

**软件设计**

**文档**

**目 录**

[第一章 软件系统总体结构 2](#_Toc23399)

[1.1 子模块的设计 3](#_Toc16715)

[1.1.1 模块功能 3](#_Toc6584)

[1.1.2 模块结构 4](#_Toc14358)

[1.1.3 输入输出 5](#_Toc21562)

[1.1.4 异常处理 9](#_Toc3970)

[1.2 接口设计 10](#_Toc3646)

[1.2.1 外部接口 10](#_Toc8511)

[1.2.2 内部接口 11](#_Toc6093)

[第二章 数据库设计 12](#_Toc19941)

[2.1 数据库说明 12](#_Toc9478)

[（1）实体 12](#_Toc8021)

[（2）联系 12](#_Toc11652)

[2.2 数据库逻辑结构 16](#_Toc15300)

[第三章 软件重用方案 19](#_Toc4232)

[第四章 设计类的重点服务 21](#_Toc21731)

# 第一章 软件系统总体结构

本系统采用B/S(Browser/Server)三层体系结构作为系统的总体结构，由浏览器Web服务器和数据库服务器组成，并综合运用HTML，CSS，JavaScript，Spring

Spring MVC,Mybatis等技术，由Web Server统一进行管理和发送，用户通过Web浏览器以HTTP协议向服务器发出请求，并接受和显示服务器提供的Web信息，具体如图1-1所示

本系统分为两大部分，教师版系统和学生版系统。教师版系统实现题目的增加、删除、修改、查询等功能，以及对于试题的抽取和试卷的批改，还可实现对学生所在的班级进行管理，对考试结果进行综合分析；学生版系统实现在线考试、刷题和个人成绩查询等功能。

系统总体结构图1-1所示：

|  |
| --- |
| 20190505203822380 |
| 图1-1 系统结构图 |

Web层采用SpringMVC框架，接收用户参数，封装数据，调用业务逻辑层完成处理，转发JSP页面完成显示，其中SpringMVC的组件有前端控制器、处理器映射器、处理器映射器、处理器适配器、视图解析器和视图渲染，其图解结构如图1-2所示：

|  |
| --- |
|  |
| 图1-2 SpringMVC结构图解 |

业务逻辑层通过Spring框架来构建，组合DAO层中的简单方法，形成复杂的功能（业务逻辑操作）

dao层使用MyBatis框架来构建，MyBatis 是支持定制化 SQL、存储过程以及高级映射的优秀的持久层框架，其主要就完成2件事情：

①封装JDBC操作

②利用反射打通Java类与SQL语句之间的相互转换

MyBatis的主要设计目的就是让我们对执行SQL语句时对输入输出的数据管理更加方便，所以方便地写出SQL和方便地获取SQL的执行结果才是MyBatis的核心竞争力。

## 1.1 子模块的设计

实质上，计算机科学与工程学院题库系统的综合性相对较强，复杂程度相对较高，可对现有软件进行充分利用，进行系统设计与规划。构建完善成熟的题库管理系统，其中涉及到以下内容，即前台网页界面、处理程序、MySQL后台数据库系统等，在网站页面中显示出以下内容，例如登录页面、课程信息、在线考试、在线作业、在线答疑等。处理程序其实也就是对用户提交表单与相关操作进行处理，存储在后台数据库的信息有用户数据、刷题数据、考试数据、班级数据和分数数据等。

### 1.1.1 模块功能

系统按照功能主要分为学生、老师、系统管理员三个模块，学生具有学生注册、学生登录、学生搜索题库、学生进行刷题、学生查看个人信息；老师具有老师注册登录、老师添加题目、老师设置试卷等功能；系统管理功能包括管理用户、管理科目、管理题目等功能，整个系统模块功能具有如图1-3所示：

|  |
| --- |
|  |
| 图1-3 系统模块功能 |

### 1.1.2 模块结构

系统总体结构可分为系统前台和系统后台两个功能模块。

其中，前台功能实现以下功能，用户注册、用户登录、课程选择、课程信息、在线考试、在线作业、在线答疑、题库、我的提交、我的题库和个人设置。系统前台功能如图1-4所示：

|  |
| --- |
|  |
| 图1-4 系统前台功能 |

系统后台功能实现以下功能，用户管理、题库管理、试卷管理、题型管理和系统设置。系统后台功能如图1-5所示：

|  |
| --- |
|  |
| 图1-5 系统后台功能 |

### 1.1.3 输入输出

输入输出（input/output，I/O）描述的是在计算机上输入输出数据的操作系统、程序或设备。一般的输入输出设备有打印机、硬盘、键盘和鼠标。实际上，有些设备只有输入功能，如键盘和鼠标；有些设备只有输出功能，如打印机；还有些设备具有输入输出2种功能，如硬盘、磁碟和可写性只读光盘（CD-ROM）。

本文主要从程序的输入输出项目来分析一下该系统的输入输出，该系统程序的输入项目如表1-1所示：

**表 1-1 输入项目表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 名称 | 数据类型 | 输入方式 | 数据来源 |
| 1 | 手机号 | char(11) | 手动输入 | 用户 |
| 2 | 用户名 | varchar(30) | 手动输入 | 用户 |
| 3 | 密码 | char(32) | 手动输入 | 用户 |
| 4 | 学号 | varchar(12) | 手动输入 | 用户 |
| 5 | 题目相关信息 | varchar(50) | 手动输入 | 用户 |

该系统程序的输出项目如表1-2所示：

**表 1-2 输出项目表**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 数据类型 | 数据来源 |
| 1 | 成绩 | int(10) | 管理员 |
| 2 | 账号是否冻结（默认0未冻结密码） | varchar(30) | 管理员 |
| 3 | 状态（上架1 下架0） | char(32) | 用户、管理员 |

在数据输入输出的基础上对系统数据流向进行分析就比较容易了。首先，数据流程图（Data Flow Diagram，DFD/Data Flow Chart），是一种能全面地描述系统数据流程的主要工具，它用一组符号来描述整个系统中信息的全貌，综合地反映出信息在系统中的流动、处理和存储情况。

其次，它有两个特征：抽象性和概括性。抽象性指的是数据流程图把具体的组织机构、工作场所、物质流都去掉，只剩下信息和数据存储、流动、使用以及加工情况。概括性则是指数据流程图把系统对各种业务的处理过程联系起来考虑，形成一个总体。综合这两个特征对该系统的数据流向进行分析。

（1）顶层数据流图

|  |
| --- |
|  |
| 图1-6 系统顶层图 |

（2）用户登录数据流

|  |
| --- |
|  |
| 图1-7 题库管理系统访问数据流图 |

（3）查询操作数据流

|  |
| --- |
|  |
| 图1-8 题库管理系统数据查询流图 |

（4）试卷数据流

|  |
| --- |
|  |
| 图1-9 试卷数据流图 |
|  |
| 图1-9（续） 试卷数据流图 |

### 1.1.4 异常处理

异常处理（又称为错误处理）是编程语言或计算机硬件里的一种机制，用于处理软件或信息系统中出现的异常状况（即超出程序正常执行流程的某些特殊条件）。其功能提供了处理程序运行时出现的任何意外或异常情况的方法。本系统的异常处理机制如下：

（1）在应用程序遇到异常情况（如被零除情况或内存不足警告）时，就会产生异常。发生异常时，控制流立即跳转到关联的异常处理程序（如果存在）。

（2）如果给定异常没有异常处理程序，则程序将停止执行，并显示一条错误信息。

（3）可能导致异常的操作通过try关键字来执行。

（4）程序可以使用throw关键字显式地引发异常。

（5）异常对象包含有关错误的详细信息，其中包括调用堆栈的状态以及有关错误的文本说明。

（6）即使引发了异常，finally块中的代码也会执行，从而使程序可以释放资源。

（7）异常处理使用try、catch和finally关键字来尝试可能未成功的操作，处理失败，以及在事后清理资源。异常处理通常是防止未知错误产生所采取的处理措施。

采用异常处理机制的好处是开发者不用再绞尽脑汁去考虑各种错误，这为处理某一类错误提供了一个很有效的方法，使项目软件开发效率大大提高。

异常由公共语言运行库(CLR)、第三方库或使用throw关键字的应用程序代码生成。用户也可以用自定义的异常处理类来扩展php内置的异常处理类。以下的代码说明了在内置的异常处理类中，哪些属性和方法在子类中是可访问和可继承的。

## 1.2 接口设计

本系统软件的接口设计遵循六大基本原则：分别是：单一职责原则、依赖倒置原则、接口隔离原则、迪米特法则、里氏替换原则、开闭原则。这6个原则是建立稳定的，灵活和健壮设计的基础，但设计的时候还要根据实际情况考虑。

（1）单一职责原则(SRP)：该原则要求我们在设计类或者接口的时候。尤其在设计接口的时候把职责分清楚，通常一个职责不是单一的方法，是一类方法的组合。

（2）依赖倒置原则：为高层模块不应该依赖底层模块，他们都应该依赖抽象。抽象不能依赖细节。细节应该依赖抽象，这个原则要求设计的时候尽量用抽象（抽象类或者接口）把各个模块独立开来，实现解耦，使各模块相对独立。简单来说就是要用面向接口设计。

（3）接口隔离原则(ISP)：接口分两种，分别为类实例接口和类接口。这个原则依赖建立在最小的接口之上，依赖自己需要的接口。

（4）迪米特法则(LoD)：也称为最少知识原则。就是一个对象应该对其他对象有最少的了解。这个法则的本质就是解耦，解耦是有限度的不能为了解耦而解耦。

（5）里氏替换原则：（所有引用基类的地方，都能透明地使用其子类的对象。）简单来说就是父类出现的地方，替换为子类不会产出异常。里氏替换原则的好处就是增强程序的健壮性，保持程序的兼容性。

（6）开闭原则：软件的实体如类，模块和函数应该对扩展开发，对修改关闭。这个原则要求我们设计的系统扩展性好，因为需要不会一直不变的，我们需要应对的永远是变化。

### 1.2.1 外部接口

（1）用户界面：在界面设计上，应做到简单明了，易于操作，并且要注意到界面的布局，应突出的显示重要以及出错信息。外观上也要做到合理化，考虑到用户多对Windows风格较熟悉，所以该系统尽量向这一方向靠拢。

（2）软件与硬件接口：本系统设有人机操作界面，考虑到操作简单，易于管理方面，主要硬件与接口设备为pc、鼠标、键盘。而软件接口主要以Windows平台为基本平台。

### 1.2.2 内部接口

内部接口也被称为嵌套接口，这意味着在另一个接口内声明一个接口。例如，Entry接口声明在Map接口中。由于各模块之间相互独立又彼此关联，系统主要通过函数调用实现各部分连接。

# 第二章 数据库设计

数据库的设计关系到整个应用系统的运行效率，数据库设计得好，不仅有利于日常数据的维护更新，而且可以提高系统的运行效率，缩短数据查询响应周期，增加网站的流量。合理的数据库设计可以使围绕它支持的Web页面的Java代码简单化，易于实现，并且可以提高数据存储的效率，保证数据的完整一致。校园二手商品交易系统采用MySQL作为后台数据库开发工具。

## 2.1 数据库说明

概念模型用于信息世界的建模，与具体的DBMS无关。为了把现实世界中的具体事物抽象、组织为某一DBMS支持的数据模型。人们常常首先将现实世界抽象为信息世界，然后再将信息世界转换为机器世界。也就是说，首先把现实世界中的客观对象抽象为某一种信息结构，这种信息结构并不依赖于具体的计算机系统和具体的DBMS，而是概念级的模型，然后再把模型转换为计算机上某一个DBMS支持的数据模型。实际上，概念模型是现实世界到机器世界的一个中间层次。

信息世界中包含的基本概念有实体和联系。

（1）实体

客观存在并可相互区别的事物称为实体。实体可以是具体的人、事、物，也可以是抽象的概念或联系。例如，一个学生、一门课、一个供应商、一个部门、一本书、一位读者等都是实体。

（2）联系

在现实世界中，事物内部以及事物之间是有联系的，这些联系在信息世界中反映为实体内部的联系和实体之间的联系。实体内部的联系通常是组成实体的各属性之间的联系。两个实体型之间的联系可分为3类，一对一联系，(1:1)；一对多联系(1:n)；多对多联系(m:n)。

概念模型是对信息世界建模，所以概念模型应该能够方便、准确地表示信息世界中的常用概念。概念模型的表示方法很多，其中最为常用的是P.P.S.Chen于1976年提出的实体，联系方法(Entity-Relationship Approach)简记为E-R表示法)。该方法用E-R图来描述现实世界的概念模型，称为实体-联系模型，简称E-R模型。

根据数据流程分析，绘制计算机科学与工程学院题库管理系统的全局E-R模型如图2-1所示。

|  |
| --- |
| 未命名文件 (2) |
| 图2-1 系统全局E-R图 |

题库管理系统主要包含教师实体、学生实体、题库实体、试卷信息实体等，如题库实体的属性共包含试题编号、试题章节、试题难度、试题类型、试题分值、试题内容、试题答案和备注等属性。图2-2是题库实体属性图：

|  |
| --- |
|  |
| 图2-2 题库实体属性图 |

试卷信息实体主要包含：试卷编号、课程号、试题编号、试题章节、试题难

度、试题类型、试题分值、试题内容、试题答案和备注等属性。图 2-3 是试卷信息实体的属性图。

|  |
| --- |
|  |
| 图2-3 试卷实体属性图 |

学生实体主要包含：学号、姓名、性别、出生日期、身份证号码、邮箱

照片、备注等属性。图 2-4 是学生实体的属性图。

|  |
| --- |
| UTLWHPFHH]LM`{9~VO0@R{E |
| 表2-4 学生实体属性图 |

E-R 图是由实体型、属性和联系三个基本要素组成。在题库管理系统中，通过前期的需求分析及其数据库理论知识的理解，得出：

1. 一个专业可以包含多名授课教师，一位授课教师只能归属于一个专业，因此专业和授课教师之间具有一对多的联系。
2. 一位授课教师可以教多门课程（科目），一门课程也可以让多个授课教师去教，因此授课教师和科目之间具有多对多的联系。
3. 一个专业可以有多门课程，而一门课程可以归属于多个专业来管理，因此专业和科目之间具有多对多的联系。
4. 一门课程可以由多个章节构成，而一个章节只能归属于一门课程，因此章节和科目之间是一对多的联系。
5. 一个题型可以包含多个题目，而一个题目只能对应于一个题型，因此题型与题目之间是一对多的联系。
6. 一份试卷可以包含多个题型，而一个题型也可以在多个试卷中出现，因此试卷和题型之间是多对多的关系。
7. 一个科目可以有多份试卷，而一份试卷只能是一门课程所出，因此科目和试卷之间是一对多的关系。

题库与试卷信息之间的 E-R 图如图 2-5 所示：

|  |
| --- |
| 0QY[}O17CQWTL[22%0%0GYF |
| 图2-5 题库与试卷之间的E-R图 |

## 2.2 数据库逻辑结构

数据库逻辑设计主要是把数据库概念设计时设计好的基本E-R图转换为与选用DBMS产品所支持的数据模型相符合的逻辑结构。它包括数据项、记录及记录间的联系、安全性和一致性约束等等。导出的逻辑结构是否与概念模式一致，从功能和性能上是否满足用户的要求，要进行模式评价。

通过对系统进行的需求分析，设计出了题库管理系统的主要数据库表。如表 2-1 所示的是学生信息的数据字典，学生信息表主要包含的字段有：学号、姓名、性别、出生年月日、身份证号码、照片、邮箱、备注等。其中学号是学生信息表的主键（主属性），其数据类型是 int 型，不能为空。

表2-1 学生信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 学生信息表 | | | | | |
| 序号 | 名称 | 允许为空 | 数据类型 | 最大长度 | 是否主键 |
| 1 | 学号 | No | char | 12 | 是 |
| 2 | 姓名 | Yes | varchar | 8 | 否 |
| 3 | 性别 | Yes | char | 2 | 否 |
| 4 | 出生年月日 | Yes | datetime | 8 | 否 |
| 5 | 身份证号码 | Yes | char | 18 | 否 |
| 6 | 照片 | Yes | char | 10 | 否 |
| 7 | 邮箱 | Yes | varchar | 15 | 否 |
| 8 | 备注 | Yes | varchar | 10 | 否 |

教师信息表主要包含的字段有：教师号、姓名、性别、出生年月日、工龄、

教研室、职称、照片、备注等。其中教师号是教师信息表的主属性（主键），其

数据类型是 int 型，不能为空，详细情况见表 2-2 所示：

表2-2 教师信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 教师信息表 | | | | | |
| 序号 | 名称 | 允许为空 | 数据类型 | 最大长度 | 是否主键 |
| 1 | 教师号 | No | char | 10 | 是 |
| 2 | 姓名 | Yes | varchar | 8 | 否 |
| 3 | 性别 | Yes | char | 2 | 否 |
| 4 | 出生年月日 | Yes | datetime | 8 | 否 |
| 5 | 教研室 | Yes | varchar | 8 | 否 |
| 6 | 工龄 | Yes | Int | 3 | 否 |
| 7 | 照片 | Yes | char | 10 | 否 |
| 8 | 邮箱 | Yes | varchar | 15 | 否 |
| 9 | 备注 | Yes | varchar | 10 | 否 |

试卷信息表主要包含的字段有：试卷编号、课程号、试题编号、试题章节、

试题难度、试题类型、试题分值、试题内容、试题答案和备注。其中试卷编号是

主键，其数据类型是 int 型，不能为空，试卷信息表的详细信息如表 4-3 所示。

表2-3 试卷信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 试卷信息表 | | | | | |
| 序号 | 名称 | 允许为空 | 数据类型 | 最大长度 | 是否主键 |
| 1 | 试卷编号 | No | Int | 8 | 是 |
| 2 | 课程号 | No | Int | 12 | 否 |
| 3 | 试题编号 | Yes | Int | 10 | 否 |
| 4 | 试题章节 | Yes | char | 20 | 否 |
| 5 | 试题难度 | Yes | varchar | 5 | 否 |
| 6 | 试题类型 | Yes | varchar | 8 | 否 |
| 7 | 试题分值 | Yes | double | 8 | 否 |
| 8 | 试题内容 | Yes | varchar | 1000 | 否 |
| 9 | 试题答案 | Yes | varchar | 1000 | 否 |
| 10 | 备注 | Yes | varchar | 10 | 否 |

题库管理系统的数据库主要包含学生信息表、题型表、案例分析题表、选课信息表等。表 2-4 是题库管理系统的数据库表的清单。

表2-4系统数据表清单

|  |  |  |
| --- | --- | --- |
| 序号 | 数据表名称 | 数据表用途 |
| 1 | 学生信息表 | 用于保存学生身份信息表 |
| 2 | 教师信息表 | 用于保存教师身份信息表 |
| 3 | 管理员信息表 | 用于存储管理员身份信息表 |
| 4 | 用户权限表 | 保存系统各个用户的权限 |
| 5 | 题库表 | 用于存储所有试卷的表 |
| 6 | 试卷信息表 | 用于存储已组成的试卷 |
| 7 | 单选题表 | 保存系统单项选择题的表 |
| 8 | 多选题表 | 保存系统多项选择题的表 |
| 9 | 填空题表 | 保存系统填空题的表 |
| 10 | 判断题表 | 保存系统判断题的表 |
| 11 | 简答题表 | 保存系统简答题的表 |
| 12 | 名词解释表 | 保存系统名词解释的表 |
| 13 | 案例分析题 | 保存系统案例分析的表 |
| 14 | 论述题 | 保存系统论述题的表 |
| 15 | 计算题 | 用于存储系统计算题型的表 |
| 16 | 科目表 | 用于存储科目信息的表 |
| 17 | 难易程度表 | 存储题目难易程度的表 |
| 18 | 章节表 | 用于存储各科题目章节 |
| 19 | 题型表 | 用于存储各种试题的题型 |
| 20 | 课程号表 | 学院课程号总表 |

# 第三章 软件重用方案

从广义的角度来说，本系统在以下三个层次进行了重用，知识的重用，方法和标准的重用以及软件成分的重用。

知识的重用是如软件工程等相关知识的重用，应用相同的知识设计规划、建设不同的软件产品。方法和标准的重用是比如本产品是利用面向对象方法设计又或者是采用国家标准局制定的软件开发文档规范。

软件成分的重用进一步我们划分为三个级别，即代码重用，设计结果重用和分析结果重用。而本软件产品的重用主要在源代码级，这与我们的开发工具有很大的关系，我们选用的开发工具是eclipse和IDE，采用spring MVC框架。利用该框架可以快速开发出一个程序的框架，是一个灵活性较强的框架。

我们的重用设计是建立在MVC框架之上。我们分析可能重用部分的实例有两种方式，一种是继承类库中的构件要用到的基本功能的类。主要是一些界面元素，如菜单活框、列表框，这些构件在很多模块中都要用到，且处理逻辑大致相同，并进行扩充其处理逻辑，如增加输入合法性检测等，这样我们在使用这些经过扩充的构件时不必每个都去重复那些处理逻辑，大大提高了效率，而且，这些构件都是经过测试或其他人使用过的，质量也有保证。

另一种方式是我们的构件在类库找不到相似的类，我们将从头创建自己的类，但为了将自己创建的类，如用户权限处理排序打印预览等，也纳入MVC的框架，我们自己创建的类都继承框架的一个抽象类，这样做的好处是把自己创建的类纳入Controller类的层次化管理，且这也是为实现虚拟函数和动态联编方便。我们也为每个件的功能和接口建立了文档，供应用开发中使用。这样我们就在MVC的框架下，建立了我们自己的一个重用构件库。最后是在应用中重用构件库中的构件。当用户需求变化时，我们能用构件库快速重构我们的应用。

使用重用明显会将开发效率提高一个层次，它不仅提高了开发效率，还保证了应用的风格和质量。特别是对我们这种新手较多的团队开发，采用重用的方法，让有经验的成员负责整个应用的框架，让新手使用重用来创建应用，这非常有利于提高效率，在保证质量的同时也为新手提供一个循序渐进的学习机会，有利于新手的成长。当然使用重用容易，但自己建立重用构件库也是需要代价的，是一个需时间积累的过程，也有一定风险。我们在重用中也遇到了一些问题，一个比较突出的问题是我们的重用主要是在源代码级，因此我们的开发必须在MVC的框架下。

我们的重用主要是在源代码级，通过类的继承来实现。其实可重用的范围是很大的，如设计的重用，测试用列的重用，可运行的代码的重用等。我们想将来扩大重用的粒度，在框架基础上，进一步根据我们单位的软硬件环境定制出一个适用于我们系统的框架，如加进注册功能，数据库连接的功能等。这样，我们可直接重用这个框架，这可以极大地提高软件的开发和维护的效率。最后，我们还设想，将来应有专门的人员管理重用，把重用的维护和应用开发分开，责任明确这样可以更进一步地做好重用工作。

# 第四章 设计类的重点服务

本系统主要设计了三个大的关键类:student,teacher,admin,即学生类，老师类，管理类。基于这三个父类的基础上，继承更行，查询，修改等子类。

设计的类图如图4-1所示：

|  |
| --- |
|  |
| 图4-1 题库系统设计类图 |