成功，最基本的取决于能否有效的沟通。沟通是人与人之间交流的方式。主动沟通说到底是对沟通的一种态度。在项目中，我们极力提倡主动沟通，尤其是当已经明确了必须要去沟通的时候。当沟通是项目经理面对用户或上级、团队成员面对项目经理时，主动沟通不仅能建立紧密的联系，更能表明你对项目的重视和参与，会使沟通的另一方满意度大大提高，对整个项目非常有利  
 项目中的沟通形式是多种多样的，通常分为书面和口头两种形式。书面沟通一般在以下情况使用：项目团队中使用的内部备忘录，或者对客户和非公司成员使用报告的方式，如正式的项目报告、年报、非正式的个人记录、报事帖。书面沟通大多用来进行通知、确认和要求等活动，一般在描述清楚事情的前提下尽可能简洁，以免增加负担而流于形式。口头沟通包括会议、评审、私人接触、自由讨论等。这一方式简单有效，更容易被大多数人接受，但是不象书面形式那样“白纸黑字”留下记录，因此不适用于类似确认这样的沟通。口头沟通过程中应该坦白、明确，避免由于文化背景、民族差异、用词表达等因素造成理解上的差异，这是特别需要注意的。沟通的双方一定不能带有想当然或含糊的心态，不理解的内容一定要表示出来，以求对方的进一步解释，直到达成共识。除了这两种方式，还有一种作为补充的方式。沟通看似简单，实际很复杂。这种复杂性表现在很多方面，比如说，当沟通的人数增加时，沟通渠道急剧增加，给相互沟通带来困难。典型的问题是“过滤”，也就是信息丢失。下面将介绍我们在开发项目中所采用的一些重要的方法。

## 2.2 我们的协作方式

### 2.2.1 开会讨论

在老师的带领下，我们采用“头脑风暴”的方法进行项目讨论。我们会找一个固定时间在一起讨论项目开发的事，讨论的主要内容为在开发中遇到的困难与对策。同时在场的时候有专人记录当场的讨论重点内容，开会结束后，再回去整理成电子档的会议记录文档。下面介绍有关”头脑风暴“的相关方法。

“头脑风暴”法是指让讨论小组的每个成员自由的提出各种想法，而不管这些想法看起来多么幼稚和离奇。在实际工作中，又几乎80%的创新任务可以使用头脑风暴法。我们开会讨论的结果所图2-1所示。

我们这次开发项目主要使用了到头脑风暴里的以下几种方法：

1. 思维线索。为了鼓励认真听取别人的想法，你和其他小组中的成员应该注意其思维的线索——主题，即想法之间的紧密联系而不是想法本身。然后，通过这些线索来激发其他方面的新主意。

2.点名进行。小组的每个成员依次被点名说出自己的主意，这样使大家都有机会发表自己的意见。鼓励性格比较被动的成员积极参与，同时限制一般情况下侃侃而谈的成员，使他们不至于主宰整个会议。

****

图2-1

打开其中的两份文档，我们会看到会议的记录内容包括开会时间，开会地点，开会内容主题，下次开会时间的规定等等内容。详情参见下表2-2 与表2-3

计算机系点名系统第五次会议记录

|  |  |
| --- | --- |
| 活动时间 | 2011-5-5 (星期四) 20：00-21：00 |
| 活动地点 | 计算机系 5-407 实验室 |
| 软件公司 |  |
| 记录人 | 方争由 |

|  |  |
| --- | --- |
| 项目 | 讨论内容 |

|  |  |
| --- | --- |
| 1、权限设置问题 | (1) 超级管理员给其他用户权限扩大权限时，只需要将该用户升级，比如说，将老师权限升级到普通管理员，这样老师就有普通管理员权限了 |
| 2、确定功能ID字母编号 | (2) 功能ID字母编号确定，请看新文档 |
| 3、工具使用问题 | (3) 工具使用问题，留到大家一起来讨论 |
| 4、系统整体架构问题 | (4) 系统整体架构问题，还得继续讨论 |
|  | 下次会议：  时间：两周后的星期五，即：5-18号晚上8点。  地点：老地方，5-427  内容：  （1）完善功能ID表。  （2）设计好数据库表及表项。 |

表2-2

计算机系点名系统第八次会议记录

|  |  |
| --- | --- |
| 会议时间 | 2011-6-10(星期五) 20：00-22：00 |
| 会议地点 | 计算机系 5-427 实验室 |
| 软件公司 |  |
| 记录人 | 赵榜样 |

|  |  |
| --- | --- |
| 项目 | 讨论内容 |

|  |  |
| --- | --- |
| 1、数据库表项讨论 | * 1. 初步确定：数据库具有以下逻辑表：座位表、课程表、相机、   （管理员、教师）、学生、科目表   * 1. 意见分岐：1、是否将管理员和教师放在同一个表   2、座位号的安排方式  1．3） 表中的数据项还未明确，小组成员间交换了各自的意见  拟定：各成员回去整合各成员建议，完善自己方案，待下次会议再行决定。 |
| 2、开发环境讨论 | 小组各成员交换意见后，初步商定开发环境如下：   1. 开发软件：Eclipse + Oracle 2. 开发语言：JSP + cpp |
| 3、老师答疑 | 1、问：视频器具有的功能？  Dr彭：尽量利用市场已有的设备，但也可以自行开发  2、问：数据库是否由学校方面提供？  Dr彭：不是。和已有系统整合存在一定难度，我们这个系统先做成是自己独立的一个数据库系统。 |
| 下次会议：  时间：两周后的星期五，即：6-24号晚上8点。  地点：老地方，5-427  内容：对数据表项做进一步的讨论确认，完善数据库设计 | |

### 2.2.2 功能列表文档记录

从开会讨论得出接下来主要做的工作 ，然后依据得出的解决方案进行工作划分功能列表。在第一次讨论得出后做出功能列表文档，在做的过程中遇到一些困难无法解决的时候，把困难记下来，待下一次开会讨论之时，提出来跟团队一起讨论分析，最后得出新一个开会记录，再去修改原有的文档。由此不断迭代升级。下面图2-4就是我们不断讨论得出来的功能列表文档记录。打开较新的一次文档记录，可看到超级管理员操作平台的详细内容记载如表2-5所示。

****

图2-4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 平台 | 模块 | 功能 | 类型 | 编号 | 设计者 |
| 超级管理员操作平台 | 人员权限控制 | 管理员操作设置 | 界面 | HC001 | 方争由 |
|  |  | 学生操作设置 | 界面 | HC002 | 方争由 |
|  |  | 家长操作设置 | 界面 | HC003 | 方争由 |
|  |  | 老师操作设置 | 界面 | HC004 | 方争由 |
|  |  | 人员角色设置 | 界面 | HC005 | 方争由 |
|  | 视频处理器管理 | 拍照照片查询 | 界面 | HC006 | 方争由 |
|  |  | 照片删除 | 界面 | HC007 | 方争由 |
|  |  | 拍照时间设置 | 界面 | HC008 | 方争由 |
|  |  | 拍照频率设置 | 界面 | HC009 | 方争由 |
|  |  |  |  |  |  |
|  | 退出系统 |  | 界面 | HC011 | 赵榜样 |
|  | 重新登录 |  | 界面 | HC012 | 赵榜样 |
|  | 修改密码 |  | 界面 | HC013 | 赵榜样 |
|  | 查询基本信息（12345） | 超级管理员基本信息查询 | 界面 | HC014 | 赵榜样 |
|  |  | 普通管理员基本信息查询 | 界面 | HC015 | 赵榜样 |
|  |  | 老师基本信息查询 | 界面 | HC016 | 赵榜样 |
|  |  | 学生基本信息查询 | 界面 | HC017 | 赵榜样 |
|  |  | 家长基本信息查询 | 界面 | HC018 | 赵榜样 |
|  |  | 查询个人基本信息 | 界面 | HC019 | 赵榜样 |
|  |  | 修改个人基本信息 | 界面 | HC020 | 赵榜样 |
|  | 数据备份与恢复 | 自动备份时间修改 | 界面 | HC021 | 赵榜样 |
|  |  | 数据备份 | 界面 | HC022 | 赵榜样 |
|  |  | 数据恢复 | 界面 | HC023 | 赵榜样 |
|  |  | 数据日志 | 界面 | HC024 | 赵榜样 |
|  | 学生出勤信息处理 | 按条件查询学生出勤信息 | 界面 | HC025 | 江伟志 |
|  |  | 修改学生出勤信息 | 界面 | HC026 | 江伟志 |
|  |  | 下载学生出勤表单 | 界面 | HC027 | 江伟志 |
|  |  | 统计出勤率 | 界面 | HC028 | 江伟志 |
|  | 课表信息处理 | 录入学生课表 | 界面 | HC029 | 江伟志 |
|  |  | 下载学生课表 | 界面 | HC030 | 江伟志 |
|  |  | 查询学生课表 | 界面 | HC031 | 江伟志 |
|  | 课室更改 | 学生课室更改 | 界面 | HC032 | 江伟志 |
|  | 更改学生座位 | 学生座位更改 | 界面 | HC033 | 江伟志 |
|  | 公告处理 | 发布公告 | 界面 | HC034 | 江伟志 |
|  |  | 查看公告 | 界面 | HC035 | 江伟志 |
|  | 节假日修改 |  | 界面 | HC036 | 江伟志 |

表2-5

### 2.2.3 每日工作汇报

在三个人的协作工作当中，也会遇到很多合作和技术上的问题。因此我们小组会在每天工作结束以后，总结一天当中的不足与收获，把自己的遇到的问题也一并通过电子邮箱发送给团队其它成员与老师审阅，方便被组中其它成员看到其工作内容，另一方面其它成员也碰到这种情况，还可以帮助他解决那个问题。

另外一点，在编写与阅读文档的时候，我们尽量使用英语模板去编写文档，而非中文。这样子有利于我们提高的英文阅读与写作能力。下面简单介绍一下每日工作汇报采用的模板。并给出几个实际当中交流发送的邮件如图2-6 图2-7所示。

Project Name: Attendance System（开发项目名称）

Team Member Name:XXX（记录成员）

Date:2011.07.17（记录时间）

Work completed today: （今天完成了什么内容）

What’s going well and why:（今天在哪方面做的好，为什么）

1.N/A.

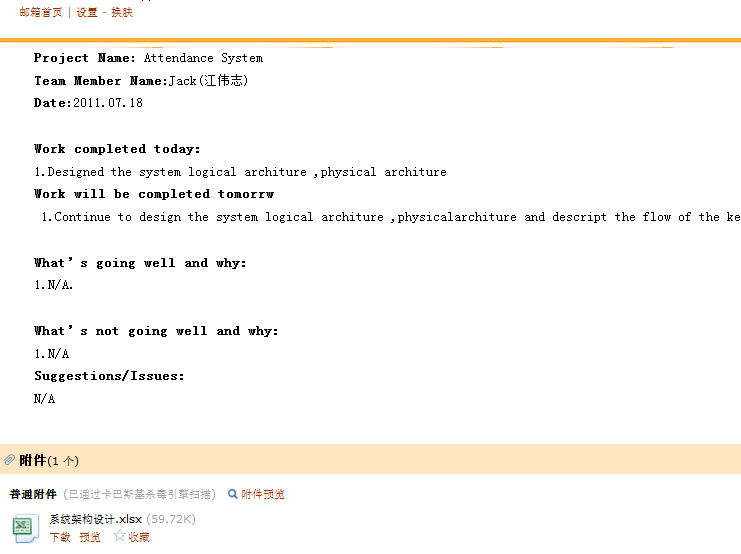
What’s not going well and why:（今天在哪方面遇到困难，为什么）

1.N/A

Suggestions/Issues:（有什么建议/明天的工作内容）

1.N/A

下面给出几个在开发过程中的几个实例例子如下所示：

****

**图 2-6**

****

**图 2-7**

## 2.3 团队开发流程

设计过程中，我们采用了彭刚老师从日本带回来的文档开发方式。我们采用老师的功能详细设计书模板（EXCEL工具）。把每个单独的功能划分出来，每个功能做一个详细的文档说明。文档的内容包括主要包括：

1. 封面

介绍了该份文档的功能名，与一些简要的功能组成说明。如下图2-1所示，其中包括了功能编号，功能名，与由几个画面组成等简要信息。

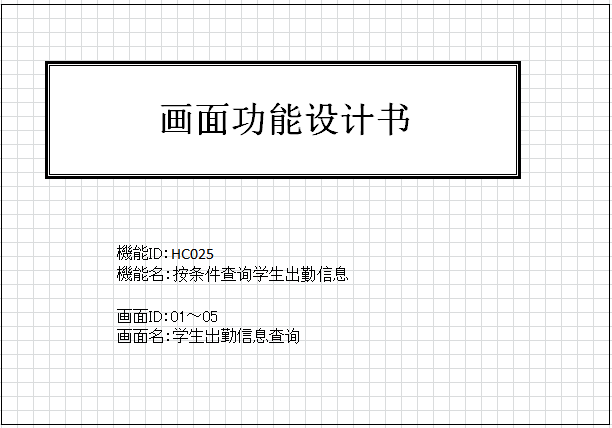


图2-1

1. 变更记录

变更记录主要用来记录对文档的修改说明。一份文档的需求或功能可能随时改变，所以在对文档修改时，有必要记录第N次修改的说明。如图2-2所示，该页面中主要包括更改日期，更改内容与变更人等信息。



图 2-2

1. 功能设计

该页面主要是对当前功能的阐述。里面主要包括功能概要，还有前提条件及功能明细等内容。就查询考勤结果这一功能而言，如图2-3所示。

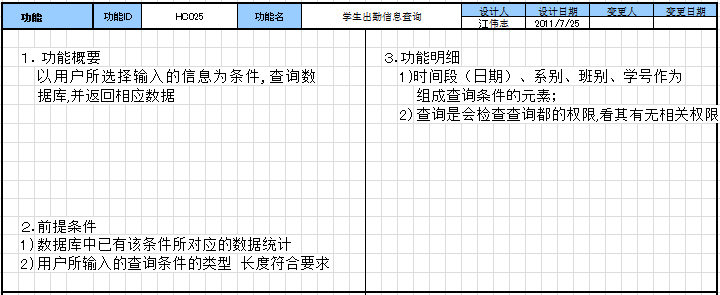


图2-3

1. 画面迁移图

该页面主要是编写当前功能的画面逻辑跳转。针对人员角色设置这一功能，要经由后台首页------>人条件输入---🡪结果显示等几个画面组成该功能。如下图2-4所示。

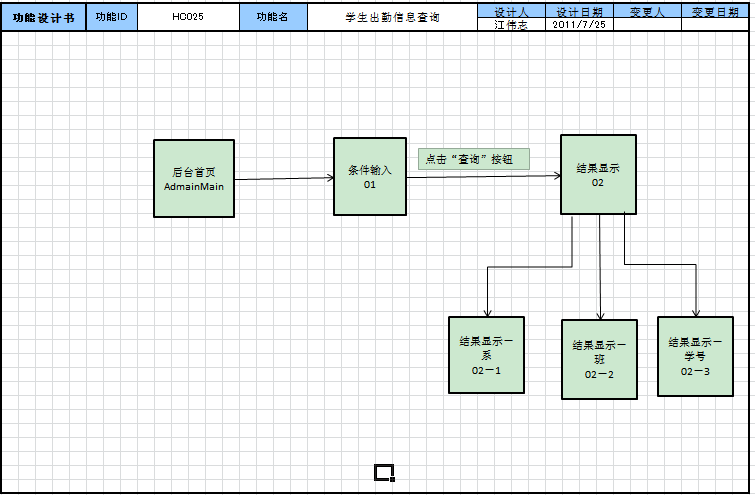


图2-4

1. 画面构成

主要是针对该功能文档中的画面具体的形象表达与具体说明。包括画面构成，与画面的具体细节说明。如图2-5与图2-6里面有具体某个跳转页面的画面图，在右侧有对该画面图的具体说明。

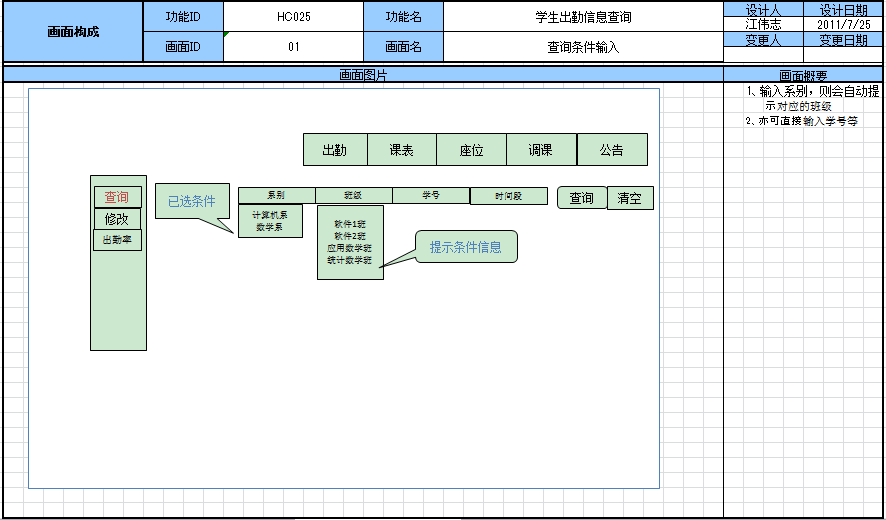


图2-5

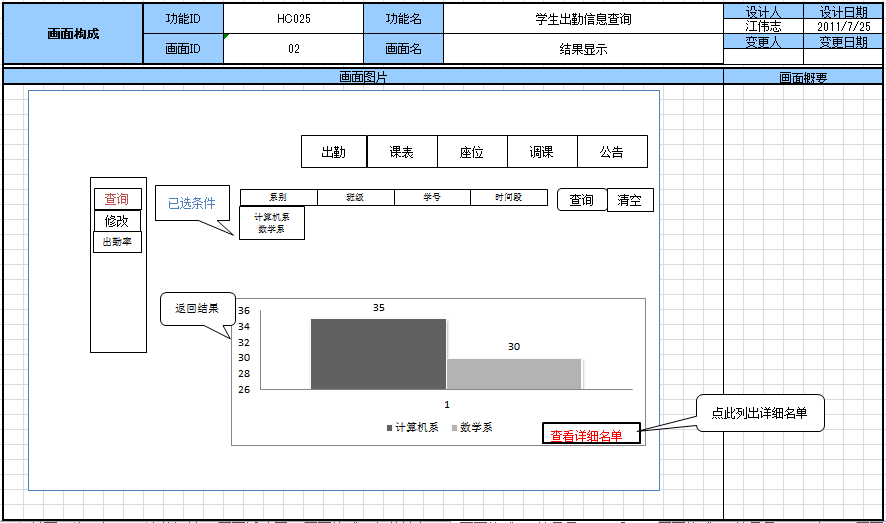


图2-6

1. DB操作

该页面主要是编写当前功能所涉及到的数据库操作说明，主要包括数据库的SQL操作说明。例如查询出勤信息功能。此时会用到

SELECT语句查询相对应的数据库部分。如图2-7所示。

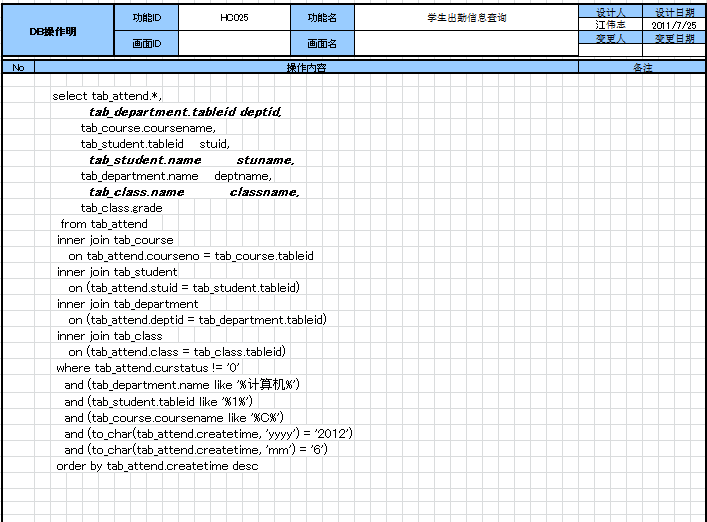


图2-7

# 第三章 系统开发环境与技术

## 3.1 系统开发环境

课室自动点名系统以Eclipse为开发平台，使用MyEclipse作为Eclipse的插件，并由MyEclipse 自带的服务器充当Web服务器来开发实现的。

### 3.1.1 MyEclipse插件介绍

MyEclipse是一个商业软件（商业插件）、基于Java和Eclipse的可扩展开发平台（比如MS Visual Studio开发平台中的VC、VB等。Borland公司的Delphi、Borland、C++、Builder）。就其本身而言，它只是一个框架和一组服务，用于通过插件组件构建开发环境。

虽然大多数用户很乐于将MyEclipse当做Java IDE（Integrated Development Environment）来使用，但MyEclipse的目标并不仅限于此。MyEclipse还包括插件开发环境（Plug-in Development Environment-PDE），当然，这个功能的实现主要是Eclipse，这个组件主要针对希望扩展Eclipse核心的软件开发人员，因为它允许他们构建与Eclipse环境无缝集成的工具。由于Eclipse中的每样东西都是插件，对于给Eclipse提供插件，以及给用户提供一致和统一的集成开发环境而言，所有工具开发人员都具有同等的发挥场所。

这种平等和一致性并不仅限于Java开发工具。尽管Eclipse是使用Java语言开发的，但它的用途并不限于Java语言。例如：支持诸如C/C++、COBOL和Eiffel及Perl等编程语言的插件。

MyEclipse是一款功能强大的J2EE集成开发环境，支持代码编写、配置、测试及除错。程序功能包括：HTML智能编辑器、Struts、JSF、CSS、JavaScript、SQL、Hibernate、Spring等，带有自动完成与语法高亮显示功能的J2EE编辑器。在Eclipse中安装这个插件可以大大提高开发J2EE的效率。

### 3.1.2 Tomcat服务器介绍

Tomcat是一个免费开源的Servlet容器，它是Apache 软件基金会（Apache Software Foundation）Jakarta 项目中的一个核心项目，由Apache、Sun 和其他一些公司及个人共同开发而成。由于有了Sun 的参与和支持，最新的Servlet 和JSP 规范总是能在Tomcat 中得到体现，Tomcat 7 支持最新的Servlet 3.0 和JSP 2.2 规范。Tomcat不仅是一个Servlet容器，它也具有传统的Web服务器的功能，如处理HTML页面，简单的域名管理，配置JNDI等。因为Tomcat 技术先进、性能稳定，而且免费，因而深受Java 爱好者的喜爱并得到了部分软件开发商的认可，成为目前比较流行的Web 应用服务器。

Tomcat 很受广大程序员的喜欢，因为它运行时占用的系统资源小，扩展性好，支持负载平衡与邮件服务等开发应用系统常用的功能；而且它还在不断的改进和完善中，任何一个感兴趣的程序员都可以更改它或在其中加入新的功能。

Tomcat 是一个小型的轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选。对于一个初学者来说，可以这样认为，当在一台机器上配置好Apache 服务器，可利用它响应对HTML 页面的访问请求。实际上Tomcat 部分是Apache 服务器的扩展，但它是独立运行的，所以当你运行Tomcat 时，它实际上作为一个与Apache 独立的进程单独运行的。

这里的诀窍是，当配置正确时，Apache 为HTML页面服务，而Tomcat 实际上运行JSP 页面和Servlet。另外，Tomcat和IIS、Apache等Web服务器一样，具有处理HTML页面的功能，另外它还是一个Servlet和 JSP容器，独立的Servlet容器是Tomcat的默认模式。不过，Tomcat处理静态HTML的能力不如Apache服务器。

## 3.2 系统开发技术

### 3.2.1 JSP与Servlet技术

JSP技术是建立在Servlet技术之上的。先有Servlet技术，后推出的JSP技术。JSP最终要编译为Servlet运行。进行Web应用开发的时候，结合Servlet和JSP能够使开发的分工和层次更加清晰。JSP注重于页面的呈现，处理逻辑应该尽量减少，这样方便网页设计人员在美工上多下功夫，Servlet程序则侧重于对逻辑的控制，根据需要对JSP页面进行导航。

Servlet是一种独立于平台和协议的服务器端的Java应用程序，可以生成动态的Web页面。Servlet位于Web服务器内部，由Web服务器进行加载，该Web服务器必须包含支持Servlet的Java虚拟机。

JSP(Java Server Pages）是一种动态网页技术标准，是基于Java Servlet和整个Java体系的Web开发技术。它在动态网页的建设中有着强大而特别的功能，它的以下优点使很多人认为它将是最有发展前途的动态网页技术[3]。

1. 将内容的生成和显示进行分离

用JSP技术，Web页面开发人员可以使用HTML或者XML标识来设计和格式化最终页面，并使用JSP标识或者小脚本来生成页面上的动态内容。Web管理人员和页面设计者，能够编辑和使用JSP页面，而不影响内容的生成。

1. 采用标识

Web页面开发人员不会都是熟悉脚本语言的编程人员。JSP技术封装了许多功能，这些功能是在易用的、与JSP相关的XML标识中进行动态内容生成所需要的。

JSP的执行流程是这样的：第一个用户请求JSP文件的时候，JSP容器（如Tomcat）把JSP文件转换成Java文件（Servlet类文件），然后编译成class文件，常驻内存；当有客户请求的时候，直接开一个线程，无须重新编译，直接执行第一次已经编译好的class文件，速度比每次都要重新编译JSP文件要快的多。

**JSP的工作原理**

JSP页面以扩展名为JSP的纯文本文件的形式存在服务器上，当服务器接收到客户端对某个JSP页面的请求时，如果这是对该页面创建或修改后的第一次请求时，那么JSP引擎就把该JSP页面内容转换成一个纯Java语言的Servlet源码，然后将其编译成Servlet字节代码，最后将Servlet的执行结果返回至浏览器。JSP页面被编译成Servlet之后，其执行代码以独立线程的形式一直存在服务器的内存中，以后对该JSP页面提出请求时，不需要重新编译，直到该页面被修改或WEB应用程序重新启动。其工作原理如图3-1所示。



图3-1 JSP的工作原理

**JSP技术语法**

1. 常用JSP命令：

1)page---页面指令，主要用来指定当前JSP页面的属性。

2)<jsp:include>---允许包含其他动态和静态文件。

3)<jsp:forward>---将一个JSP文件传给另一个JSP文件。

4)<jsp:plugin>---可以通过Java插件在浏览器中运行Java Applet 。

5)<jsp:useBean>---标记是用来访问JavaBean的动作标记。当JSP页面使用<jsp:useBean>标记时，表示将引用一个JavaBean的实例。JavaBean标记的语法格式如下：

<jsp:useBean id=“实例名称” scope=“范围” class=“类名称” type=“类的类型” beanName=“JavaBean的名称/>。

1. JSP内置对象[6]：

为了方便Web页面的开发，JSP提供了一些内置的对象。这些内置对象可以不经显式声明而在程序中直接使用，JSP的几个主要内置对象有Request、Response、Session、Out。通过这些内置对象，可以实现页面的一些基本的请求输入、输出功能，且可实现数据在页面之间的交互。本系统主要用到的有以下几个对象[4]：

1)Request对象：主要功能是接受客户端以HTTP方式传送的数据，使服务器端的应用程序可以根据这些数据做进一步的信息处理，实现与客户进行交互的功能。一般常用的是它的getParameter(String name)这个方法来获取传递的参数。其接口为javax.servlet.http.HttpServletRequest。

2)Response对象：主要功能是用来对客户的请求做出响应，与Request对象对应的就是Response对象。Response对象用于将服务器的数据发送到客户端，输出的数据可以是各种数据类型，甚至是文件。一般经常用到的是它的sendRedirect(URL url)这个方法来重定向当前用户访问的url。Response对象的接口为javax.servlet.http.HttpServletResponse。

3)Session对象：主要功能用来保存每个用户在会话期间的数据信息，这样就方便了会话处理工作。当用户登录网站时，系统将为其生成一个独一无二的session对象，用以记录该用户的个人信息，一旦该用户退出网站，那么这个session对象将会注销。Session对象可以绑定若干个人信息或Java对象。Session对象的接口为javax.servlet.http.HttpServletSession。

4)Out对象：主要功能是用于动态地向客户端输出数据信息，其主要作用是在Web浏览器内输出信息。这个对象使用比较频繁，在数据的输出方面经常用到它的println(parameter)这个方法。同时，它也是进行纠错处理时一个比较有效的输出方法。

### 3.2.2 JavaScript简介

JavaScript是适应动态网页制作的需要而诞生的一种新的编程语言，如今越来越广泛地使用于Internet网页制作上。JavaScript是由Netscape公司开发的一种脚本语言(scripting language)，或者称为描述语言。在HTML基础上，使用JavaScript可以开发交互式Web网页。JavaScript的出现使得网页和用户之间实现了一种实时性的、动态的、交互性的关系，使网页包含更多活跃的元素和更加精彩的内容。JavaScript短小精悍，又是在客户机上执行的，大大提高了网页的浏览速度和交互能力。同时它又是专门为制作Web网页而量身定做的一种简单的编程语言。

**JavaScript加入网页有两种办法**[7]：

1.直接加入HTML文档

这是最常用的方法，大部分含有JavaScript的网页都采用这种方法，如：

<script language="Javascript">

<!--

document.writeln("这是Javascript！采用直接插入的方法！");

//-Javascript结束-->

</script>

<script language="Javascript”>用来告诉浏览器这是用JavaScript编写的程序，需要调动相应的解释程序进行解释。

HTML的注释标签<!--和-->：用来去掉浏览器所不能识别的JavaScript源代码的，这对不支持Javascript语言的浏览器来说是很有用的。

//-Javascript结束：双斜杠表示JavaScript的注释部分，即从//开始到行尾的字符都被忽略。<script></script>的位置并不是固定的，可以包含在<head></head> 或<body></body>中的任何地方。

2.引用方式：如果已经存在一个JavaScript源文件（以js为扩展名），则可以采用这种引用的方式，以提高程序代码的利用率。其基本格式如下：

<script src＝url language="Javascript"></script>

其中的url就是程序文件的地址。同样的，这样的语句可以放在HTML文档头部或主体的任何部分。如果要实现“直接插入方式”中所举例子的效果，可以首先创建一个JavaScript源代码文件“Script.js”，其内容如下：document.writeln("这是Javascript！采用直接插入的方法！")。在网页中可以这样调用程序：

<script src=”Script.js” language=”Javascript”></script>

### 3.2.3 MVC模式

MVC即Model-View-Controller，M代表业务逻辑，V代表视图，C代表控制层。通过使用MVC模式就把软件层次分为三个层：模型层、视图层、控制层[1]。

模型（Model）层：也就是软件业务的实现部分。一个软件有什么功能，它如何实现功能，这些功能之间的联系都是在模型层进行处理的。可以这样说，模型层是一个软件的核心，软件功能的好坏、软件功能是否完善，业务层起到了决定性的作用。

视图（View）层：就是用户能看到的并与之交互的界面。在J2EE技术中，常用的表示层技术主要由JSP、JSF、Freemarker等组成。视图向用户显示相关的数据，并能接受用户的输入数据，但是它不能进行任何实际的业务处理。视图还可以接受模型发出来的数据更新事件，从而对用户界面进行同步更新。

控制（Controller）层：Struts中的控制层和Servlet的功能功能一模一样，就是调用业务层，然后进行视图的切换。控制层几乎不做任何与业务有关的事情，比如登录验证这些。所以在软件的分层结构设计中，控制层的代码量几乎是最少的，但它的作用也是非常重要的。

通过将模型、视图与控制层的分离，各个层之间的联系非常少，但这样做的好处是，控制层可以随意的切换到自己想要展示的页面，然后控制层调用业务逻辑层和持久化保存数据层，来实现软件的功能，再转回到控制层，最后将最终的结果显示到视图层，这实际上是一种模型的变化——传播机制。模型、视图、控制层三者之间的关系和各自的主要功能如图3-2所示。



图3- 2 MVC组件关系和功能

### 3.2.4 Struts框架

Struts已经提供了一个非常好的MVC框架，利用Struts开发MVC系统时可以大大加快开发速度。在开发时可以采用的开发流程如下。

1. 收集和定义应用需求。
2. 基于数据采集和显示的原则定义和开发用户界面的需求。
3. 为每一个用户界面JSP文件定义访问路径。
4. 定义ActionMapping建立应用业务逻辑之间的联系。
5. 开发满足用户界面需求的所有支持对象。
6. 基于每一个用户界面需求提供的数据属性来创建对应的ActionForm对象。
7. 开发被ActionMapping调用的Action对象。
8. 开发应用业务逻辑对象（Bean、EJB等）。
9. 对应ActionMapping设计的流程创建JSP页面。
10. 建立合适的配置文件struts-config.xml、web.xml。
11. 开发/测试/部署。

利用Sruts开发MVC流程，如图3-3所示。

模 型

封装应用程序状态

响应状态查询

应用程序功能

通知视图改变



图3- 3 Struts框架MVC流程图

在使用Struts框架时，对应各个部分的具体开发工作主要包括一下部分。

Model部分：采用JavaBean或EJB组件或Spring的Bean，设计和实现系统的业务逻辑。根据不同的请求从Action派生具体Action处理对象。完成“做什么”的任务来调用由Bean构成的业务组件。创建由ActionForm的派生类实现对客户端表单数据的封装及简单的校验。

Controller部分：Struts为我们提供了核心控制部分的实现。只需要配置ActionMapping对象即可完成URI地址的映射及匹配用户界面表单和ActionForm类的对应关系。

View部分：为了使用Model中的ActionForm对象，必须用Struts提供的自定义标记创建HTML表单。利用Struts提供的自定义标记库编写用户界面，把应用逻辑和显示逻辑分离。Struts框架通过这些自定义标记建立了View和Model之间的联系。Struts的自定义标记还提供了很多定制页面的功能。

同时，需要编辑两个配置文件：web.xml和struts-config.xml。通过它们配置Struts系统中的各个模块之间的交互。

图2-2描述了一个Struts应用程序的简要执行流程。

1. 客户端浏览器发出请求。
2. 服务器端的Struts中心类ActionServlet找到struts-config.xml文件，并放入到内存，将文件中的内容作为请求路径映射。
3. ActionServlet类在struts-config.xml文件中找到相关的请求路径映射后填充ActionForm类，将前台传进来的表单域打包成Bean。
4. 然后转到Action类进行业务逻辑功能的实现，比如增、删、改、查数据库中的数据。
5. 再通过struts-config.xml文件的映射，找到功能逻辑处理结束后显示给客户端用户看到的页面，转发功能通过ActionForward对象实现。

由图2-2“Struts框架MVC流程图”结构来看，Struts框架可以将软件的功能进行分层化，这样可以将精通不同技术的人员工作模块化、分工化。比如精通UI用户界面设计的可以只管View视图层，而精通Java程序设计的程序员可以开发Model层的代码。

### 3.2.5 Spring框架

Spring框架是一个开放源代码的轻量级Java框架。主要作用是为了解决企业应用程序维护的复杂性而创建的。Spring框架的解决办法就是用分层架构，分层架构允许程序员选择使用哪一个组件，同时为J2EE应用程序开发提供集成的容器。

Spring框架就是实现了AOP功能的IOC容器。在IOC容器的基础上加入AOP不仅可以做到松耦合开发，还具有面向切面编程的功能。

Spring2.0七大模块的架构，如图3-4所示。



图3- 4 Spring架构

组成 Spring 框架的每个模块（或组件）都可以单独存在，或者与其他一个或多个模块联合实现。每个模块的功能如下：

Spring Core：核心容器提供 Spring 框架的基本功能。核心容器的主要组件是 BeanFactory，它是工厂模式的实现。BeanFactory 使用控制反转 （IOC） 模式将应用程序的配置和依赖性规范与实际的应用程序代码分开。

Spring Context：Spring Context是一个配置文件，向 Spring 框架提供上下文信息。Spring Context包括企业服务，例如 JNDI、EJB、电子邮件、国际化、校验和调度功能。

Spring AOP：通过配置管理特性，Spring AOP 模块直接将面向方面的编程功能集成到了 Spring 框架中。所以，可以很容易地使 Spring 框架管理的任何对象支持 AOP。Spring AOP 模块为基于 Spring 的应用程序中的对象提供了事务管理服务。通过使用 Spring AOP，不用依赖 EJB 组件，就可以将声明性事务管理集成到应用程序中。

Spring DAO：JDBC DAO 抽象层提供了有意义的异常层次结构，可用该结构来管理异常处理和不同数据库供应商抛出的错误消息。异常层次结构简化了错误处理，并且极大地降低了需要编写的异常代码数量（例如打开和关闭连接）。Spring DAO 的面向 JDBC 的异常遵从通用的 DAO 异常层次结构。

Spring ORM：Spring 框架插入了若干个 ORM 框架，从而提供了 ORM 的对象关系工具，其中包括 JDO、Hibernate 和 iBatis SQL Map。所有这些都遵从 Spring 的通用事务和 DAO 异常层次结构。

Spring Web 模块：Web 上下文模块建立在应用程序上下文模块之上，为基于 Web 的应用程序提供了上下文。所以，Spring 框架支持与 Jakarta Struts 的集成。Web 模块还简化了处理多部分请求以及将请求参数绑定到域对象的工作。

Spring Web MVC：MVC 框架是一个全功能的构建 Web 应用程序的 MVC 实现。通过策略接口，MVC 框架变成为高度可配置的，MVC 容纳了大量视图技术，其中包括 JSP、Velocity、Tiles、iText 和 POI。

Spring 框架的功能可以用在任何 J2EE 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 J2EE 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 J2EE 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。

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。下表列出了 IOC 的一个实现模式。

|  |  |
| --- | --- |
| 类型 1 | 服务需要实现专门的接口，通过接口，由对象提供这些服务，可以从对象查询依赖性（例如，需要的附加服务） |
| 类型 2 | 通过 JavaBean 的属性（例如 setter 方法）分配依赖性 |
| 类型 3 | 依赖性以构造函数的形式提供，不以 JavaBean 属性的形式公开 |

Spring 框架的 IOC 容器采用类型 2 和类型3 实现。

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。

Spring 设计的核心是 org.springframework.beans 包，它的设计目标是与 JavaBean 组件一起使用。这个包通常不是由用户直接使用，而是由服务器将其用作其他多数功能的底层中介。下一个最高级抽象是 BeanFactory 接口，它是工厂设计模式的实现，允许通过名称创建和检索对象。BeanFactory 也可以管理对象之间的关系。

BeanFactory 支持两个对象模型。

1. 单态 模型提供了具有特定名称的对象的共享实例，可以在查询时对其进行检索。Singleton 是默认的也是最常用的对象模型。对于无状态服务对象很理想。
2. 原型 模型确保每次检索都会创建单独的对象。在每个用户都需要自己的对象时，原型模型最适合。

bean 工厂的概念是 Spring 作为 IOC 容器的基础。IOC 将处理事情的责任从应用程序代码转移到框架。正如我将在下一个示例中演示的那样，Spring 框架使用 JavaBean 属性和配置数据来指出必须设置的依赖关系。

### 3.2.6 Hibernate框架

Hibernate是目前最流行的ORM框架，其采用非常优雅的方式将Oracle操作完全包装成对象化的操作。其作者Gavin King在持久层设计上极富经验，采用非常少的代码实现了整个框架，同时完全开放源代码，即使偶尔遇到无法理解的情况，也可以参照源代码来理解其在持久层上灵巧而智能的设计。

目前Hibernate在国内的开发人员相当多，Hibernate的文档也非常丰富，这些都为学习Hibernate铺平了道路，因而Hibernate的学习相对简单一些。下面通过对比来了解Hibernate和传统JDBC操作数据库持久层之间的差异。

当前的软件开发语言已经全面转向面向对象，而数据库系统仍停留在关系数据库阶段。面对复杂的企业环境，同时使用面向对象语言和关系数据库是相当麻烦的，不但中间的过度难以理解，而且其开发周期也相当长。

Hibernate是一个面向Java环境的对象/关系数据库映射工具（Object/Relational Mapping），用来把对象模型表示的对象映射到基于Oracle的关系模型数据结构中去。

Hibernate的目标是：释放开发者通常的数据持久化相关的变成任务的95%。对于以数据为中心的程序而言，往往在数据库中使用存储过程来实现商业逻辑，Hibernate可能不是最好的解决方案。但对于那些基于Java的中间件应用中，设计采用面向对象的业务模型和商业逻辑时，Hibernate是最有用的。不管怎样，Hibernate能消除那些针对特定数据库厂商的Oracle代码，并且把结果集由表格式的形式转换成值对象的形式。

Hibernate不仅管理Java类到数据库表的映射（包括Java数据类型到SQL数据类型的映射），还提供数据查询和获取数据的方法，可以大幅度地减少在开发时人工使用SQL和JDBC处理数据的时间。

Hibernate能在众多的ORM框架中脱颖而出，因为Hibernate与其他ORM框架对比具有如下优势。

1. 开源和免费的License，方便需要是研究源代码、改写源代码并进行功能定制。
2. 轻量级封装，避免引入过多复杂的问题，调试容易，减轻程序员负担。
3. 具有可扩展性，API开放。功能不够用时，可以自己编码进行扩展。
4. 开发者活跃，产品有稳定的发展保障。

# 第四章 系统需求分析与前台设计

## 4.1 需求分析

在计算机广泛运用的今天， 应用计算机已经大大的提高了社会的各方面的效率，而作为学校，社会知识的高度集中区，计算机已经成为一种必需品。有了计算机，那么我们就应该在方方面面想到用它来提高我们的效率，对于大学学生的考勤方面，现在越来越体现出其重要性，虽然大学生都已经步入成年，但是其自制力还是很不容乐观，纪律方面还扔需要代课老师和学校的监督，但是对于上课时间点名很耗时间，且老师点名以后对学生的出勤状况也没有一个总体的感知。通过自动点名系统的服务器会根据摄像头采样的对比结果分析学生的出勤情况，并保存在数据库中，再通过系统对出勤情况的统计分析，最终得出各类出勤相关结果，此过程无需任课老师参与，全由计算机后台实现。而学生通过网络，便可以轻松查看自己一个星期，一个月等的出勤情况。本系统重在实现无纸化点名，还有对点结果的统计与分析，同时节约时间与物质成本，让老师的教学更加便捷。

### 4.1.1 系统超级管理员操作平台设计概述

课室自动点名系统按操作平台可分为前台操作与后台操作，从用户角度出发，可以分为“超级管理员”、“学生”、“家长”、“老师”四大角色。其中超级管理员权限最高，具备了其它人员的功能。下面为后台超级管理员的操作说明：

1. 在用户登录之后，选择视频管理器，能够设置拍照参数、拍空课室照片、照片查询、删除照片。
2. 用户可以输入查询条件，查询各用户的基本信息，查询学生时可以对学生的出勤信息进行处理，如修改、下载表单、统计出勤率。
3. 用户可以对已有的出勤信息进行备份和恢复，还会生产数据日志。
4. 用户可以处理课表信息，录入、下载、查询学生课表。
5. 此外，还有课室更改、更改学生座位、公告处理、节假日修改等功能。

对于前台三类角色的主要操作大同小异，说明如下:

1. 用户登录后，可以查询具体相对应的学生的出勤情况，同时也可以

下载相应的出勤表等。

1. 用户可以查看课表信息，对上课的科目，地点，时间还有当节次课的座位号等内容进行浏览。
2. 用户如果有疑惑，可以邮箱留言方式联系管理员。
3. 用户如果在申请了更改座位成功以后，可以在线申请更改座位并等待管理员的审核通知。（该操作仅限学生）
4. 用户可以对后台发布的公告进行浏览查看。

系统功能列表平台划分如下图4-1所示



图4-1

系统结构总图如下图4-2所示



图 4-2

### 4.1.2 系统用例图

系统用例图如下图4-3所示：



图4-3 系统结构用例图

超级管理员的用例图如下3-4所示



图4-4 超级管理员操作用例图

学生操作用例图如下图4-5所示



图4-5 学生操作用例图

## 4.2 系统设计

### 4.2.1 系统层次划分

该系统模块中，模块系统共分为五层，分别为表现层、控制层、业务逻辑层、数据访问层，数据层，如图4-6所示。

图4- 6 网站模块分层

表现层中，系统主要采用了JSP技术，将Action传递过来的数据封装在ActionForm中，进行一定的处理之后，再反馈给前台页面。

控制层主要是为了控制协调系统中各模块的执行流程，在控制层中，Struts中的ActionServlet控制系统流程转向。

业务逻辑层，主要采用Javabean，来实现具体系统的业务逻辑处理。

数据访问层，主要是采用Hibernate持久层方案，通过Hibernate封装对数据层的操作，方便于在不同的数据库系统间移植。Hibernate也采用了面向对象的思想来操作数据库。

数据层存放系统中的数据，本系统采用Oracle数据库

### 4.2.2 数据库设计

数据库就是存放数据的仓库，即数据按照一定的规律存放在计算机中，是信息系统的核心和基础，是信息系统开发和建设的重要组成部分，把信息系统中大量的数据按一定的模型组织起来，提供存储、维护、检索数据的功能，使信息系统可以方便、及时、准确地从数据库中获得所需的信息。数据库设计是信息资源管理最有效的手段，数据库设计就是根据具体的应用环境，构造合理数据库模式，建立数据库逻辑结构，有效存储数据，满足用户实际需求和系统应用处理要求。在动态网站设计中数据库的重要性不言而喻。合理地设计数据库结构可以提高系统运行的效率。如果设计不当，查询起来就非常吃力，程序的性能也会受到影响[2]。

数据库的设计一般主要包括两个部分：

* 对网站系统的设计进行需求分析，确定所需要在数据库中保存的信息。
* 确定每个表的字段属性等数据库信息。

根据系统设计和业务逻辑分析，与我所做模块连接比较紧密的数据表有：

* STUDENT表
* PICTURE表
* COURSE表
* ATTEND表
* STU\_COUNT表
* TAB\_OPERATE表

如表4-7所示，该表为STUDENT表结构，用于保存登录用户的信息。

表4-7 STUDENT表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑名称 | 列名 | 数据类型 | 长度 | 主键 | 是否允许为空 | 说明 |
| 编号 | ID | Number | 10 | Y | N |  |
| 姓名 | Name | NVARCHAR2 | 20 | N | Y |  |
| 权限等级 | Prority | Number | 1 | N | Y |  |
| 系别班级 | Depart | NVARCHAR2 | 20 | N | Y | 外键Class(ID) |
| 座位号 | SeatNo | Number | 3 | N | Y |  |
| 住址 | Address | NVARCHAR2 | 50 | N | Y |  |
| 出生 | Birth | DATE |  | N | Y |  |
| 籍贯 | Origo | Number | 2 | N | Y |  |
| 邮箱 | Email | NVARCHAR2 | 25 | N | Y |  |
| 密码 | Password | NVARCHAR2 | 20 | N | N |  |
| 联系方式 | PhoneNo | VARACHAR | 13 | N | N |  |

如表4-8所示，该表为PICTURE表结构，用于图片信息.

表4-8 PICTURE表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑名称 | 列名 | 数据类型 | 长度 | 主键 | 是否允许为空 | 说明 |
| 编号 | ID | NVARCHAR2 |  | Y | N |  |
| 教室号 | RoomNo | NVARCHAR2 | 7 | N | N | 外键Seat(RoomNo) |
| 日期 | Date | Date |  | N | N |  |
| 节次 | Period | Number | 1 | N | N |  |
| 图片 | Picture | Blob |  | N | N |  |

如表4-9所示，该表为COURSE表结构，用于保存学科的信息。

表4-9 COURSE表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑名称 | 列名 | 数据类型 | 长度 | 主键 | 是否允许为空 | 说明 |
| 编号 | ID | NVARCHAR2 | 未定 | Y | N |  |
| 教室号 | RoomNo | NVARCHAR2 | 7 | N | N | 外键Seat(RoomNo) |
| 科目号 | SubNo | NVARCHAR2 |  | N |  |  |
| 科目名 | Subject | NVARCHAR2 | 20 | N | Y |  |
| 学年 | Year | Number | 4 | N | N |  |
| 学期 | Session | Number | 1 | N | N |  |
| 起始周 | StaWeek | Number | 2 | N | Y |  |
| 结束周 | EndWeek | Number | 2 | N | Y |  |
| 星期几 | Day | Number | 1 | N | N |  |
| 起始时间段 | StaPer | Number | 2 | N | Y |  |
| 上课长度 | Length | Number | 1 | N | Y |  |
| 单双周 | SD | boolean |  | N | Y |  |

如表4-10所示，该表为ATTEND表结构，用于保存某学生的出勤信息。

表4-10 ATTEND表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑名称 | 列名 | 数据类型 | 长度 | 主键 | 是否允许为空 | 说明 |
| 编号 | ID |  |  |  |  |  |
| 学生编号 | StuID | Number | 10 | Y | N | 外键Student(ID) |
| 日期 | Date | Date |  | Y | N |  |
| 节次 | Period | Number | 2 | Y | N |  |
| 课号 | CourseNo |  |  | N | N | 外键Course(ID) |
| 班级 | Class | NVARCHAR2 | 10 | N | N | 外键Class(ID) |
| 备注 | Remark | Number | 1 | N | N |  |

如表4-11所示，该表为STU\_COUNT表结构，用于保存某学生的出勤信息。

表4-11 STU\_COUNT表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑名称 | 列名 | 数据类型 | 长度 | 主键 | 是否允许为空 | 说明 |
| 学生编号 | StuID | Number | 10 | N | N |  |
| 缺勤号 | AttdID |  |  | Y | N |  |
| 个人缺勤次数 | Count | Number | 3 | N | N |  |
| 缺勤科目 | SubID |  |  | N | Y |  |
| 缺勤时间 | Period | Number | 2 | N | Y |  |
| 统计周 | Week | Number | 2 | N | Y |  |
| 日期 | Date | Date |  | N | Y |  |

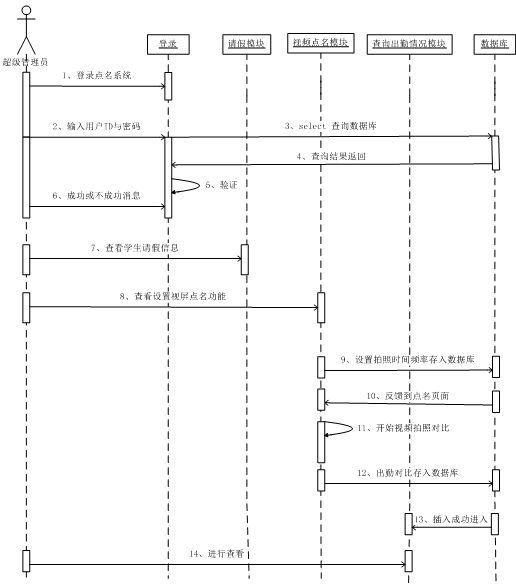
如表4-12所示，该表为TAB\_OPERATE表结构，用于保存某学生的出勤信息。

表4-12 TAB\_OPERATE表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 逻辑名称 | 列名 | 数据类型 | 长度 | 主键 | 是否允许为空 | 说明 |
| 表格编号 | TABLEID | Nvarchar2 | 20 | Y | N |  |
| 用户编号 | USERID | Nvarchar2 | 20 | N | N |  |
| 人员权限控制 | AUTORITYCTR | Number | 2 | N | N |  |
| 视频管理器 | CAMERACTR | Number | 2 | N | Y |  |
| 学生出勤管理 | QUERYBASEIFO | Number | 2 | N | Y |  |
| 课表信息处理 | STUDENTATTCTR | Number | 2 | N | Y |  |
| 课室更改 | COURSELISTCTR | Number | 2 | N | Y | 、 |
| 公告处理 | NOTICECTR | Number | 2 | N | Y |  |
| 节假日修改 | HOLIDAYCTR | Number | 2 | N | Y |  |
| 更改学生坐位 | SEACHANGE | Number | 2 | N | Y |  |

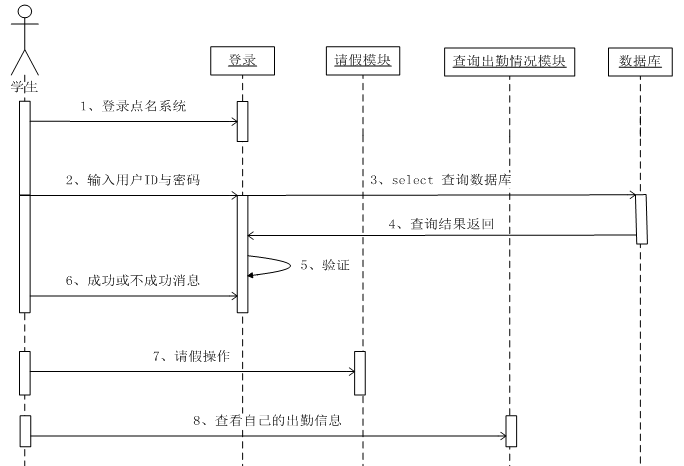
### 4.2.3 点名系统时序图

超级管理员操作时序图如下图4-12 所示



超级管理员操作时序图4-12

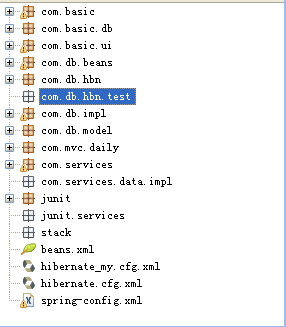
**学生操作时序图如下图4-13 所示：**



学生操作时序图4-13

# 第五章 系统详细设计与功能实现

## 5.1 系统项目的文件夹结构



说明:

com.basic:系统的基础类;主要用以执行系统业务等;

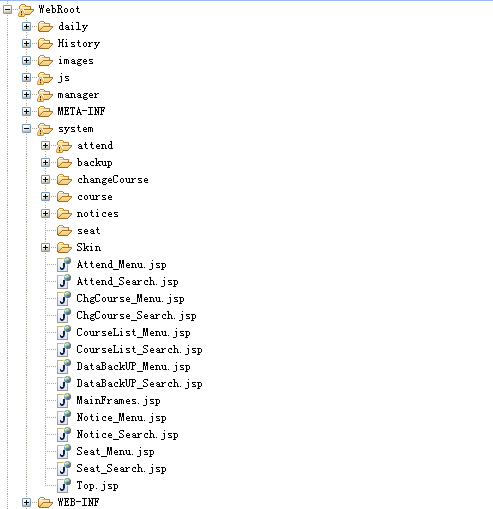
com.basic.db:数据库连接类;

com.db.hbn:Hibernate映射文件;

com.db.beans:与数据库表相对应的JavaBean;

junit:用Junit来测试项目的用例;

WebRoot根目录文件夹



## 5.2 系统运行过程细化分析

**权限设置功能实现中的思路:**

### 5.2.1 项目中的JSP

jsp是在传统的html文件里面插入java程序段而形成了jsp文件，所插入的java程序段可以实现网页的动态性，如操作数据库、重新定向网页等。

如在权限设置功能的页面（authority.jsp）中除了一些表示页面控件table、select、form外有一部分用“<% %>”括起来的java代码。如下：

<%

String authority=request.getParameter("authority");

String authorityval=**null**;

String msg = **null**;

String table = **null**;

**if**(authority!=**null**){

**if**(authority=="comadmin"){

authorityval="1";

table = "TAB\_ADMIN";

}

**if**(authority=="student"){

authorityval="2";

table = "TAB\_STUDENT";

}

**if**(authority=="teacher"){

authorityval="3";

table = "tab\_TEACHER";

}

**if**(authority=="parent"){

authorityval="4";

table = "tab\_PARENT";

}

AuthorityDAO authorityDAO = **new** AuthorityDAO();

**if**(authorityDAO.updateAuthPrority(table,authorityval,(String)session.getAttribute("userid"))){

msg = "修改成功!";

}**else**{

msg = "修改失败!";

}

System.out.println("<br><font color=red>"+ msg +"</font><br><br>");

}

%>

内置对象request.getParameter("authority")获取客户端输入的值，再用if语句判断其值。用DAO类AuthorityDAO 定义一个对象authorityDAO ，并用该对象调用类中的方法updateAuthPrority()，把刚获取的authority值赋给table后作为其参数，对象session.getAttribute()方法获得用户id即userid。

方法updateAuthPrority()属于boolbean类型，返回true或faulse。该方法的实现要看5.2.2节。

### 5.2.2项目中的DAO

DAO (Data Access Objects) 数据访问对象是第一个[面向对象](http://baike.baidu.com/view/125370.htm)的接口，在核心J2EE模式中是这样介绍DAO模式的：为了建立一个健壮的J2EE应用，应该将所有对数据源的访问操作抽象封装在一个公共API中。用程序设计的语言来说，就是建立一个接口，接口中定义了此应用程序中将会用到的所有事务方法。在这个应用程序中，当需要和数据源进行交互的时候则使用这个接口，并且编写一个单独的类来实现这个接口在逻辑上对应这个特定的数据存储。

**package** com.AttSys.dao;

**import** com.AttSys.util.db.DBUtil;

**public** **class** AuthorityDAO {

**private** **static** DBUtil *dbUtil* = **null**;

**static**{

*dbUtil* = **new** DBUtil();

}

**public** **boolean** updateAuthPrority(String table,String authorityval,String userId){

String sql = "update "+table+" set PRORITY='"+authorityval

+"' where u.id='"+userId+"'";

**boolean** flag = **false**;

**if**(*dbUtil*.update(sql)>0){

flag = **true**;

}

**return** flag;

}

}

上一节中的updateAuthPrority()方法是在这里定义的，也就是在AuthorityDAO这个类中定义为一个公共的boolean类型的成员函数，函数中接受了客户端传来的参数作为sql语句查询的条件，即定义了String类型的 sql作为查询语句。

我们看到在类中，有一个DBUtil类定义了个dbUtil对象，且后面该对象实现了*dbUtil*.update(sql)的调用，对该对象的说明在下一节5.2.3。

### 5.2.3项目中的JDBC

JDBC（Java Data Base Connectivity）是一种可用于执行SQL语句的Java API，可以为多种关系数据库提供统一的访问。在没有JDBC这个规范的时候，因为各个DBMS都不一样，我们要访问MySQL数据库要专门为其写一个程序访问;访问Oracle数据库的时候又要为其专门写一个程序，或者访问Informix数据库的时候依然这样。为了解决这问题，JDBC出现了！JDBC使用已有的SQL标准并支持与其它数据库连接的标准。

JDBC应用步骤

1.注册加载一个driver驱动

2.创建数据库连接（Connection）

3.创建一个Statement（发送sql）

4.执行sql语句

5.处理sql结果（select语句）

6.关闭Statement

7.关闭连接Connection。

package com.AttSys.util.db;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

public class DBUtil {

public static final String URL = "jdbc:oracle:thin:@127.0.0.1:1521:orcl";

public static final String USERNAME = "SYSTEM";

public static final String PASSWORD = "abc123";

static{

try {

//加载驱动Oracle

Class.forName("oracle.jdbc.driver.OracleDriver");

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

}

//连接数据库Connection对象

public Connection getConnection(String url,String username,String password){

Connection con = null;

try {

con = DriverManager.getConnection(url, username, password);

} catch (SQLException e) {

e.printStackTrace();

}

return con;

}

//定义一个连接对象 stmt

public Statement getStatement(){

Statement stmt = null;

try {

stmt = getConnection(URL, USERNAME, PASSWORD).createStatement();

} catch (SQLException e) {

e.printStackTrace();

}

return stmt;

}

//处理结果ResultSet

public ResultSet getResultSet(String sql){

Statement stmt = getStatement();

ResultSet rs = null;

try {

rs = stmt.executeQuery(sql);

} catch (SQLException e) {

e.printStackTrace();

}

return rs;

}

//更新数据库的方法update()

public int update(String sql){

Statement stmt = getStatement();

int count = -1;

try {

count = stmt.executeUpdate(sql);

} catch (SQLException e) {//用于处理异常信息

e.printStackTrace();

}

return count;

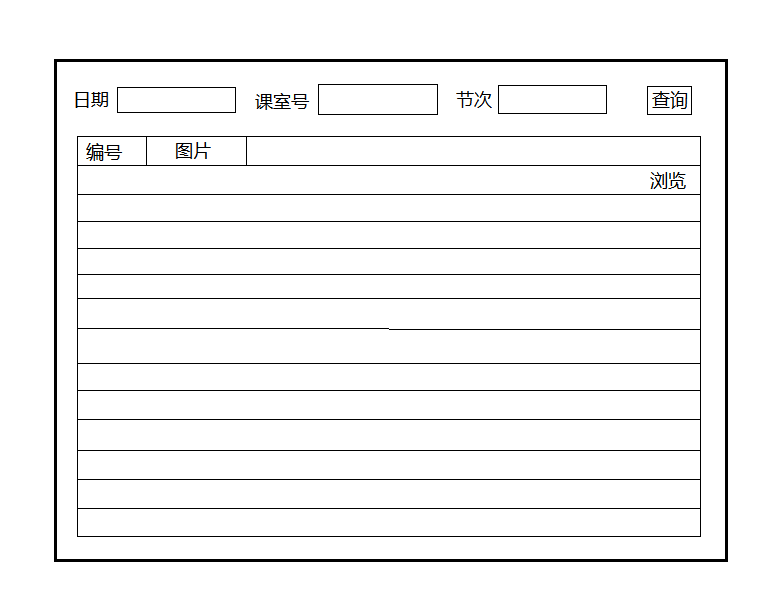
}

}

JDBC之所以可以解决访问多种DBMS的问题，是因为它可以装载各DBMS自带的驱动，如Class.forName("oracle.jdbc.driver.OracleDriver");装载了Oracle提供的驱动。装载驱动后需要与其建立一个连接，如getConnection(String url,String username,String password);这里的username和password为访问数据库的账号和密码。Statement类定义了一个stmt对象用于发送sql语句，即在方法update()方法中执行count = stmt.executeUpdate(sql);，ResultSet接口（结果集），是用来接收select语句返回的查寻结果的。其实质类似于集合。

## 5.3 视频管理器模块的具体实现

### 5.3.1 拍照照片查询功能的实现





上图为照片查询页面，可对输入的日期、课室号、节次来实现查询符合条件的照片列表，点击该列“浏览”可调到下一个界面的图片浏览，并可实现上下张照片的切换。实现的核心代码如下：

<%

String date = request.getParameter("date");

String roomNo = request.getParameter("roomNo");

String period = request.getParameter("period");

List<PictureVO> list = **null**;

//根据条件查询图片信息,返回List.

**if**(date!=**null**||roomNo!=**null**||period!=**null**){

PictureDAO pictureDAO = **new** PictureDAO();

list = pictureDAO.getPictureVO(Date.valueOf(date),roomNo,Integer.valueOf(period));

}

%>

用JSP内置对象request获得date、roomNo和period三个搜索条件，并调用JavaBean中由PictureDAO类对象pictureDAO的getPictureVO()方法进行搜索，而函数getPictureVO() 的定义如下：

**public** List<PictureVO> getPictureVO(Date date,String no,**int** period){

String sql = "select tableID RoomNo Period Picture createtime curstatus from tab\_picture where " +

date!=**null**?"createtime ='"+date+"' ":""+no!=**null**?"RoomNo ='"+no+"' ":""+period!=**null**?"period ='"+period+"'":"";

/\*定义查询符合条件的sql语句\*/

ResultSet rs = *dbUtil*.getResultSet(sql);

PictureVO pictureVO = **null**;

List<PictureVO> list = **new** ArrayList<PictureVO>();

**try** {

**while**(rs.next()){

pictureVO = **new** PictureVO();

pictureVO.setId(rs.getNString(1));

pictureVO.setRoomNo(rs.getNString(2));

pictureVO.setPeriod(rs.getInt(3));

pictureVO.setDate(rs.getDate(4));

pictureVO.setCurstatis(rs.getInt(5));

list.add(pictureVO);

}

} **catch** (SQLException e) {

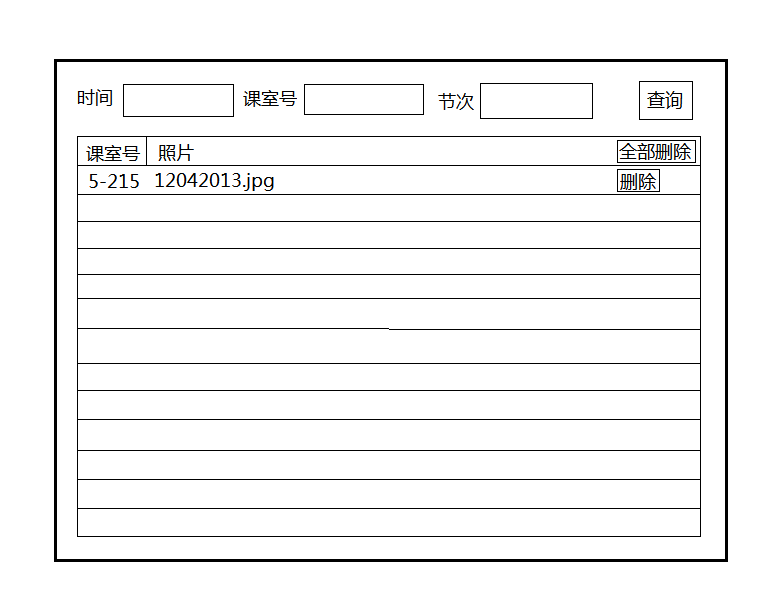
e.printStackTrace();

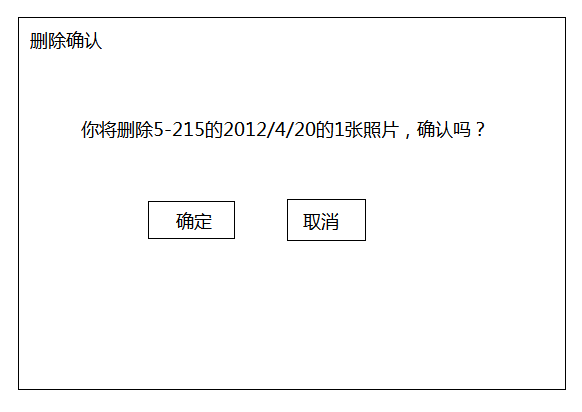
}

**return** list;

}

### 5.3.2 照片删除功能的实现



****

功能实现原理跟上一个功能差不多，也是根据输入的条件来搜索出符合条件的图片列表，再对其进行操作，搜索代码不再重复展示，这里需要提到的是删除照片的代码，根据所选的pictureid作为sql操作条件。

**public** **boolean** deletePictureVO(String id){

**boolean** flag = **false**;

String sql = "delete from tab\_picture "+id!=**null**?"where tableID = '"+id+" '":"";

**if**(*dbUtil*.update(sql)>0){

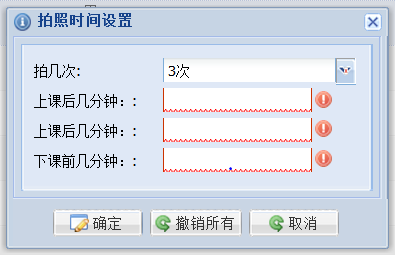
flag = **true**;

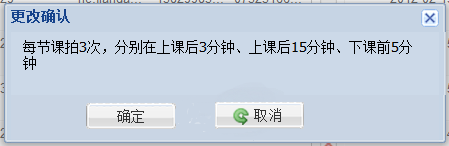
}

**return** flag;

}

### 5.3.3 拍照时间设置功能的实现





在下拉框中选择一节课里的拍照次数，然后设置拍照时间，并更改确认信息后计入数据库表。

<%

String times = request.getParameter("times");

String firstTime = request.getParameter("firstTime");

String secondTime = request.getParameter("secondTime");

String thirdTime = request.getParameter("thirdTime");

String strId = request.getParameter("id");

/\*获取用户输入的拍照次数及时间\*/

String msg = **null**;

/\*修改拍照时间\*/

**if**(times!=**null**&&firstTime!=**null**){

CameraDAO cameraDAO = **new** CameraDAO();

**boolean** flag = cameraDAO.updateCameraTime(Integer.valueOf(times),Integer.valueOf(firstTime),Integer.valueOf(secondTime),Integer.valueOf(thirdTime),Integer.valueOf(strId));

**if**(flag){

msg = "修改成功!";

}**else**{

msg = "修改失败!";

}

System.out.println("<br><font color=red>"+ msg +"</font><br><br>");

}

%>

其中的updateCameraTime()的具体实现如下：

**public** **boolean** updateCameraTime(**int** time,**int** firstTime,**int** secondTime,**int** thirdTime,**int** id){

**boolean** flag = **false**;

String sql = "update tab\_CAMERA set time ='" + time + "' firstTime = '"+firstTime+"' secondTime = '"+secondTime+"' thirdTime = '"+thirdTime+"' where " +"tableID = '"+id+ "'";/\*定义sql更新语句\*/

**if**(*dbUtil*.update(sql)>0){

flag = **true**;

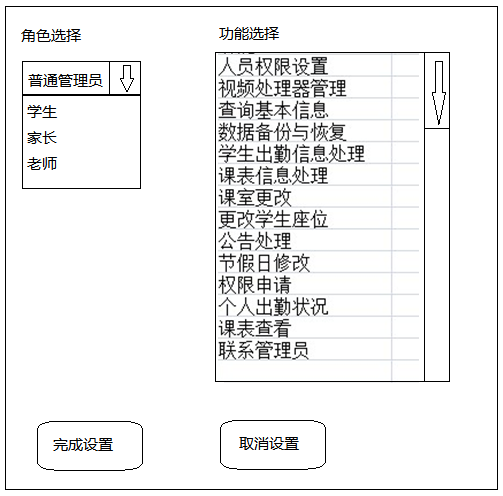
}

**return** flag;

}

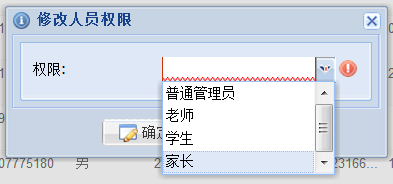
## 5.4 人员权限控制模块的具体实现

### 5.4.1 角色操作功能设置的功能实现



选择角色，并多选其可操作的功能。

### 5.4.2 人员角色设置功能的实现



在查询人员列表的页面中，选中某一行信息单击“权限”选项以进入修改人员权限页面，其中对人员权限jsp中处理业务的java代码为：

<%

String authority=request.getParameter("authority");

String authorityval=**null**;

String msg = **null**;

String table = **null**;

**if**(authority!=**null**){

**if**(authority=="comadmin"){

authorityval="1";

table = "TAB\_ADMIN";

}

**if**(authority=="student"){

authorityval="2";

table = "TAB\_STUDENT";

}

**if**(authority=="teacher"){

authorityval="3";

table = "tab\_TEACHER";

}

**if**(authority=="parent"){

authorityval="4";

table = "tab\_PARENT";

}

AuthorityDAO authorityDAO = **new** AuthorityDAO();

**if**(authorityDAO.updateAuthPrority(table,authorityval,(String)session.getAttribute("userid"))){

msg = "修改成功!";

}**else**{

msg = "修改失败!";

}

System.out.println("<br><font color=red>"+ msg +"</font><br><br>");

}

%>

AuhorityDAO对象authorityDAO的updateAuthPrority()方法实现为：

**public** **boolean** updateAuthPrority(String table,String authorityval,String userId){

String sql = "update "+table+" set PRORITY='"+authorityval

+"' where u.id='"+userId+"'";

**boolean** flag = **false**;

**if**(*dbUtil*.update(sql)>0){

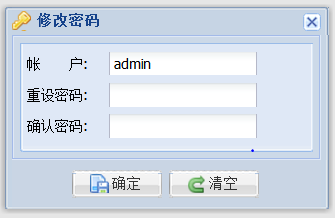
flag = **true**;

}

**return** flag;

}

## 5.5 修改密码模块的具体实现



账户table获取当前session对象的username ，根据用户身份在不同的表里面查找到该用户再对其password进行修改。而更改的操作数据库方法跟上一个功能类同，不再展示。

**if**(prority!=**null**&&username!=**null**&&password!=**null**&&cerPswd!=**null**){

**if**(!password.equals(cerPswd)){

response.sendRedirect("updateUserInfo.jsp?prority="+prority);}

UserDAO userDAO = **new** UserDAO();

**if**(prority=="1"){

table = "TAB\_ADMIN";

}**else** **if**(prority=="2"){

table = "TAB\_STUDENT";

}**else** **if**(prority=="3"){

table = "tab\_TEACHER";

}**else** **if**(prority=="4"){

table = "tab\_PARENT";

}

**if**(userDAO.updateUserPswd(table,password,username)){

msg = "修改成功";

}**else**{

msg = "修改失败";

}

System.out.println("<br><font color=red>"+ msg +"</font><br><br>");

}

%>

# 第六章 总结与展望

## 6.1 课题总结

通过三个月的努力，在老师与同学们的指导帮助下，课室自动点名系统顺利的完成了。

本系统是按照软件工程学的方法来进行设计和开发的，以实现课室自动化点名为目标。通过这次毕业设计，使我对一个系统开发的流程有了更为深刻的理解和认识。极大地增强了自己的动手能力，同时将自己在课堂中学到的理论知识更好地运用在实际应用中。

同时，在这次设计中，我也发现了自己的许多不足。首先，最初开发本系统时，小组对这系统的具体要实现的功能讨论了好久，也走了不少的弯路。其次，对系统没有一个完整的概貌，考虑不是很全面，所以系统功能扩充时，碰到不少困难。再次，我还应该全面地掌握SSH框架技术，好多的细节需要自己去研究学习，不断提高自己网站的开发能力。

通过这次毕业计，在以下几方面有了很大的提高：

1．熟悉了系统完整的开发流程，对开发的流程有了更为深刻的认识。

2．提高了英文阅读和理解能力，逻辑思维能力得到了提高。

3．分析问题和解决问题的能力得到了提高，掌握了解决问题的一般方法。

4．提高了技术总结和撰写论文的能力。

## 6.2 进一步开发的展望

当然，在短短的几个月中，开发出来的这个系统还有很多不足，功能还不是很完善，还有进一步扩展的空间。

1.后台用户应该可以可以设置各个角色的操作,如为学生添加”下载个人出勤表单” 等功能.

2.系统的交互界面可以变得更加友好，对页面的布局和版块，用户可以通过设置改变布局，个性化的定制自己的界面，只显示自己关注的部分功能。

3. 课室自动点名系统从长远来说，并不局限于学生出勤状况的自动化点名。以图像对比为工作原理的点名系统日后更可以为更多的场所提供服务，如停车场的空车位统计、会议厅的出勤状况统计。

# 

# 参考文献

**[1]《Oracle 10g宝典》路川,胡欣杰等编著,电子工业出版社,**

**[2]《**[**Hibernate开发与实战》 刘伟,张利国编著**](http://seek.hzu.edu.cn/detail.htm?292857)**,电子工业出版社**

**[3]《**[**Spring攻略》Gary Mak著；丁雪丰[等]译**](http://seek.hzu.edu.cn/detail.htm?298757)**,人民邮电出版社**

**[4]《Web 2.0快速入门与项目实践》张桂元,贾燕枫,姜波编著；人民邮电出版社**

**[5]《Ajax实战》Dave Crane, Eric Pascarello, Darren James著；人民邮电出版社**

# 致 谢

近三个月时间的毕业设计是我大学生活中忙碌而又充实一段时光。这里有治学严谨而又亲切的老师，有团队里互相帮助的同学，更有积极、向上、融洽的学习生活氛围。短短的时间里，我学到了很多的东西。不仅学到就更多的理论知识，扩展了知识面，提高了自己的实际操作能力；而且学会了如何去学习新的知识，学会了面对困难和挑战，学会了团结合作，互助互利。借此论文之际，向所有帮助、关心、支持我的老师、朋友同学，表达我最真诚的谢意。

首先感谢指导老师。本论文是在老师悉心指导和帮助下完成的。在此，我对彭刚老师的耐心指导和帮助表达我最真诚的谢意，感谢他在这几个月来对我的帮助。在这段时间里，我从他身上，不仅学到了许多的专业知识，更感受到了他工作中严谨的态度,对每一个细节都力求完美这都是非常值得我们学习的地方。

非常感谢队友们。当我在毕业设计过程中遇到问题和困难时，是他们给我提出许多关键性的意见和建议，使我对整个毕业设计的思路有了总体的把握，并耐心的帮我解决了许多实际问题，使我获益良多。

同时，感谢四年来传授我知识的老师们，更要感谢我的家人及朋友对我学业上的支持和鼓励，感谢所有关心、帮助过我的人。同时感谢我的大学，我不会忘记你们。

总之，在以后的学习、工作、生活中我将更加努力，用自己的行动回报社会、学校、老师及同学。

# 附录A 外文原文

**(出处： Malcolm Davis. Struts--an open-source MVC implementation**

**[J]. IBM Systems Journal, 2006,44(2):33-37.)**

**Struts——an open-source MVC implementation**

[**Malcolm Davis**](http://www.ibm.com/developerworks/cn/java/j-struts/#author#author)

This article introduces Struts, a Model-View-Controller implementation that uses servlets and JavaServer Pages (JSP) technology. Struts can help you control change in your Web project and promote specialization. Even if you never implement a system with Struts, you may get some ideas for your future servlets and JSP page implementation.

**Introduction**

Kids in grade school put HTML pages on the Internet. However, there is a monumental difference between a grade school page and a professionally developed Web site. The page designer (or HTML developer) must understand colors, the customer, product flow, page layout, browser compatibility, image creation, JavaScript, and more. Putting a great looking site together takes a lot of work, and most Java developers are more interested in creating a great looking object interface than a user interface. JavaServer Pages (JSP) technology provides the glue between the page designer and the Java developer.

If you have worked on a large-scale Web application, you understand the term *change*. Model-View-Controller (MVC) is a design pattern put together to help control change. MVC decouples interface from business logic and data. Struts is an MVC implementation that uses Servlets 2.2 and JSP 1.1 tags, from the J2EE specifications, as part of the implementation. You may never implement a system with Struts, but looking at Struts may give you some ideas on your future Servlets and JSP implementations.

**Model-View-Controller (MVC)**

JSP tags solved only part of our problem. We still have issues with validation, flow control, and updating the state of the application. This is where MVC comes to the rescue. MVC helps resolve some of the issues with the single module approach by dividing the problem into three categories:

* **Model**   
  The model contains the core of the application's functionality. The model encapsulates the state of the application. Sometimes the only functionality it contains is state. It knows nothing about the view or controller.
* **View**   
  The view provides the presentation of the model. It is the *look* of the application. The view can access the model getters, but it has no knowledge of the setters. In addition, it knows nothing about the controller. The view should be notified when changes to the model occur.
* **Controller**   
  The controller reacts to the user input. It creates and sets the model.

**MVC Model 2**

The Web brought some unique challenges to software developers, most notably the stateless connection between the client and the server. This stateless behavior made it difficult for the model to notify the view of changes. On the Web, the browser has to re-query the server to discover modification to the state of the application.

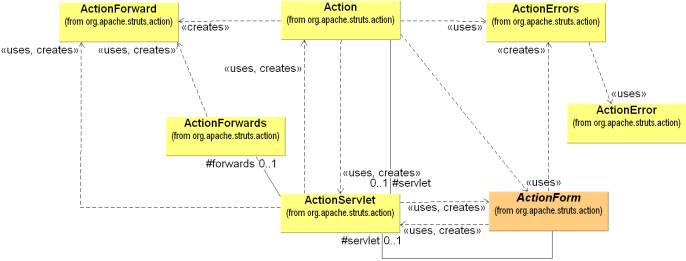
Another noticeable change is that the view uses different technology for implementation than the model or controller. Of course, we could use Java (or PERL, C/C++ or what ever) code to generate HTML. There are several disadvantages to that approach:

* Java programmers should develop services, not HTML.
* Changes to layout would require changes to code.
* Customers of the service should be able to create pages to meet their specific needs.
* The page designer isn't able to have direct involvement in page development.
* HTML embedded into code is ugly.

For the Web, the classical form of MVC needed to change. Figure 4 displays the Web adaptation of MVC, also commonly known as MVC Model 2 or MVC 2.

**Struts details**

Displayed in Figure 6 is a stripped-down UML diagram of the org.apache.struts.action package. Figure 6 shows the minimal relationships among ActionServlet (Controller), ActionForm (Form State), and Action (Model Wrapper).

**Figure 6. UML diagram of the relationship of the Command (ActionServlet) to the Model (Action & ActionForm)**  


**The ActionServlet class**   
Do you remember the days of function mappings? You would map some input event to a pointer to a function. If you where slick, you would place the configuration information into a file and load the file at run time. Function pointer arrays were the good old days of structured programming in C.

Life is better now that we have Java technology, XML, J2EE, and all that. The Struts Controller is a servlet that maps events (an event generally being an HTTP post) to classes. And guess what -- the Controller uses a configuration file so you don\_t have to hard-code the values. Life changes, but stays the same.

ActionServlet is the Command part of the MVC implementation and is the core of the Framework. ActionServlet (Command) creates and uses Action, an ActionForm, and ActionForward. As mentioned earlier, the struts-config.xml file configures the Command. During the creation of the Web project, Action and ActionForm are extended to solve the specific problem space. The file struts-config.xml instructs ActionServlet on how to use the extended classes. There are several advantages to this approach:

* The entire logical flow of the application is in a hierarchical text file. This makes it easier to view and understand, especially with large applications.
* The page designer does not have to wade through Java code to understand the flow of the application.
* The Java developer does not need to recompile code when making flow changes.

Command functionality can be added by extending ActionServlet.

**The ActionForm class**

ActionForm maintains the session state for the Web application. ActionForm is an abstract class that is sub-classed for each input form model. When I say *input form model*, I am saying ActionForm represents a general concept of data that is set or updated by a HTML form. For instance, you may have a UserActionForm that is set by an HTML Form. The Struts framework will:

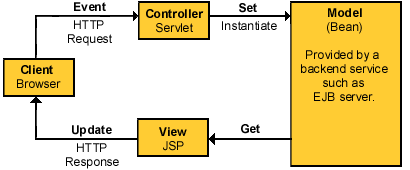
* Check to see if a UserActionForm exists; if not, it will create an instance of the class.
* Struts will set the state of the UserActionForm using corresponding fields from the HttpServletRequest. No more dreadful request.getParameter() calls. For instance, the Struts framework will take fname from request stream and call UserActionForm.setFname().
* The Struts framework updates the state of the UserActionForm before passing it to the business wrapper UserAction.
* Before passing it to the Action class, Struts will also conduct form state validation by calling the validation() method on UserActionForm. **Note:** This is not always wise to do. There might be ways of using UserActionForm in other pages or business objects, where the validation might be different. Validation of the state might be better in the UserAction class.
* The UserActionForm can be maintained at a session level.

Notes:

* The struts-config.xml file controls which HTML form request maps to which ActionForm.
* Multiple requests can be mapped UserActionForm.
* UserActionForm can be mapped over multiple pages for things such as wizards.

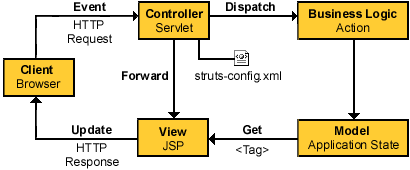
**The Action class**   
The Action class is a wrapper around the business logic. The purpose of Action class is to translate the HttpServletRequest to the business logic. To use Action, subclass and overwrite the process() method.

The ActionServlet (Command) passes the parameterized classes to ActionForm using the perform() method. Again, no more dreadful request.getParameter() calls. By the time the event gets here, the input form data (or HTML form data) has already been translated out of the request stream and into an ActionForm class.

**Figure 4. MVC Model 2**  


**Struts, an MVC 2 implementation**

Struts is a set of cooperating classes, servlets, and JSP tags that make up a reusable MVC 2 design. This definition implies that Struts is a framework, rather than a library, but Struts also contains an extensive tag library and utility classes that work independently of the framework. Figure 5 displays an overview of Struts.

**Figure 5. Struts overview**  


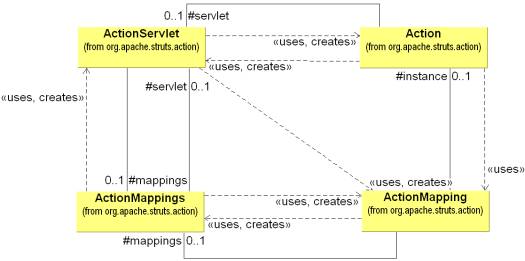
**Struts overview**

* **Client browser**   
  An HTTP request from the client browser creates an event. The Web container will respond with an HTTP response.
* **Controller**   
  The Controller receives the request from the browser, and makes the decision where to send the request. With Struts, the Controller is a command design pattern implemented as a servlet. The struts-config.xml file configures the Controller.
* **Business logic**   
  The business logic updates the state of the model and helps control the flow of the application. With Struts this is done with an Action class as a thin wrapper to the actual business logic.
* **Model state**   
  The model represents the state of the application. The business objects update the application state. ActionForm bean represents the Model state at a session or request level, and not at a persistent level. The JSP file reads information from the ActionForm bean using JSP tags.
* **View**   
  The view is simply a JSP file. There is no flow logic, no business logic, and no model information -- just tags. Tags are one of the things that make Struts unique compared to other frameworks like Velocity.

Note: "Think thin" when extending the Action class. The Action class should control the flow and not the logic of the application. By placing the business logic in a separate package or EJB, we allow flexibility and reuse.

Another way of thinking about Action class is as the Adapter design pattern. The purpose of the Action is to "Convert the interface of a class into another interface the clients expect. Adapter lets classes work together that couldn\_t otherwise because of incompatibility interface" (from *Design Patterns - Elements of Reusable OO Software* by Gof). The client in this instance is the ActionServlet that knows nothing about our specific business class interface. Therefore, Struts provides a business interface it does understand, Action. By extending the Action, we make our business interface compatible with Struts business interface. (An interesting observation is that Action is a class and not an interface. Action started as an interface and changed into a class over time. Nothing's perfect.)

**The Error classes**   
The UML diagram (Figure 6) also included ActionError and ActionErrors. ActionError encapsulates an individual error message. ActionErrors is a container of ActionError classes that the View can access using tags. ActionErrors is Struts way of keeping up with a list of errors.

**Figure 7. UML diagram of the relationship of the Command (ActionServlet) to the Model (Action)**  


**The ActionMapping class**   
An incoming event is normally in the form of an HTTP request, which the servlet Container turns into an HttpServletRequest. The Controller looks at the incoming event and dispatches the request to an Action class. The struts-config.xml determines what Action class the Controller calls. The struts-config.xml configuration information is translated into a set of ActionMapping, which are put into container of ActionMappings. (If you have not noticed it, classes that end with *s* are containers)

The ActionMapping contains the knowledge of how a specific event maps to specific Actions. The ActionServlet (Command) passes the ActionMapping to the Action class via the perform() method. This allows Action to access the information to control flow.

**ActionMappings**   
ActionMappings is a collection of ActionMapping objects.

**Struts pros**

* **Use of JSP tag mechanism**   
  The tag feature promotes reusable code and abstracts Java code from the JSP file. This feature allows nice integration into JSP-based development tools that allow authoring with tags.
* **Tag library**   
  Why re-invent the wheel, or a tag library? If you cannot find something you need in the library, contribute. In addition, Struts provides a starting point if you are learning JSP tag technology.
* **Open source**   
  You have all the advantages of open source, such as being able to see the code and having everyone else using the library reviewing the code. Many eyes make for great code review.
* **Sample MVC implementation**   
  Struts offers some insight if you want to create your own MVC implementation.
* **Manage the problem space**   
  Divide and conquer is a nice way of solving the problem and making the problem manageable. Of course, the sword cuts both ways. The problem is more complex and needs more management.

**Struts cons**

* **Youth**   
  Struts development is still in preliminary form. They are working toward releasing a version 1.0, but as with any 1.0 version, it does not provide all the bells and whistles.
* **Change**   
  The framework is undergoing a rapid amount of change. A great deal of change has occurred between Struts 0.5 and 1.0. You may want to download the most current Struts nightly distributions, to avoid deprecated methods. In the last 6 months, I have seen the Struts library grow from 90K to over 270K. I had to modify my examples several times because of changes in Struts, and I am not going to guarantee my examples will work with the version of Struts you download.
* **Correct level of abstraction**   
  Does Struts provide the correct level of abstraction? What is the proper level of abstraction for the page designer? That is the $64K question. Should we allow a page designer access to Java code in page development? Some frameworks like Velocity say no, and provide yet another language to learn for Web development. There is some validity to limiting Java code access in UI development. Most importantly, give a page designer a little bit of Java, and he will use a lot of Java. I saw this happen all the time in Microsoft ASP development. In ASP development, you were supposed to create COM objects and then write a little ASP script to glue it all together. Instead, the ASP developers would go crazy with ASP script. I would hear "Why wait for a COM developer to create it when I can program it directly with VBScript?" Struts helps limit the amount of Java code required in a JSP file via tag libraries. One such library is the Logic Tag, which manages conditional generation of output, but this does not prevent the UI developer from going nuts with Java code. Whatever type of framework you decide to use, you should understand the environment in which you are deploying and maintaining the framework. Of course, this task is easier said than done.
* **Limited scope**   
  Struts is a Web-based MVC solution that is meant be implemented with HTML, JSP files, and servlets.
* **J2EE application support**   
  Struts requires a servlet container that supports JSP 1.1 and Servlet 2.2 specifications. This alone will not solve all your install issues, unless you are using Tomcat 3.2. I have had a great deal of problems installing the library with Netscape iPlanet 6.0, which is supposedly the first J2EE-compliant application server. I recommend visiting the Struts User Mailing List archive (see [Resources](http://www.ibm.com/developerworks/library/j-struts/index.html?S_TACT=105AGX52&S_CMP=cn-a-j#resources#resources)) when you run into problems.
* **Complexity**   
  Separating the problem into parts introduces complexity. There is no question that some education will have to go on to understand Struts. With the constant changes occurring, this can be frustrating at times. Welcome to the Web.
* **Where is...**   
  I could point out other issues, for instance, where are the client side validations, adaptable workflow, and dynamic strategy pattern for the controller? However, at this point, it is too easy to be a critic, and some of the issues are insignificant, or are reasonable for a 1.0 release. The way the Struts team goes at it, Struts might have these features by the time you read this article, or soon after.

**Future of Struts**

Things change rapidly in this new age of software development. In less than 5 years, I have seen things go from cgi/perl, to ISAPI/NSAPI, to ASP with VB, and now Java and J2EE. Sun is working hard to adapt changes to the JSP/servlet architecture, just as they have in the past with the Java language and API. You can obtain drafts of the new JSP 1.2 and Servlet 2.3 specifications from the Sun Web site. Additionally, a standard tag library for JSP files is appearing.

# 附录B 外文译文

**(译自： Malcolm Davis. Struts--an open-source MVC implementation**

**[J]. IBM Systems Journal , 2006,44(2):33-37.)**

**Struts——MVC 的一种开放源码实现**

c[**Malcolm Davis**](http://www.ibm.com/developerworks/cn/java/j-struts/#author#author)

本文介绍 Struts，它是使用 servlet 和 JavaServer Pages 技术的一种 Model-View-Controller 实现。Struts 可帮助您控制 Web 项目中的变化并提高专业化水平。尽管您可能永远不会用 Struts 实现一个系统，但您可以将其中的一些思想用于您以后的 servlet 和 JSP 网页的实现中。

**简介**

小学生也可以在因特网上发布 HTML 网页。但是，小学生的网页和专业开发的网站有质的区别。网页设计人员（或者 HTML 开发人员）必须理解颜色、用户、生产流程、网页布局、浏览器兼容性、图像创建和 JavaScript 等等。设计漂亮的网站需要做大量的工作，大多数 Java 开发人员更注重创建优美的对象接口，而不是用户界面。JavaServer Pages (JSP) 技术为网页设计人员和 Java 开发人员提供了一种联系钮带。

如果您开发过大型 Web 应用程序，您就理解 *变化* 这个词的含义。“模型-视图-控制器”(MVC) 就是用来帮助您控制变化的一种设计模式。MVC 减弱了业务逻辑接口和数据接口之间的耦合。Struts 是一种 MVC 实现，它将 Servlet 2.2 和 JSP 1.1 标记（属于 J2EE 规范）用作实现的一部分。尽管您可能永远不会用 Struts 实现一个系统，但了解一下 Struts 或许使您能将其中的一些思想用于您以后的 Servlet 的 JSP 实现中。

**模型-视图-控制器 (MVC)**

JSP 标记只解决了部分问题。我们还得处理验证、流程控制和更新应用程序的状态等问题。这正是 MVC 发挥作用的地方。MVC 通过将问题分为三个类别来帮助解决单一模块方法所遇到的某些问题：

* **Model（模型）**   
  模型包含应用程序的核心功能。模型封装了应用程序的状态。有时它包含的唯一功能就是状态。它对视图或控制器一无所知。
* **View（视图）**   
  视图提供模型的表示。它是应用程序的 *外观*。视图可以访问模型的读方法，但不能访问写方法。此外，它对控制器一无所知。当更改模型时，视图应得到通知。
* **Controller（控制器）**   
  控制器对用户的输入作出反应。它创建并设置模型。

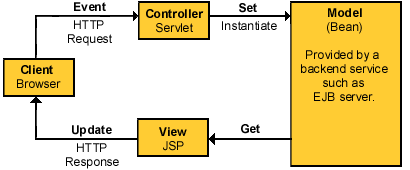
**MVC Model 2**

Web 向软件开发人员提出了一些特有的挑战，最明显的就是客户机和服务器的无状态连接。这种无状态行为使得模型很难将更改通知视图。在 Web 上，为了发现对应用程序状态的修改，浏览器必须重新查询服务器。

另一个重大变化是实现视图所用的技术与实现模型或控制器的技术不同。当然，我们可以使用 Java（或者 PERL、C/C++ 或别的语言）代码生成 HTML。这种方法有几个缺点：

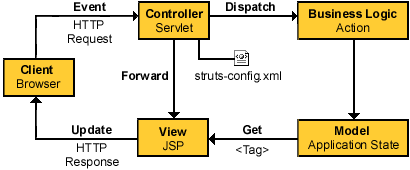
* Java 程序员应该开发服务，而不是 HTML。
* 更改布局时需要更改代码。
* 服务的用户应该能够创建网页来满足它们的特定需要。
* 网页设计人员不能直接参与网页开发。
* 嵌在代码中的 HTML 很难看。

对于 Web，需要修改标准的 MVC 形式。图 4 显示了 MVC 的 Web 改写版，通常也称为 MVC Model 2 或 MVC 2。

**图 4. MVC Model 2**  


**Struts，MVC 2 的一种实现**

Struts 是一组相互协作的类、servlet 和 JSP 标记，它们组成一个可重用的 MVC 2 设计。这个定义表示 Struts 是一个框架，而不是一个库，但 Struts 也包含了丰富的标记库和独立于该框架工作的实用程序类。图 5 显示了 Struts 的一个概览。

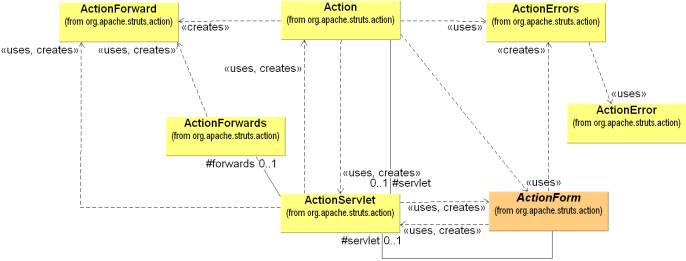
**图 5. Struts 概览**  


**Struts 概览**

* **Client browser（客户浏览器）**   
  来自客户浏览器的每个 HTTP 请求创建一个事件。Web 容器将用一个 HTTP 响应作出响应。
* **Controller（控制器）**   
  控制器接收来自浏览器的请求，并决定将这个请求发往何处。就 Struts 而言，控制器是以 servlet 实现的一个命令设计模式。 struts-config.xml 文件配置控制器。
* **业务逻辑**   
  业务逻辑更新模型的状态，并帮助控制应用程序的流程。就 Struts 而言，这是通过作为实际业务逻辑“瘦”包装的 Action 类完成的。
* **Model（模型）的状态**   
  模型表示应用程序的状态。业务对象更新应用程序的状态。ActionForm bean 在会话级或请求级表示模型的状态，而不是在持久级。JSP 文件使用 JSP 标记读取来自 ActionForm bean 的信息。
* **View（视图）**   
  视图就是一个 JSP 文件。其中没有流程逻辑，没有业务逻辑，也没有模型信息 -- 只有标记。标记是使 Struts 有别于其他框架（如 Velocity）的因素之一。

**详细分析 Struts**

图 6 显示的是 org.apache.struts.action 包的一个最简 UML 图。图 6 显示了 ActionServlet (Controller)、 ActionForm (Form State) 和 Action (Model Wrapper) 之间的最简关系。

**图 6. Command (ActionServlet) 与 Model (Action & ActionForm) 之间的关系的 UML 图**  


**ActionServlet 类**

您还记得函数映射的日子吗？在那时，您会将某些输入事件映射到一个函数指针上。如果您对此比较熟悉，您会将配置信息放入一个文件，并在运行时加载这个文件。函数指针数组曾经是用 C 语言进行结构化编程的很好方法。

现在好多了，我们有了 Java 技术、XML、J2EE，等等。Struts 的控制器是将事件（事件通常是 HTTP post）映射到类的一个 servlet。正如您所料 -- 控制器使用配置文件以使您不必对这些值进行硬编码。时代变了，但方法依旧。

ActionServlet 是该 MVC 实现的 Command 部分，它是这一框架的核心。 ActionServlet (Command) 创建并使用 Action 、 ActionForm 和 ActionForward 。如前所述， struts-config.xml 文件配置该 Command。在创建 Web 项目时，您将扩展 Action 和 ActionForm 来解决特定的问题。文件 struts-config.xml 指示 ActionServlet 如何使用这些扩展的类。这种方法有几个优点：

* 应用程序的整个逻辑流程都存储在一个分层的文本文件中。这使得人们更容易查看和理解它，尤其是对于大型应用程序而言。
* 网页设计人员不必费力地阅读 Java 代码来理解应用程序的流程。
* Java 开发人员也不必在更改流程以后重新编译代码。

可以通过扩展 ActionServlet 来添加 Command 功能。

**ActionForm 类**

ActionForm 维护 Web 应用程序的会话状态。 ActionForm 是一个抽象类，必须为每个输入表单模型创建该类的子类。当我说 *输入表单模型* 时,是指 ActionForm 表示的是由 HTML 表单设置或更新的一般意义上的数据。例如，您可能有一个由 HTML 表单设置的 UserActionForm 。Struts 框架将执行以下操作：

* 检查 UserActionForm 是否存在；如果不存在，它将创建该类的一个实例。
* Struts 将使用 HttpServletRequest 中相应的域设置 UserActionForm 的状态。没有太多讨厌的 request.getParameter() 调用。例如，Struts 框架将从请求流中提取 fname ，并调用 UserActionForm.setFname() 。
* Struts 框架在将 UserActionForm 传递给业务包装 UserAction 之前将更新它的状态。
* 在将它传递给 Action 类之前，Struts 还会对 UserActionForm 调用 validation() 方法进行表单状态验证。 **注：** 这并不总是明智之举。别的网页或业务可能使用 UserActionForm ，在这些地方，验证可能有所不同。在 UserAction 类中进行状态验证可能更好。
* 可在会话级维护 UserActionForm 。

注：

* struts-config.xml 文件控制 HTML 表单请求与 ActionForm 之间的映射关系。
* 可将多个请求映射到 UserActionForm 。
* UserActionForm 可跨多页进行映射，以执行诸如向导之类的操作。

**Action 类**

Action 类是业务逻辑的一个包装。 Action 类的用途是将 HttpServletRequest 转换为业务逻辑。要使用 Action ，请创建它的子类并覆盖process()方法。

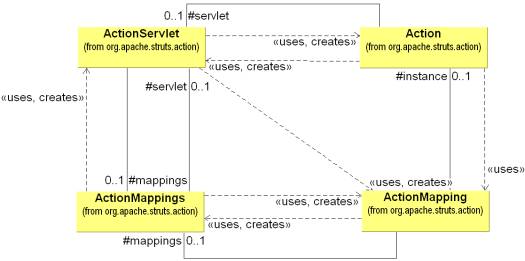
ActionServlet (Command) 使用 perform() 方法将参数化的类传递给 ActionForm 。仍然没有太多讨厌的 request.getParameter() 调用。当事件进展到这一步时，输入表单数据（或 HTML 表单数据）已被从请求流中提取出来并转移到 ActionForm 类中。

注：扩展 Action 类时请注意简洁。 Action 类应该控制应用程序的流程，而不应该控制应用程序的逻辑。通过将业务逻辑放在单独的包或 EJB 中，我们就可以提供更大的灵活性和可重用性。

考虑 Action 类的另一种方式是 Adapter 设计模式。 Action 的用途是“将类的接口转换为客户机所需的另一个接口。Adapter 使类能够协同工作，如果没有 Adapter，则这些类会因为不兼容的接口而无法协同工作。”（摘自 Gof 所著的 *Design Patterns - Elements of Reusable OO Software* ）。本例中的客户机是 ActionServlet ，它对我们的具体业务类接口一无所知。因此，Struts 提供了它能够理解的一个业务接口，即 Action 。通过扩展 Action ，我们使得我们的业务接口与 Struts 业务接口保持兼容。（一个有趣的发现是， Action 是类而不是接口）。 Action 开始为一个接口，后来却变成了一个类。真是金无足赤。）

**Error 类**

UML 图（图 6）还包括 ActionError 和 ActionErrors 。 ActionError 封装了单个错误消息。 ActionErrors 是 ActionError 类的容器，View 可以使用标记访问这些类。 ActionError 是 Struts 保持错误列表的方式。

**图 7. Command (ActionServlet) 与 Model (Action) 之间的关系的 UML 图**  


**ActionMapping 类**

输入事件通常是在 HTTP 请求表单中发生的，servlet 容器将 HTTP 请求转换为 HttpServletRequest 。控制器查看输入事件并将请求分派给某个 Action 类。 struts-config.xml 确定 Controller 调用哪个 Action 类。 struts-config.xml 配置信息被转换为一组 ActionMapping ，而后者又被放入 ActionMappings 容器中。（您可能尚未注意到这一点，以 *s*结尾的类就是容器）

ActionMapping 包含有关特定事件如何映射到特定 Action 的信息。 ActionServlet (Command) 通过 perform() 方法将 ActionMapping 传递给 Action 类。这样就使 Action 可访问用于控制流程的信息。

**ActionMappings**

ActionMappings 是 ActionMapping 对象的一个集合。

**Struts 的优点**

* **JSP 标记机制的使用**   
  标记特性从 JSP 文件获得可重用代码和抽象 Java 代码。这个特性能很好地集成到基于 JSP 的开发工具中，这些工具允许用标记编写代码。
* **标记库**   
  为什么要另发明一种轮子，或标记库呢？如果您在库中找不到您所要的标记，那就自己定义吧。此外，如果您正在学习 JSP 标记技术，则 Struts 为您提供了一个起点。
* **开放源码**   
  您可以获得开放源码的全部优点，比如可以查看代码并让使用库的每个人检查代码。许多人都可以进行很好的代码检查。
* **MVC 实现样例**   
  如果您希望创建您自己的 MVC 实现，则 Struts 可增加您的见识。
* **管理问题空间**   
  分治是解决问题并使问题可管理的极好方法。当然，这是一把双刃剑。问题越来越复杂，并且需要越来越多的管理。

**Struts 的缺点**

* **仍处于发展初期**   
  Struts 开发仍处于初级阶段。他们正在向着发行版本 1.0 而努力，但与任何 1.0 版本一样，它不可能尽善尽美。
* **仍在变化中**   
  这个框架仍在快速变化。Struts 1.0 与 Struts 0.5 相比变化极大。为了避免使用不赞成使用的方法，您可能隔一天就需要下载最新的 Struts。在过去的 6 个月中，我目睹 Struts 库从 90K 增大到 270K 以上。由于 Struts 中的变化，我不得不数次修改我的示例，但我不保证我的示例能与您下载的 Struts 协同工作。
* **正确的抽象级别**   
  Struts 是否提供了正确的抽象级别？对于网页设计人员而言，什么是正确的抽象级别呢？这是一个用 $64K 的文字才能解释清楚的问题。在开发网页的过程中，我们是否应该让网页设计人员访问 Java 代码？某些框架（如 Velocity）说不应该，但它提供了另一种 Web 开发语言让我们学习。在 UI 开发中限制访问 Java 有一定的合理性。最重要的是，如果让网页设计人员使用一点 Java，他将使用大量的 Java。在 Microsoft ASP 的开发中，我总是看到这样的情况。在 ASP 开发中，您应该创建 COM 对象，然后编写少量的 ASP 脚本将这些 COM 对象联系起来。但是，ASP 开发人员会疯狂地使用 ASP 脚本。我会听到这样的话，“既然我可以用 VBScript 直接编写 COM 对象，为什么还要等 COM 开发人员来创建它呢？”通过使用标记库，Struts 有助于限制 JSP 文件中所需的 Java 代码的数量。Logic Tag 就是这样的一种库，它对有条件地生成输出进行管理，但这并不能阻止 UI 开发人员对 Java 代码的狂热。无论您决定使用哪种类型的框架，您都应该了解您要在其中部署和维护该框架的环境。当然，这项任务真是说起来容易做起来难。
* **有限的适用范围**   
  Struts 是一种基于 Web 的 MVC 解决方案，所以必须用 HTML、JSP 文件和 servlet 来实现它。
* **J2EE 应用程序支持**   
  Struts 需要支持 JSP 1.1 和 Servlet 2.2 规范的 servlet 容器。仅凭这一点远不能解决您的全部安装问题，除非使用 Tomcat 3.2。我用 Netscape iPlanet 6.0 安装这个库时遇到一大堆问题，按理说它是第一种符合 J2EE 的应用程序服务器。我建议您在遇到问题时访问 Struts 用户邮件列表的归档资料（请参阅 [参考资源](http://www.ibm.com/developerworks/cn/java/j-struts/#resources#resources)）。
* **复杂性**   
  在将问题分为几个部分的同时也引入了复杂性。毫无疑问，要理解 Struts 必须接受一定的培训。随着变化的不断加入，这有时会令人很沮丧。欢迎访问本网站。

**Struts 的前景**

在这个软件开发的新时代，一切都变得很快。在不到 5 年的时间内，我已经目睹了从 cgi/perl 到 ISAPI/NSAPI、再到使用 VB 的 ASP、一直到现在的 Java 和 J2EE 的变迁。Sun 正在尽力将新的变化反映到 JSP/servlet 体系结构中，正如他们对 Java 语言和 API 所作的更改一样。您可以从 Sun 的网站获得新的 JSP 1.2 和 Servlet 2.3 规范的草案。此外，一个标准 JSP 标记库即将出现。