**编号**

JNDX_M

**本科生毕业设计（论文）**

**题目：** 面向业务的考试系统的设计与实现

计算机 学院 网络工程 专业

学 号 20122100014

学生姓名 叶豪

指导教师 王立彬 教授

二〇〇一六年六月

摘 要

论文摘要以浓缩的形式概括研究课题的内容，中文摘要在400字左右，外文摘要与中文内容相同，关键词一般以3～5个为妥，词与词之间以“；”为分隔.

设计总说明主要介绍设计任务来源、设计标准、设计原则及主要技术资料，中文字数要在1000～2000字以内，外文字数以500～1000个左右为宜，关键词一般以3～5个为妥，词与词之间以“；”分隔.

**关键词：**毕业论文；模板；规范；图；表

ABSTRACT

It is the English translation of the Chinese abstract. Font: Times New Roman, Word Size: 12. (same as “小四”).

**Keywords**: Thesis; template; criterion; figure; table

目录

[**第1章 绪论 3**](#_Toc448483705)

[**1.1系统设计概要 3**](#_Toc448483706)

[**1.2系统实现概要 3**](#_Toc448483707)

[**第2章 系统业务流程分析与设计 4**](#_Toc448483708)

[**2.1 登陆流程 4**](#_Toc448483709)

[**2.2 系统管理员业务流程 4**](#_Toc448483710)

[**2.3 考生业务流程 5**](#_Toc448483711)

[**第3章 考试系统域模型设计 6**](#_Toc448483712)

[**3.1 数据库表设计 6**](#_Toc448483713)

[**3.1.1 考生表设计 6**](#_Toc448483714)

[**3.1.2 考试表设计 6**](#_Toc448483715)

[**3.1.3 试题表设计 6**](#_Toc448483716)

[**3.1.4 成绩表设计 6**](#_Toc448483717)

[**3.1.5 考生-考试关系表设计 7**](#_Toc448483718)

[**3.1.6 考试-试题关系表设计 7**](#_Toc448483719)

[**第4章 考试管理的设计 8**](#_Toc448483720)

[**4.1 考试时间编辑 8**](#_Toc448483721)

[**4.2 考试参与人员导入、删除 8**](#_Toc448483722)

[**4.3 考试试题导入、删除 8**](#_Toc448483723)

[**第5章 考生管理的设计 9**](#_Toc448483724)

[**5.1 考生列表 9**](#_Toc448483725)

[**5.2 添加考生 9**](#_Toc448483726)

[**5.3 编辑考生 9**](#_Toc448483727)

[**5.4 删除考生 9**](#_Toc448483728)

[**5.4.1 考生的级联删除 9**](#_Toc448483729)

[**5.4.2 考生的批量删除 9**](#_Toc448483730)

[**第6章 试题管理的设计 10**](#_Toc448483731)

[**6.1 添加、编辑试题 10**](#_Toc448483732)

[**6.2 删除试题 10**](#_Toc448483733)

[**6.2.1 数据库中试题的删除 10**](#_Toc448483734)

[**6.2.2 试题对应考试页面文件的删除 10**](#_Toc448483735)

[**6.2.3 试题删除步骤顺序及发生错误的事务回滚 10**](#_Toc448483736)

[**第7章 考生考试及考试业务逻辑的设计 11**](#_Toc448483737)

[**7.1 考试列表 11**](#_Toc448483738)

[**7.2 进入考试 11**](#_Toc448483739)

[**7.3 考试作答 11**](#_Toc448483740)

[**7.4 考试答案提交 11**](#_Toc448483741)

[**第8章 试题结构与试题评分的设计 12**](#_Toc448483742)

[**8.1 试题结构 12**](#_Toc448483743)

[**8.1.1 按钮（Button） 12**](#_Toc448483744)

[**8.1.2 选择框（Select） 12**](#_Toc448483745)

[**8.1.3 文本输入框（Input） 12**](#_Toc448483746)

[**8.2 控件监听 12**](#_Toc448483747)

[**8.2.1 按钮的监听（Button） 13**](#_Toc448483748)

[**8.2.2 文本输入框的监听（Input） 13**](#_Toc448483749)

[**8.2.3 选择框的监听（Select） 13**](#_Toc448483750)

[**8.3 控件操作的消息推送 13**](#_Toc448483751)

[**8.3.1 控件消息推送间的关联以及触发推送 13**](#_Toc448483752)

[**8.3.2 页面动态javascript 13**](#_Toc448483753)

[**8.3.3 控件操作顺序及order-number的设置 14**](#_Toc448483754)

[**8.4 试题评分 14**](#_Toc448483755)

[**8.5 成绩查看 14**](#_Toc448483756)

[**第9章 系统的改进拓展与结论 15**](#_Toc448483757)

[**9.1 系统的改进拓展 15**](#_Toc448483758)

[**9.11 系统管理员不能批量导入考生 15**](#_Toc448483759)

[**9.1.2 试题的添加不面向系统管理员开放 15**](#_Toc448483760)

[**9.1.3 考生考试操作消息推送的并发处理 15**](#_Toc448483761)

[**9.1.4 考生试题提交的并发处理 15**](#_Toc448483762)

[**9.2 结论 15**](#_Toc448483763)

[**参考文献 16**](#_Toc448483764)

[**致 谢 17**](#_Toc448483765)

第1章 绪论

随着信息科技发展，企业逐渐使用电子化的管理系统来进行自身业务的管理。企业对员工的培训中也着重考核员工对于其业务系统操作的熟练程度。本文论点主要围绕一个面向业务的考试系统的设计与实现。

1.1系统设计概要

本系统设计主要分为两部分：

（1）考试系统管理员的管理页面，其中包括

➀ 考生管理；

➁ 考试管理；

➂ 试题管理；

（2）普通考生的的页面，其中包括

➀ 考试页面；

➁ 考试成绩查询；

➂ 个人信息管理；

用户在登陆时选择用户类型（系统管理员或考生），登陆后会跳转到对应类型的页面。

1.2系统实现概要

**1.2.1 交互模式**

本系统采用 B/S 结构，用户通过浏览器与服务器进行交互。

**1.2.2 实现技术**

本系统使用JAVA作为服务器端开发语言，页面使用HTML/CSS/JS作为前端开发技术[1]

开发环境：

JDK版本 1.7

Tomcat 7.1

MySQL 5.6

Eclipse Kepler

操作系统 Ubuntu 14.4

使用框架：

JAVA后台：Spring-MVC、Spring、Hibernate、JPA、FreeMarker

CSS、JS：BootStrap3、jQuery

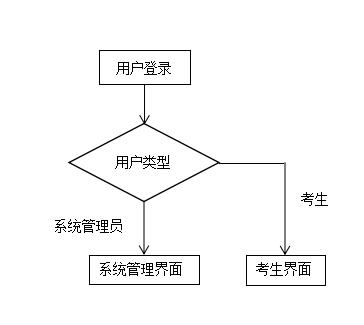
项目构建管理：

Maven、Git

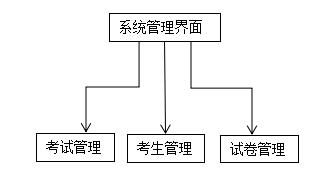
后台系统开发严格遵循MVC模式及表现层、业务层、持久层三层开发模式，使用Spring-MVC 作为表现层框架，Spring为业务逻辑层及Java Bean容器框架，Hibernate为持久层框架。页面 渲染使用FreeMarker，前端页面使用BootStrap及jQuery等开源框架[2]。

1. 系统业务流程分析与设计

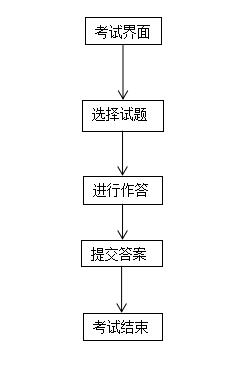
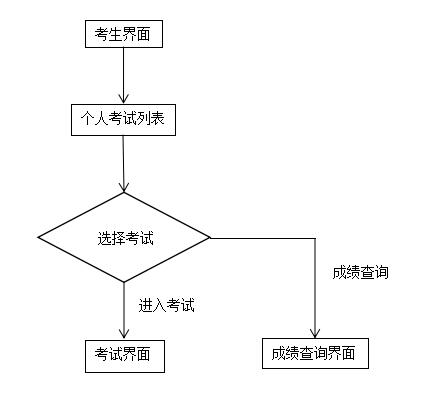
2.1 登陆流程



2.2 系统管理员业务流程



2.3 考生业务流程



第3章 考试系统域模型设计

3.1 数据库表设计

后台域模型共有4张实体域模型表，分别为用户表（User)，考试表(Exam)，试题表(ExamItem)，成绩表(ExamGrade)。

另外域模型还有2张关系表，分别是用户--考试关系表、考试--试题关系表。

3.1.1 考生表设计

用户表与实体类User对应，用于存储系统的用户信息，包括后台管理员与普通用户（考生），两者通过表中的user\_type字段进行区分。包含字段：

id 表中唯一主键

user\_type 用于区分用户类型

username 用户名，表中唯一

password 存储经过哈希函数哈希的用户登录口令

true\_name 用户真实姓名

no 用户编号

unit 用户单位

position 用户职位

basic\_info 用户备注信息

3.1.2 考试表设计

考试表与实体类(Exam)对应，用于存储系统考试信息，包括考试开始，结束时间。包含字段：

id 表中唯一主键

exam\_name 考试名字

generate\_time 考试生成时间

start\_time 考试开始时间

end\_time 考试结束时间

3.1.3 试题表设计

试题表与实体类(ExamItem)对应，用于存储试题信息。包含字段：

id 表中唯一主键

exam\_item\_name 试题名字

generate\_time 试题生成时间

question\_struct 试题内容，使用json格式存储，具体实现请参考第八章

3.1.4 成绩表设计

成绩表与实体类(ExamGrade)对应，用于存储考生考试成绩信息。包含字段：

id 表中唯一主键

candidate\_answer 考生提交的对应考试成绩的答案

is\_submitted 用于检查考生考试提交状态

candidate\_id 指向考生表的外键

exam\_id 指向考试表的外键

3.1.5 考生-考试关系表设计

用户-考试关系表用来存储考生与考试之间的多对多关系，即一个考生可参加多个考试，一个考试也可以包含多个考生。包含字段：

exam\_id 指向考试表的外键

user\_id 指向用户表的外键

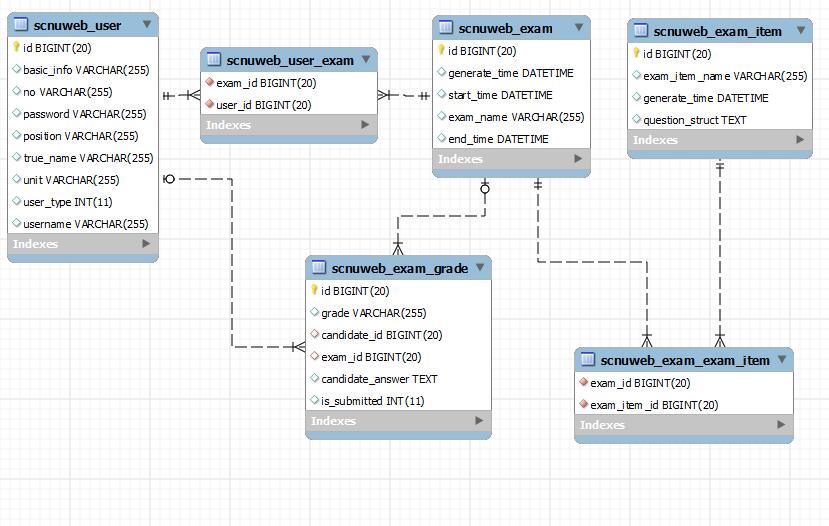
3.1.6 考试-试题关系表设计

考试-试题关系表用来存储考试与试题的多对多关系，即一场考试可以包含多个试题，一个试题也可被多个考试包含。包含字段：

exam\_id 指向考试表的外键

exam\_item\_id 指向试题表的外键

数据库EER图：



第4章 考试管理的设计

后台管理员对考试管理主要是管理考试的起始时间，考试参与人员，考试试题。

4.1 考试时间编辑

考试起始时间分别由2个DATE字段保存。管理员在考试管理界面可以选择起始时间。如果输入的起始时间不合法系统会拒绝接受。

起始时间用来控制考试的开始与结束时间。当考生选择进入考试与提交考试答案时系统会根据当前时间与起始时间的比较去判定操作是否有效。仅当当前时间在起始时间内系统才允许当前操作继续。

4.2 考试参与人员导入、删除

考试与考生存在多对多的关系。即一场考试可有多名考生参加，一名考生可参加多场考试。所以在考试表与考生表（用户表）之间有一个张关系表，用来存储考试与考生之间的多对多关系。

管理员在考试管理页面可以选择已存在的考生进入当前考试。系统对于考试选定考生的实现采用ajax技术，在选择界面选择考生时只需勾选checkbox，即可完成选择。管理员在勾选时页面js脚本监听到后会把勾选的信息在页面不需要重新加载通过异步形式同步到后台。

4.3 考试试题导入、删除

考试与试题存在多对多的关系。即一场考试可包含多个试题，一个试题也可被多个考试包含。考试表与试题表之间有一张关系表，用于存储考试与试题之间的多对多关系。

管理员在考试管理页面可以选择已存在的试题作为当前考试的题目。对于试题选定同样采用ajax技术，提高交互性。

第5章 考生管理的设计

后台管理员拥有对考生的管理能力，即对考生进行添加，编辑，删除等操作。

5.1 考生列表

管理员在考生列表可以通过编号，用户名，姓名，单位来筛选得到考生列表。

考生列表每个单元行包含考生的基本信息，包括编号，用户名，姓名，单位，职务。管理员可通过每行的操作栏对考生进行编辑或删除操作。点击编辑即进入对应考生的编辑界面。

5.2 添加考生

管理员通过点击考生列表底端的“添加考生”按钮进入新增考生界面。在新增考生界面可以输入考生信息来完成考生的注册添加。在页面表单验证使用了jquery validator框架，通过js制定表单验证规则（正则表达式），当管理员输入的表单有误时及时提醒，增强页面交互性。然而对于像用户名这样全局唯一的字段必须保证其在提交到后台前未被其他考生使用。页面通过ajax把管理员输入的用户名异步传送到后台，后台检索并判断当前用户名是否可用，把反馈信息通过json回调到页面来响应用户。

5.3 编辑考生

管理员通过点击考生列表每行的操作栏中的编辑进入考生编辑界面。在考生编辑界面可以编辑考生的信息。编辑考生的页面表单同样使用了jquery validator来进行表单验证。管理员通过覆盖的方式更新考生信息。

5.4 删除考生

5.4.1 考生的级联删除

管理员通过点击考生列表每行操作栏中的删除按钮系统会再次确认是否删除，确认后系统将根据该考生id传至后台将对应考生纪录从数据库中删除。

由于考生表与考试表，考试成绩表均存在级联关系，故删除考生纪录时必须制定其关联表的级联规则。当删除考生时:

在考生-考试关系表中外键user\_id指向当前考生id的纪录都会被删除；

在成绩表中外键candidate\_id指向当前考生id的纪录都会被删除。

上面2种规则可通过JPA里面的注解设置，分别在对应实体类的属性上设置其级联规则为CascadeType.MERGE 和 CascadeType.REMOVE 。

5.4.2 考生的批量删除

考生列表每行的右侧有选定框，勾选多行后点击上方的删除按钮可实现批量删除。批量删除通过js脚本，用jquery迭代选择勾选行来生成删除考试的id列表并用ajax异步将删除列表提交到后台，后台解析列表后根据id迭代删除考生。

第6章 试题管理的设计

管理员组织一场新的考试时，除了需要选定参加考试的考生，还要选定考试内容，即试题。一场考试可包含多个试题。管理员在组织新考试的界面可以选择多个试题作为考试的试题内容。

6.1 添加、编辑试题

考试试题结构以json格式存储在数据库中，试题的添加、编辑并不能很直观通过界面来完成， 故对于试题的添加、编辑是不直接面向管理员，而是通过系统开发维护人员的个性化定制来完成。其中个性化的定制包括了试题对应的考试界面的定制以及对应的试题结构的设计。

6.2 删除试题

在删除某试题的时候必须保证所有包含该试题的考试的状态不能为正在进行。若存在正在进行的考试包含该试题，系统将拒绝该删除操作，否则将会导致正在进行的考试出错。

由于试题与考试直接存在多对多的关系，故试题的删除也需要设置其级联规则。试题的删除分为两个步骤。

6.2.1 数据库中试题的删除

第一步，删除试题存储在数据库中的纪录。考虑该业务逻辑，当一个试题删除时，应该删除该试题与所有包含该试题的考试的关系。试题删除的级联规则如下：

在考试-试题关系表中外键exam\_item\_id指向当前试题id的纪录都被删除。

这样即可完成数据库中试题删除的操作。

6.2.2 试题对应考试页面文件的删除

第二步，除了删除存储在数据库中的纪录，还需删除与试题对应的考试页面。所有的试题的考试页面都存储在服务器目录下的一个文件夹中，并按照试题id进行分类。当试题删除时，系统根据试题id检索并删除在该文件夹中与当前试题对应的考试页面文件。

6.2.3 试题删除步骤顺序及发生错误的事务回滚

由于试题删除操作是分步骤的，系统在删除试题要确保操作的原子性与同步性。删除操作中的数据库操作部分是可由事务管理控制的、可回滚的，而文件操作部分是不可回滚、不可逆的。故在进行删除操作时，应严格确保顺序为数据库操作、文件操作并将这两个操作放入同一个事务中。这样，如果在进行到删除操作的第二步，即使文件操作失败，根据事务的回滚也可保证数据库删除操作的撤销。反之则可能出现两步操作不一致的问题。

第7章 考生考试及考试业务逻辑的设计

考生登录后进入到考生页面后会看到与自己对应的考试列表。考试列表右侧的考试状态栏会提示当前考试的状态，根据当前时间与考试起始时间的比较关系分别有未开始、正在进行、已结束三种状态。当考试状态为正在进行时，考生可在状态右侧点击“进入考试”按钮进入考试界面。当考试状态为已结束时，考生可在状态右侧点击“成绩查询”按钮查看成绩。

7.1 考试列表

考试列表每行展示对应考生的每场考试的信息，包括考试起始时间与考试状态。考试列表是依次根据考试状态、考试结束时间排序的。对于考试状态不同的考试，正在进行的考试会优先排列在顶端；对于相同状态的考试，考试结束时间靠后的会优先排列在前面。这种排列方式可以帮助考生更快速地选择进入考试。

考试状态是由当前时间与考试起始时间决定的，数据库并不会纪录当前考试状态，因为要维护当前考试状态开销太大，更有效率的实现方法是在表现层根据参数判断出状态。在考生的考试列表不大的情况下，在表现层进行考试状态的计算开销是可接受的。而且考生的考试列表是可以分页显示的，每页显示的考试列表不会太大。

7.2 进入考试

考生在考试列表选择要进入的考试并点击“进入考试”即进入对应考试页面。考试页面的左侧为试题列表，可包含多个试题。考试内容即为多个试题的内容。考生需逐个完成试题并选择提交。

7.3 考试作答

考生在考试页面左侧选择试题作答后页面右侧将展示对应题目的业务考核页面。考生通过在该考核页面进行指定的操作来进行作答。考生对于业务考核页面的操作一般仅限于按钮的点击、选择框的选择、输入框的输入。考生通过一系列这样的操作来完成对某一系统某一特定业务流程的操作。考试系统会监听考生的操作信息并在考生选择提交答案时把纪录下来的操作传入后台系统进行存储并对考生的操作进行评分。考生在考试结束后可在考试列表中对应的考试点击查询成绩来查看自己在该场考试的得分。

7.4 考试答案提交

考生在完成试题作答后可选择该试题答案的提交。若考生提交答案的时间需在考试的考试时间范围内，否则系统将拒绝考生的答案提交。若考生在考试结束的时间到达时仍未提交答案，系统将自动把仍未提交答案的试题逐一提交，并强制考生退出当前考试界面。

第8章 试题结构与试题评分的设计

本章将讨论考试试题的存储结构，系统如何监听考生的考试操作以及考试的评分标准。

8.1 试题结构

试题结构是由试题页面控件的集合构成的。控件与后台代码中的控件类(Button、Select、Input)对应。试题结构则是对应后台代码中下面讨论试题页面的类(AnswerList)对应，其中包含了关于控件的一个集合。AnswerList以json格式持久化到数据库中。当需要读取试题结构时则从数据库中读取json的AnswerList并将它实例化为Java对象。下面将介绍对应的控件类。

8.1.1 按钮（Button）

试题页面中含有大量的按钮，包括页面跳转链接，确认按钮，表单提交按钮等。按钮控件的主要目的是监听考核页面中需要操作的按钮。该控件在后台系统代码中对应类Button，包含如下属性：

item id 控件标识id，用于在页面对控件进行操作，监听

order number 控件操作顺序，用于控制监听考生对控件的操作顺序

8.1.2 选择框（Select）

试题页面可能包含有选择框，需要考生从多个选项中选择一个作为该控件的输入值。该控件对应系统后台代码类Select，包含如下属性：

item id 控件标识id，用于在页面对控件进行操作，监听

order number 控件操作顺序，用于控制监听考生对控件的操作顺序

select value 该控件选中的值

is value sensitive 考生对该控件的选择值是否要求与给定值严格一致

8.1.3 文本输入框（Input）

试题页面可能包含有文本输入框，需要考生从指定的输入框中输入文本。该控件对应系统后台代码类Input，包含如下属性：

item id 控件标识id，用于在页面对控件进行操作，监听

order number 控件操作顺序，用于控制监听考生对控件的操作顺序

input value 该控件输入的值

is value sensitive 考生对该控件的输入值是否要求与给定值严格一致

考生对上面几种控件的操作会被系统监听并记录下控件的id，操作顺序，选择或输入值等。下面讨论系统是如何监听控件的。

8.2 控件监听

系统对考试页面控件的监听是通过页面的javascript实现的。为了更方便地操作页面的控件，系统使用了jquery这样的javascript库。jquery能方便地根据控件id在页面检索与之对应的DOM对象。下面分别介绍对3种控件的监听。

8.2.1 按钮的监听（Button）

页面对于按钮的监听相对比较简单。当考试页面加载完时，页面会通过javascript对通过指定id的Button控件进行按钮点击监听器的注册。其代码实现是使用了jquery中的click()函数。在click函数内，按钮将自己的点击操作通过ajax推送到服务器端。所有控件都有自己的order-number，当一个Button控件向服务器端推送点击操作时，系统还会将order-number为Button控件的order-number – 1 的控件（可能是Input或Select）的操作信息通过ajax推送到服务器端。其实现原理请参考本章后续内容。

8.2.2 文本输入框的监听（Input）

页面对于文本输入框的监听与按钮不一样。其页面实现都是通过javascript对页面控件的监听。但其将自己的操作信息推送到服务器端并不是主动的，而是通过与之关联的按钮控件的主动推送触发的。

8.2.3 选择框的监听（Select）

页面对于选择框的监听与文本输入框一样。其页面实现都是通过javascript对页面控件的监听。但其将自己的操作信息推送到服务器端并不是主动的，而是通过与之关联的按钮控件的主动推送触发的。

8.3 控件操作的消息推送

8.3.1 控件消息推送间的关联以及触发推送

系统对所有控件都有一个order-number属性来对控件的提交顺序进行判断和控制。

对于一个考核业务流程来说，有些控件的操作顺序是无关紧要的。比如一个表单中的输入框、选择框的输入顺序是没有严格要求的。但对于按钮控件，它们的操作顺序相对于其他2种控件甚至包括同样为按钮的其他控件来说都是有要求的。也就是说，只有按钮控件是推进业务操作流程的关键点。故我们对考生操作流程的监听只需要针对按钮控件，而其他类型的控件则可通过关联的按钮控件去触发推送操作消息。下面介绍控件的关联方式。

一般来说，除了按钮之外的2种控件相互之间是不会存在关联的，即它们之间的操作顺序没有严格要求。故存在关联关系的2个控件之间必至少有1个为按钮控件。我们以每个按钮控件为监听点，将order-number为当前按钮控件的order-number -1 的控件的操作信息一起推送到服务器端。

总而言之，只有按钮控件是有主动推送能力，其他类型的控件只能通过按钮控件的推送来触发推送。这样能保证对用户操作的有序监听。

8.3.2 页面动态javascript

对于不同的试题，生成的考试页面的javascript代码也是不同的。系统根据请求的试题页面，从数据库的试题表中检索对应试题的结构，并将该包含试题所有有效控件的控件集合的信息通过代码动态构建出一个js文件返回到试题页面中。动态生成的js在页面可以针对不同试题页面的控件进行控制。

8.3.3 控件操作顺序及order-number的设置

在试题设计定制时，为保证系统能监听考生对业务的操作流程，设计人员必须对所有监听控件设置一个order-number来控制考生在考试中的操作顺序。比如现在要求考生在输入文本框的内容后点击一个按钮，则可设置文本框的order-number为1，按钮的order-number为2。若对于操作顺序无要求的控件，则可将它们的order-number设置为同一值即可。

8.4 试题评分

考生对试题的操作信息通过ajax传递到服务器端后，服务器采用session存储当前考生所有的操作信息。其操作信息以List的形式存放在session中，list中每个元素即为考生操作的控件对象（Button,Select,Input）。当考生选择提交该试题的答案时，后台将session中的list对象传递给Service层进行评分。Service层的评分实现即把传入的考生提交的list与从数据库检索得到的试题答案作比较，试题评分标准如下。

试题评分的标准是依照考生在该试题操作流程中正确操作的步骤数与总的操作步骤数的比值。实际即为考生正确操作的控件个数与总的监听控件个数的比值。

系统判别考生是否正确操控控件有2个判断步骤：

1. 考生操作控件顺序与试题答案要求的操作顺序一致；
2. 若操作的控件不为按钮，若is value sensitive 为false，则系统判断通过，若is value sensitive为true，再判断考生输入的value是否与给定的答案的输入值一致，若为一致则通过。

系统将计算的比值乘以100后保留2位有效数字的数值作为考生在该道试题的得分。

8.5 成绩查看

考生在考试结束后可以在考试列表对应考试行点击成绩查询进入该场考试的成绩查询界面。考试成绩将展示考生在该场考试的每个试题对应的成绩。若考生没有提交某个试题或为进入该试题进行作答则该道试题分数为0。若考生未参与该场考试，则系统会提示该考生成绩不存在。

第9章 系统的改进拓展与结论

9.1 系统的改进拓展

9.11 系统管理员不能批量导入考生

系统管理员在考生管理页面添加考生不能通过Excel批量导入，只能逐个导入。且导入考生不能为其设置随机的初始密码。在未来的改进中，系统可支持Excel批量导入考生，并为其设置随机的初始密码。

9.1.2 试题的添加不面向系统管理员开放

上面的章节曾经说到，考试系统的试题添加需要个性化定制，需要由系统设计人员进行考试页面的设计与试题结构的编写。这样使得系统不能灵活方便地添加试题。在未来的改进中，如果技术允许，可考虑实现所见即所得的试题编辑页面，通过大量的javascript的页面代码使得管理员对试题的编辑可视化，可拖拽、编辑控件来实现对试题的编辑，编辑完毕系统还可以对页面进行自动排版，优化。这样考试系统对试题的添加，编辑就不需要依赖系统设计人员即可完成。

9.1.3 考生考试操作消息推送的并发处理

考生在考试界面对控件的操作信息会通过ajax推送到后台服务器。如果参与该场考试的考生比较多，且其操作信息推送比较密集，导致对后台的并发访问量过大会引起服务器端的并发性能瓶颈。所以我们可以考虑以页面端缓存与服务器端缓存两者结合的方式来消除并发量。即对于考生的操作信息采取延迟推送，当操作信息达到一定的量时再向服务器推送信息。这样可有效地减少操作消息推送的并发量。

9.1.4 考生试题提交的并发处理

考生在试题提交时，系统会即时对考生提交的答案进行评分，而不是在考试结束时统一评分。这样处理的目的是为了减少在考试结束时所有考生的答案自动提交时引起并发过大。但由于很多考生还是在考试结束后统一提交试题答案，则在考试结束时的并发可能还是很大。对此的改进我们可以在服务器端使用消息缓冲队列（JMS），考生答案提交请求会存放在一个缓冲队列中，系统在空闲的时候从队列中取出请求并进行业务操作。这样可以对并发进行错峰，防止出现并发瓶颈。

9.2 结论

本文主要围绕着一个面向业务的考试系统的设计与简单实现进行探讨。该系统的主要设计难点在于系统如何探知考生的考生操作以及控制考试的有序进行。系统将考生的操作信息对象化成页面的操作控件并使用序号的方式控制其操作顺序。这种模式很好地实现了试题结构与页面控件的映射化，具有较强的可拓展性。该系统虽然存在很多不足的地方，但其设计为这种面向业务的考试系统提供了一种简单易实现的方案，希望能够为大家所借鉴与学习。

参考文献

1. Bruce Eckel，陈昊鹏译. Thinking in Java, Fourth Edition [M]. 北京：机械工业出版社，2007.

[2] Bear Bibeault. jQuery in Action 3rd Edition [M]. Manning Publications，2008.

致 谢

致谢应以简短的文字对在课题研究和设计说明书（论文）撰写过程中曾直接给予帮助的人员或单位表示自己的谢意，这不仅是一种礼貌，也是对他人劳动的尊重，是治学者应有的思想作风，比如：本文是在导师XXX教授和XXX讲师的悉心指导下完成的，表示谢意!

感谢XX.