|  |
| --- |
| **学士学位论文** |
| **软件测试用例在线评判** |
| |  |  | | --- | --- | | 姓　 名： | 王肖辉 | | 学 科 专 业： | 软件工程 | | 指 导 教 师： | 张剑波 副教授 | | 培 养 单 位： | 信息工程学院 | |
| 2017年6月 |

学士学位论文原创性声明

本人郑重声明：本人所呈交的学士学位论文《软件测试用例在线评判》，是本人在指导老师的指导下，在中国地质大学（武汉）攻读学士学位期间独立进行研究工作所取得的成果。论文中除已注明部分外不包含他人已发表或撰写过的研究成果，本人所呈交的学士学位论文没有违反学术道德和学术规范，没有侵权行为，并愿意承担由此而产生的法律责任和法律后果。

学位论文作者签名：

日 期： 年 月 日

摘要

在如今的软件测试教学中，依然偏重于理论的教学，对于学生学生的动手设计测试用例的能力的锻炼还不是很到位。在软件工程中，软件测试的地位越来越重要，设计软件测试用例，对学生以后的真正的项目开发至关重要。目前软件测试实践还限制于设备环境，要完成测试学生需要安装各种的测试工具，且对于测试的结果的评判也限制与人工，由于测试结果分散与各自的环境中也不利于总结反思。所以本论文希望设计一款系统，该系统能够接收Web用户提交的用例脚本，然后由测评系统对测试的用例分析，给出评价。

目前有许多的在线测评系统，但是大部分都是用来判断代码程序的正确性，及代码执行过程中的执行信息参数。目前市场上有着形形色色的软件测试的框架和工具，但是需要安装环境，且测试结果也不便于持久保存。所以本系统拟结合两者来实现一种在线软件测试用例评判，参照目前流行的OJ系统原理，使用Web开发技术完成Web端的开发，实现系统元素信息的操作。利用合适的软件测试工具作为测评机的开发基础，将软件测试工具整合到测评机中，实现对软件测试用例的评测，目前测评任务主要是对覆盖率的统计。二者通过ApacheMQ发送异步消息进行通信。

在本系统中Web 端和测评机相互分离的，所以扩展比较方便。Web端几乎不需要修改，只需要修改测评机，且测评机的测评对测评实现就行了抽象封装，扩展只需要实现测评接口。

**关键词**：在线测评；测评机；ApacheMQ；扩展

Abstract

In today's software testing teaching, still emphasis on the theory of teaching, for students ,to design hands-on test cases of the ability to exercise is not in place. Software testing, as an important subject area in software engineering, is critical in practical project development. The current software testing practice is also limited to the device environment, if students want to complete the test, students need to install a variety of test tools, and the test results are also limited with the evaluation of artificial, due to scattered test results and their respective environment is also not conducive to sum up reflection. Therefore, this paper hopes to design a system, to achieve the test case online evaluation.

There are many online evaluation systems, but most are used to determine the correctness of the code program, and the implementation of the code in the implementation of information parameters. There are many frameworks and tools for software testing, but the installation environment is required and the test results are not easy to keep lasting. Therefore, this system intends to combine OJ and Web development to achieve an online software test case judgments.With referenceing to the current popular OJ system principle and the application of Web development technology to complete the development of the Web side to achieve the operation of the system element information. The appropriate software testing tools integrated into the evaluation machine to achieve the evaluation of software test cases.The current evaluation of the task is mainly on the coverage of the statistics. Both send an asynchronous message via ApacheMQ for communication each other.

In this system, the Web side and the evaluation machine are separated from each other, so the expansion is more convenient. Web side almost no need to modify, only need to modify the evaluation machine, and the evaluation of the evaluation of the evaluation on the line to achieve an abstract package, the expansion only need to achieve evaluation interface.

**Keywords:** the evaluation machine; online evaluation; ApacheMQ; expansion

目录

目录

[图清单 I](#_Toc482045574)

[表清单 II](#_Toc482045575)

[第一章 绪论 1](#_Toc482045576)

[1.1 研究背景与意义 1](#_Toc482045577)

[1.2 国内外研究现状 1](#_Toc482045578)

[1.3 研究目标和内容 1](#_Toc482045579)

[1.4 论文的组织结构 1](#_Toc482045580)

[第二章 技术支持 2](#_Toc482045581)

[2.1 开发平台及框架 2](#_Toc482045582)

[2.2 消息通信MQ 2](#_Toc482045583)

[2.3 覆盖率统计 2](#_Toc482045584)

[2.3.1 java 2](#_Toc482045585)

[2.3.2 Python 2](#_Toc482045586)

[2.4本章小结 2](#_Toc482045587)

[第三章 系统需求分析及结构设计 2](#_Toc482045588)

[3.1 需求分析 2](#_Toc482045589)

[3.2.1 系统概述 2](#_Toc482045590)

[3.2.2 系统目标 2](#_Toc482045591)

[3.2.3 系统需求 2](#_Toc482045592)

[3.2 系统设计目标和原则 3](#_Toc482045593)

[3.3 系统技术架构设计 3](#_Toc482045594)

[3.2.1 逻辑架构 3](#_Toc482045595)

[3.2.2 部署架构 3](#_Toc482045596)

[3.2.3 实现架构 3](#_Toc482045597)

[3.2.4 数据架构 3](#_Toc482045598)

[3.4 系统功能架构 3](#_Toc482045599)

[3.5 本章小结 3](#_Toc482045600)

[第四章 系统详细设计 4](#_Toc482045601)

[4.1 功能模块设计 4](#_Toc482045602)

[4.1.1 XX模块1 4](#_Toc482045603)

[4.1.2 XX模块2 4](#_Toc482045604)

[4.2 数据库设计 4](#_Toc482045605)

[4.2.1 数据库总体结构 4](#_Toc482045606)

[4.2.2 数据库逻辑设计 4](#_Toc482045607)

[4.2.3 数据库物理设计 4](#_Toc482045608)

[4.2.4 数据库管理要求 4](#_Toc482045609)

[4.3 用户界面设计 4](#_Toc482045610)

[4.3.1 用户界面设计原则 5](#_Toc482045611)

[4.3.2 界面设计 5](#_Toc482045612)

[4.4 程序模块设计 5](#_Toc482045613)

[4.4.1 XX程序模块1 5](#_Toc482045614)

[4.5 安全性设计 5](#_Toc482045615)

[4.5.1 用户权限的划分 5](#_Toc482045616)

[4.5.2数据的一致性 5](#_Toc482045617)

[4.5 本章小结 5](#_Toc482045618)

[第五章 系统实现与测试 6](#_Toc482045619)

[5.1 系统整体实现 6](#_Toc482045620)

[5.2 关键点的实现 6](#_Toc482045621)

[5.2.1 XXX实现1 6](#_Toc482045622)

[5.3 系统测试 6](#_Toc482045623)

[5.3.1 系统测试环境与方案 6](#_Toc482045624)

[5.3.2 系统测试数据与过程 6](#_Toc482045625)

[5.3.3 系统测试结果与分析 6](#_Toc482045626)

[5.3 本章小结 6](#_Toc482045627)

[第六章 总结和展望 7](#_Toc482045628)

[6.1 总结 7](#_Toc482045629)

[6.2 展望 7](#_Toc482045630)

[致谢 8](#_Toc482045631)

[参考文献 9](#_Toc482045632)

[附录 10](#_Toc482045633)

# 图清单

[图表 1-1在线测评系统工作流程图表格 3](#_Toc482721592)

[图表 2-1 Spring Web MVC 处理请求流程 4](#_Toc482721593)

[图表2-3 nose-coverage覆盖率统计结果 6](#_Toc482721594)

# 表清单

# 绪论

## 研究背景和意义

软件测试是软件工程范畴中一个重要学科[1]，为了强化培养学生在软件开发领域的实践、设计以及创新能力，必须加强软件测试的实践教学。目前，在软件测试教学中实践教学存在着诸多的不足，一般都为教师布置实践内容，学生在各自的计算机上完成测试，对于测试的结果老师也不方面统计与分析。一方面，需要为每一台机器配置一套测试框架，造成了资源的浪费，另一方面，测试的结果也都是临时数据，且分散在各自的机器上，很难统一管理，也不利于软件测试的实践教学。

目前，在线考试或者测试被广泛应用于职业选拔和教学中[2-4]，相比于传统的测试，在线测试具有低成本、高效率等优势。结合在线测试，研究实现一个软件测试案例线测评系统，将有效的提高软件测试实践教学，降低教学成本。

## 国内外研究现状

软件测试作为软件开发进程的关键一个环节，其目标就是验证软件的正确性、完成性、安全性和品质，它采用自动化测试或人工测试的方式，对开发完成的软件件是否满足需求要求进行验证。软件测试的成败与否，决定了软件的最终成败，所以软件测试至关重要。

在线判题系统Online Judge起源于ACM-ICPC国际大学生程序设计竞赛和OP信息学奥林匹克竞赛，使用Online Judge 来进行判题和排名。现在各地高校纷纷建立自己的OJ系统，用于程序设计的训练和比赛。现在比较有名气的OJ如：北京大学（POJ）、西班牙Valladolid大学（UVA）等，该类系统能够编译并执行代码，并用预先准备好的数据去验证代码的正确性，及执行过程中的参数（内存占用等），并返回验证结果[2]。但尚未有针对软件测试实践的在线测评系统。

从软件测试国内使用现状上来看，按照软件开发过程中测试的流程，主要的测试工具用来进行单元测试、集成测试、回归测试和性能测试。对于单元测试和集成测试而言，目前市场上有许多的测试工具，可以用来搭建测试环境，完成相应的测试工作，如java的Junit和TestNG，C++的Google Test等。回归测试热门的工具如TestingWhiz 和Sahi等都是不错的自动化测试工具。主流的性能测试工具如LoadRunner和Jmeter，通过模拟实际用户的操作行为进行性能监控。要使用使用这些测试工具完成测试不仅需要耗费安装机器资源，也需要去花费成本去学习如何去搭建测试环境，如何去运用这些测试工具。上述这些软件或工具虽然能满足软件测试的需求，但用在教学实验平台方面会受到机房单台机器的硬件资源限制，不能模拟复杂的测试环境。从软件测试发展的趋势上，近些年软件测试从单机模式向云测试、众包测试模式转换[5-6]。云测试是一种基于云计算技术的新型测试方案，它将软件测试或者自动化软件测试工具作为一种服务提供给使用者。云测试平台可以提供多种浏览器作为用户交互的客户端，通过网络接收用户编写的测试脚本，在测试环境中运行用户的测试脚本进行测试，最后再通过网络将测试结果传输给用户。

## 研究目标和内容

本论文的主要研究目标：软件测试教学环节中，测试用例的设计是学生对软件测试技术（白盒法/黑盒法）掌握程度的直观反映。长期以来，对测试用例的执行和评判都是由教师和学生根据案例特点人工评判，容易出现漏判、误判、逻辑覆盖率低的情况。所以本文主要研究目标为，开发一套软件测试在线测评系统，该系统以OJ系统为基础，并结合软件测试工具，借鉴在线评测系统的工作原理，采用JAVA WEB开发技术，以实现软件测试的在线评判。该系统要求设计一种计算机所能接受的测试用例格式，制定评分标准，具备常规Web系统的用户管理、系统配置等基本功能；能够自动解析并执行学生上传的测试用例（遵循），计算测试数据的逻辑覆盖率，以此作为测试用例的评价标准[7]，实时统计并给出评判分数，从而实现计算机自动判别测试用例的功能。

本论文的主要研究内容：在线测评系统的工作原理[8-11]，测评机的测评机制及系统的高可用性。具体来说，包含三个部分：测评机如何与测评系统通信，测评机的实现和系统的扩展如何保证。测评机和测评系统的通信可以使用消息总线Apache ActiveMQ。测评机是系统的核心，此部分实现对测试用例的评判，测评机需要结合软件测试工具来实现，主要实现功能包含测试程序的执行，覆盖率的统计及测试测试结果的处理。系统的扩展性包含两个方面，一方面是对测试语言支持的扩展，另一方面是对测评机的扩展，当用户量过多时可以增加测评机的数量来减轻负载。



图 1- 1

图表 1-1在线测评系统工作流程图表格

## 论文的组织结构

本论文共分为六大章节，其组织与主要内容如下：

第一章 这一章主要介绍了一下本论文的研究的主要目标、内容、现状和组织结构。

第二章 这一章主要介绍了本系统实现过程中应用到的的主要技术和框架，其中Web开发的框架采用了比较流行的Spring+SpingMVC+Mybatis。Web端和测评机的通信采用消息总线 Apache ActiveMQ。测试覆盖率统计用到了java的jacoco及Python的nose-coverage。测试代码的执行使用自动化工具Ant。测试用例代码生成使用jsoup。

第三章 本章主要主要分析系统，明确系统需求，设计系统的架构包括逻辑架构涉及系统功能，物理架构设计部署，运行架构涉及运行时概念，开发架构涉及程序包，系统的功能结构。

第四章 本章主要是系统的详细设计，各功能模块的设计，主要模块包括：系统的管理模块、单元测试模块、系统维护模块。数据库设计，包含数据库E-R图及数据库表字典。操作界面设计、程序模块设计、安全性设计。

第五章 本章主要涉及系统的实现与测试，其中系统实现包含系统的整体实现、关键技术实现，包含消息的通信等、系统测试，包含测试环境和测试方案准备、测试数据与过程和测试结果与分析。

第六章 本章主要阐明本系统存在的一些不足，及改进设想，同事也对本系统做一个小结。

# 第二章 技术支持

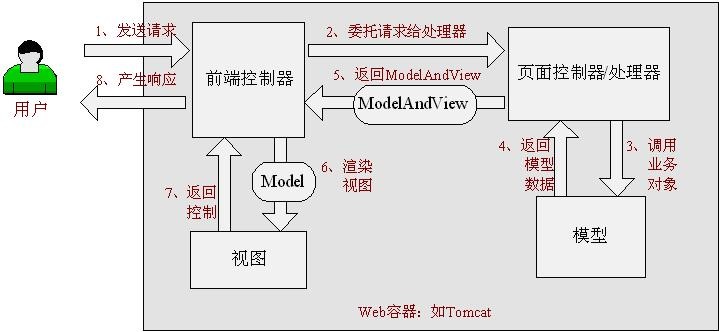
## 2.1 开发平台及框架

本系统以eclipse MARS.2作为开发工具，以Spring+SpringMVC+Mybatis 作为Web开发框架[12]。

Spring 是一个轻量级的Java开发框架，它的最初目的主要是为了简化Java EE的企业级应用开发，相比于过去EJB 时代重量级的企业应用开发而言，Spring 框架更加简单灵活。面向切面编程（AOP）和控制反转（IOC）是Spring框架的核心功能。

SpringMVC 作为一个请求驱动的Web框架，其处理控制器的实现策略，与其他的请求驱动的Web 框架在总体思路上是相似的，就是通过引入Front Controller和PageController的概念来分离流程控制逻辑与具体的Web请求处理逻辑。DispatcherServlet就是SpringMVC框架中的Front Controller，它负责接收并处理所有的Web请求，只不过针对具体的处理逻辑，它会委托给它的下一级控制器去实现即Controller。

MyBatis 是一个持久化层框架，它隐藏了繁琐的配置和JDBC的代码，可以通过xml 或注解来实现数据与Mapper接口之间的映射。



图表 2-1 Spring Web MVC 处理请求流程

## 2.2 消息通信ActiveMQ

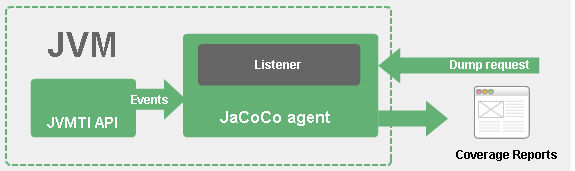
ActiveMQ 是一种支持JMS规范的中间件。它提供了一种松散耦合的灵活的应用程序集成机制，程序直接彼此之间不需要直接通信，二通过消息中间件来作为中介者进行通信。本系统使用Spring集成ActiveMQ，Spring JMS异步发收消息。

## 2.3 覆盖率统计

### 2.3.1 Java

JaCoCo是一款针对java的单元测试覆盖率统计工具，它的使用方式比较多样，既可以通过Eclipse插件的形式使用，也可以通过嵌入到maven或ant中使用，目前JaCoCo支持不同标准的覆盖率统计,其中有常用的分支覆盖，还有一些像指令级别的覆盖、行覆盖、方法类覆盖。

JaCOCO 作为Java代理运行，它负责在运行测试是对字节码进行检测。在执行测试时，JaCOCO会渗透到每一行指令并且显示那些行被执行。为了收集覆盖率数据，JaCOCO在运行中使用ASM进行代码检测，在JVM工具接口中接收事件。



本系统对于Java语言的测试用例覆盖率统计使用JaCOCO。使用JaCoCo统计Java单元测试覆盖率结果如下图2-2：

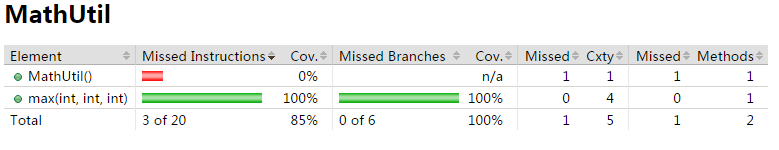


图 2- 2

### 2.3.2 Python

Nose 作为一个知名的Python单元测试框架[14]，优于unittest 。nose 可以自动识别继承于unittest.TestCase的单元测试，并且也可以测试非继承于unittest.TestCase。nose为编写单元测试提供了全面的API。Nose也可以通过内置插件实现代码覆盖、文档测试、错误查找等等，同时也可以自定义开发插件来满足测试需求。本系统对于Python语言的测试用例覆盖率统计使用nose。使用nose的cover插件统计Python单元测试覆盖率结果如下图2-3：

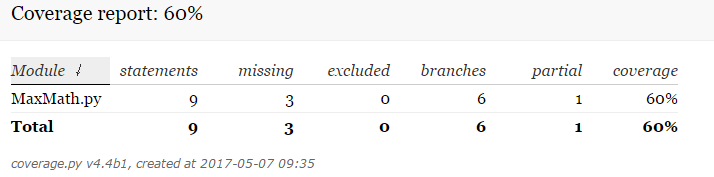


图 2- 2

图表2-3 nose-coverage覆盖率统计结果

## 2.4 其他技术

### 2.4.1 Ant

Ant作为一个跨平台的构件工具，它能够完成工程的自动化构建和部署等功能。Ant的构件文件的编写是基于XML的，默认为build.xml。构件文件有两个主要元素，project 元素是build构件文件的基本元素；target元素是Ant的基本执行单元。在本系统中Java测试的编译、执行，JaCOCo的集成通过Ant来完成。

### 2.4.2 Jsoup:

jsoup是一个Java 版本的Html解析工具包[15]。可以从连接、HTML文件和字符串中解析HTML。使用DOM遍历或者CSS选择器查找和抽取数据。可以操作HTML元素、属性和文本。本系统中对于HTML形式覆盖率结果的解析由Jsoup来完成，使用Jsoup将覆盖率结果以List对象的形式存储。

### 2.4.3 Beetl

Beetle作为新一代Java模板引擎的典范[16]， 与其他Java模板引擎，它使用更加的方便，而且功能也比较多，在性能上更是由于其他模板引擎，学习起来也比较简单，它也可以和各种Web框架进行整合，在扩展方面Beetle支持自定义方法，同时也支持自定义占位符。

本系统测中试用例的代码生成由Beetle来完成，根据设计的用例、题目案例的信息作为输入，参照对应语言的模板生成相应的测试用例代码。

## 2.5本章小结

这一章节主要介绍了系统开发过程中所应用到的主要技术，如Web端开发过程中用到的Spring的核心技术IOC和AOP，覆盖率统计统计时用到nose的cover插件等等。这些技术无论是涉及多还是少，都是支撑整个系统不可或缺的一部分。

# 第三章 系统需求分析和结构设计

## 3.1 需求分析

### 系统目标

软件测试教学环节中，测试用例的设计是学生对软件测试技术（白盒法/黑盒法）掌握程度的直观反映。长期以来，对测试用例的执行和评判都是由教师和学生根据案例特点人工评判，容易出现漏判、误判、逻辑覆盖率低的情况。如果有一款能借助计算机实现测试用例自动评判的软件系统，将能显著改善教学效果。鉴于此，本系统将借鉴在线测评系统的工作原理，实现一款软件测试用例在线评判系统。

### 系统需求

系统功能性需求

单元测试：

学生可以根据条件筛选测试案例，提交测试用例，查看提交记录，查看所有提交记录，修改个人资料等。获取学生提交的测试用例，生成测试用例代码。执行测试生成测试覆盖率报告，解析测试报告，持久化测试结果信息。反馈测试消息给Web端。

系统管理

用户信息管理，涉及到用户的登录及注册，用户的增删改查，用户权限的管理。系统本身的一些属性的管理，如系统的版本，是否允许注册，支持语言等等。

系统维护

本部分功能涉及测试相关的元素的维护，包含测试案例、用户提交的添加、删除、修改等操作。还有对不同语言测试工具的管理，对测评机的维护等

系统非功能性需求

高可用性

在系统运行过程中难免不会出现一些情况，导致测评机死机，为了保持系统的正常运行，系统提供多个测评机，当一个测评机停止工作后不至于影响系统的正常运行。

可扩展性

对于一个以在线测评系统不能只支持一种语言，还应当可以扩展其他语言。扩展的过程中应当尽量少的修改源程序代码，以模块的形式添加新语言的支持。

## 3.2 系统架构设计

### 逻辑架构

从逻辑结构上来看分为四大部分：

1. 用户展现层

本层主要为用户提供操作界面，用户目前分两种：一种为学生，学生视图提供了学生操作的界面，包括案例的选择页面，测试用例的提交页面，提交的查看页面等等。二为教师，教师视图提供了教师操作的界面（即管理员），包括学生所拥有操作页面，测试案例管理页面，用户的管理页面，提交用例管理页面，系统的基本设置页面和测试工具配置页面等等。

1. 服务层

本层次主要为上层提供服务，响应用户提交的各种请求。服务主要分三大模块：系统管理模块，本模块主要包含用户登录注册工能、用户的管理（添加、删除、修改、查询）和用户权限管理等等；系统维护模块，本模块主要包含测试案例的管理、用例提交的管理、测试工具管理和系统设置等等；单元测试模块，本模块主要包含测试用例提交请求处理，测试结果反馈等。

1. 数据层

数据层提供对系统相关数据的持久化存储和访问，主要包含三个方面的数据：测试案例数据，存储教师提交的测试案例，包含了测试案例相关的属性；用例提交，存储学生提交的测试用例信息，包含了测试案例的信息、用例、测评结果等相关信息。用户数据，存储学生和教师用户相关信息。

1. 测评机

测评机为执行学生提交测试用例的核心，当测评机的消息接受收到web端传来的测试任务创建消息后，创建测试任务并执行，为避免同步线程安全问题，此处采用synchronized来解决，保证一次只执行一个测评任务。待任务执行完毕后，使用解析工具将执行结果解析成java的List对象并将结果持久化到用例提交数据库中，并发送测试任务执行完毕的消息给Web端。同时对提交的测试用例做筛选排序处理，将优秀的测试用例持久化到数据库。



图 3- 1系统逻辑架构图

### 物理架构

本系统分两部分以一为：Web，此部分提供了用户操作界面的构建以及用户提交请求的处理，是整个项目的前台；二为engine，此是测评系统的核心部分，负责完成测评任务的处理及反馈工作。部署上二者相互独立，通过ActiveMQ来进行通信，二者公共完成测评工作。物理架构图如下图3-2。



图标 3- 2系统物理架构图

### 运行架构

学生提交测试用例，由服务层完成完成用例提交数据的部分数据持久化，同时发送异步消息给测评机，测评机收到消息创建测试任务，此过程为同步线程安全的，完成测试任务后持久用例测试结果数据到用例提交数据库。同时发送消息给Web，向用户展示测评结果。运行架构图如下图3-3



图标 3- 3系统运行架构图

### 开发架构

本系统开发的基础框架、通信和主要技术如下图3-4、图3-5，本系统的主要静态包组织结构如下表3-1；开发视图从技术及组织结构的角度指导系统的开发。



图标 3- 4 Web端相关的框架与技术



图标 3- 5测评机相关技术

表格 3- 1系统包结构

|  |  |  |
| --- | --- | --- |
| 项目名 | 包名 | 说明 |
| caseonline.engine |  | 测评机 |
|  | caseonline.judger.engine.application | 入口 |
|  | caseonline.judger.engine.core | 测评核心实现 |
|  | caseonline.judger.engine.exception | 异常 |
|  | caseonline.judger.engine.mapper | 数据接口 |
|  | caseonline.judger.engine.messenger | 消息处理 |
|  | caseonline.judger.engine.model | 数据实体 |
|  | caseonline.judger.engine.util | 工具 |
| caseonline.web |  |  |
|  | caseonline.judger.web.aspect | 切面 |
|  | caseonline.judger.web.controller | 控制 |
|  | caseonline.judger.web.exception | 异常 |
|  | caseonline.judger.web.mapper | 数据接口 |
|  | caseonline.judger.web.messenger | 消息处理 |
|  | caseonline.judger.web.model | 数据接口 |
|  | caseonline.judger.web.service | 数据服务接口 |
|  | caseonline.judger.web.util | 工具 |

### 功能结构

图标 3- 6系统功能结构

## 3.5 本章小结

本章主要介绍了了系统的需求和设计。系统的需求的分析：这里介绍了系统的设计的目标，及系统的需求。系统的架构设计：这里使用架构计的五视图法，从逻辑视图、物理视图、运行视图、开发视图的角度分析设计了系统的架构。总的来说这一部分是从需求和设计角度来为系统的开发提供全方位的指导。

# 第四章 系统详细设计

## 4.1 系统功能模块设计

### 4.1.1 系统管理模块

模块结构



图标 4- 1系统管理模块结构

模块类图



图标 4- 2系统管理模块类图

AccountsController控制器：用来处理用户的登录、注册以及修改个人资料等请求。

UserManagerController控制器：用来处理用户管理的请求。

AdminController控制器：用来处理系统设置的请求。

UserService：提供用户相关的服务。

GeneralService：提供系统一般设置的服务。

模块功能

主要实现用户信息管理和系统的设置。

用户管理

1. 用户的注册、登录

用户输入用户验证信息，通过验证通过则注册或登录成功，未通过则显示错误信息，用户重新操作。其流程图如下图 4-3 ：

流程图：



图标 4- 3用户登录、注册流程图

用户权限管理

根据角色的不同将请求提交到不同的控制器，有控制器显示不同的可操作界面。

系统设置

是否允许注册，界面风格，支持语言管理等

### 4.1.2 系统维护模块

模块结构



图标 4- 4系统维护模块结构

模块类图



图标 4- 5系统维护模块类图

SubmissionManagerController控制器：测试用例提交管理请求处理器。

TestcaseManagerController控制器：测试案例管理请求处理器。

SubmissionService：提供测试用例提交管理服务。

TestcaseService：提供测试案例管理服务。

模块功能

本模块主要用于测试案例、测试用例提交以及测试工具的管理等工作。

### 4.1.3 单元测试模块

模块结构



图标 4- 6单元测试模块结构图

类图



图标 4- 7单元测试模块类图

TestcaseController控制器：接收用户的测试用例提交请求。

SubmissionService：提供提交信息服务。

TestcaseService：提供测试案例信息服务。

Dispatcher: 测评调度器，完成测评任务的创建与执行。

ApplicationDispatcher:消息处理器，处理测评过程中消息,如测试相关文件创建完成等。

流程图



图标 4- 8单元测试模块流程图

模块功能

单元测试模块主要负责测评的工作，从用户选择测试案例，提交测试用例，交由测评机进行测试、覆盖率统计、及测试结果处理（优秀测试用例的筛选，结果消息的反馈），用户查询所有测试用例提交信息等等。

## 4.2 用户界面设计

### 4.2.1设计原则

一个成功的应用系统，只有功能的齐全不能说是一个完善的系统。系统最终是给用户去使用的，所以一个好的系统还应当有一个友好的操作界面。在设计是应当遵循一定的界面设计原则：

用户原则：界面设计首先要确定用户类型，针对用户的特点来设计。例如一个节日营销网站，页面应当是比较华丽的，而一个信息管理系统，应当简洁朴素的。

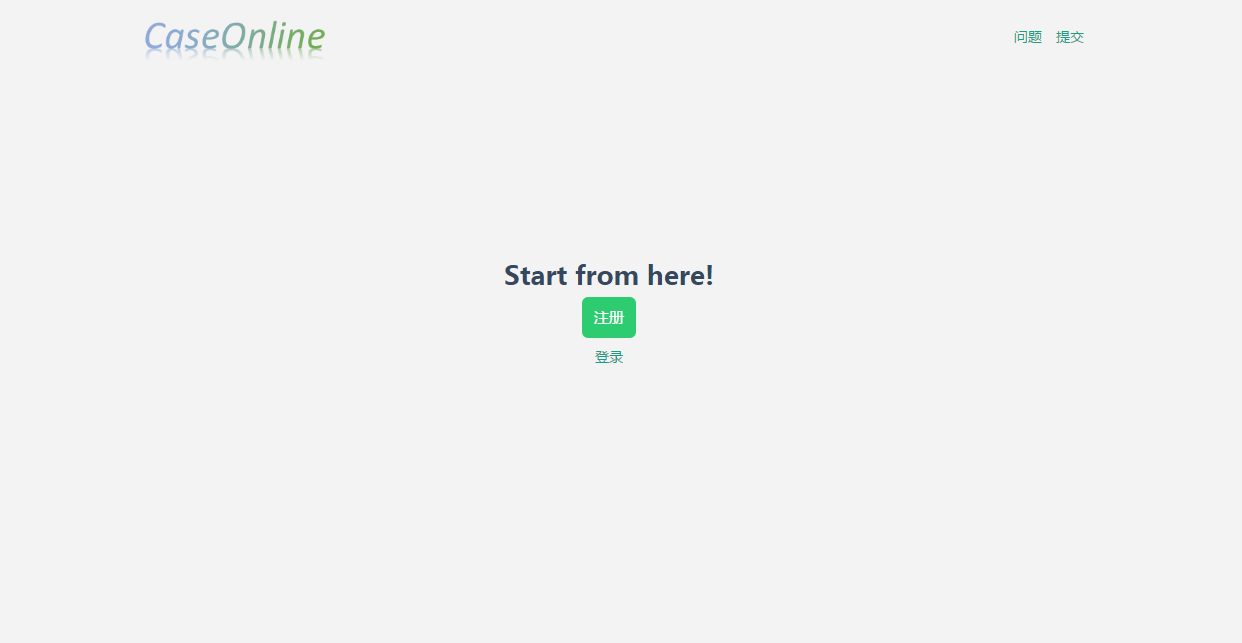
简单原则：尽量减少用户的记忆负担，一个完整的任务，仅仅需要两三步操作就可以完成，而不需要用户去记忆繁琐的操作步骤。

直接操作：在任何一个界面，用户都能以最少的点击到达另一个想要进入的界面。或许添加连接可以到达这一目的，但过多的连接、按钮会影响用户对界面功能把握， 所以要把握好这个度。

### 4.2.2 界面设计

##### 主页面

主页面为系统的首页，此页面用户操作的起点，在此可以登录、注册、查看问题以及查看提交。



图标 4- 10 主界面

##### 登录及注册界面

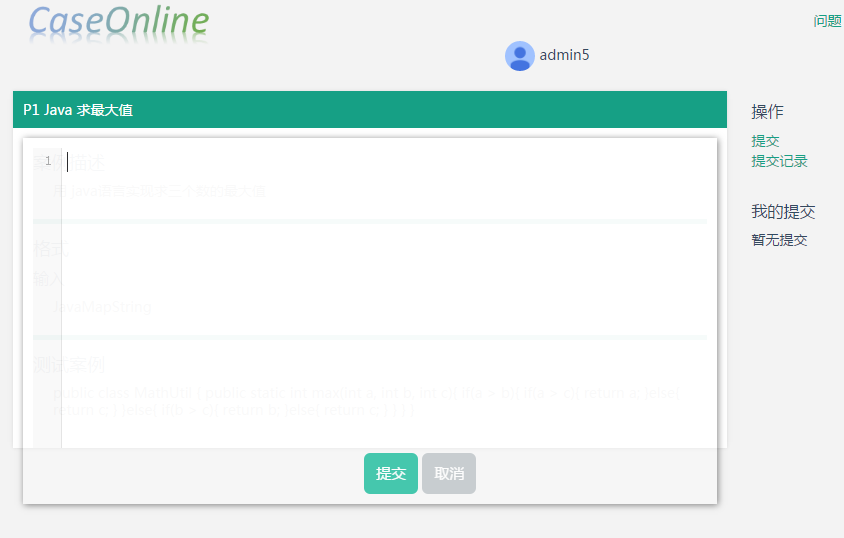
|  |  |
| --- | --- |
| 图标 4- 11 登录界面 | 图标 4- 12 注册界面 |

##### 测试案例界面



图标 4- 13测试案例界面

##### 提交测试用例界面



图标 4- 14提交测试用例界面

##### 系统管理界面



图标 4- 15系统管理界面

## 数据库设计

### 4.3.1 数据库总体结构



图标 4- 9系统数据库整体表结构

本系统总共涉及14张表，每张表的作用如下表所示：

表格 4- 1数据库表

|  |  |  |
| --- | --- | --- |
| Name | Code | Comment |
| 测试案例表 | | |
| coj\_testcase | coj\_testcase | 测试案例表 |
| coj\_testcase\_category | coj\_testcase\_category | 测试案例类型 |
| coj\_testcase\_rela\_cate | coj\_testcase\_rela\_cate | 测试案例和类型关系表 |
| 用户表 | | |
| coj\_user\_role | coj\_user\_role | 用户角色类型分类 |
| coj\_user | coj\_user | 记录用户信息 |
| 提交用例表 | | |
| coj\_submission | coj\_submission | 提交测试案例 |
| coj\_excellence\_usecase | coj\_excellence\_usecase | 优秀的测试用例设计 |
| 数据字典 | | |
| coj\_usecase\_format | coj\_usecase\_format | 向用户展示测试用例编写格式 |
| coj\_language | coj\_language | 所支持的语言，及与语言相关联的信息 |
| coj\_judge\_result | coj\_judge\_result | 测评结果 |
| coj\_option | coj\_option | 系统属性 |

## 4.4 程序模块设计

### 4.4.1 测评程序模块

程序模块类图



图标 4- 16测评程序模块核心类图

测评程序模块在设计上，将整个测评过程抽象为三步：测评前的准备，测评，测评后的结果分析。定义统一接口ICoverage其中定义三个函数，来完成对应工作。测试用例生成工具类UsercaseGenerator、测试前期准备工作工具类Preprocessor 、覆盖率结果解析工具HtmlParse。要实现某一语言的测试用例覆盖率统计扩展，需要实现ICoverage 接口，实现接口中的测评的三个方法。

## 4.5 安全性设计

### 4.5.1 用户权限的划分

目前系统提供了三种用户类型，管理员、教师、学生。管理员拥有最高权限，可以管理及维护测试案例，测试用例提交等等；教师拥有二级权限，可以管理维护自己提交的测试案例查看优秀的测试用例等；学生拥有最低权限，查看测试案例、测试用例提交等。目前的设计方案为根据用户角色类型的不同，其可见操作界面不同，处理请求的控制器Controller映射不同，如管理员请求以/admin为跟路径。

### 4.5.2数据的一致性

本系统数据库设计中并未使用外键，在相关联的数据表进行添加和删除操作时，往往会出现不一致到处出错情况。所以为了避免这种情况，在进行删除操作时，同时考虑将其相关联的表进行删除操作。

## 4.5 本章小结

这一章中详细介绍了系统的设计，从各个方面展示了系统的实现的方案。

功能模块设计展示了系统的模块划分以、各模块的功能和模块的出入参数和处理流程；数据库设计展示了系统的数据库基础，确定了系统的数据存储结构；用户界面设计展示了系统的门面，确定系统界面设计的简介明了的整体风格。程序模块设计主要涉及到一些程序代码的设计，以提高代码的可重用性；安全性设计主要考虑两个方面，一是用户权限，保障系统数据的安全性，避免一般用户对系统造成破坏。二是数据库，由与相关表之间没有设计外键，没有自动级联删除，所以需要通过代码来实现级联删除，保证数据的一致性，系统数据获取异常。

# 第五章 系统实现和测试

## 5.1 系统整体实现

系统整体上采用B/S结构实现，使用Spring+SpringMVC+Mybatis 开发框架来实现，MySQL作为数据库，Tomcat8.0 作为服务器，MQ作为消息服务。使用bootstrap作为页面开发框架。整体实现结构如下图5-1。



图 5- 1系统整体实现

## 5.2 关键技术的实现

### 5.2.1 代码生成

测试用例提交时，为了学生的方便实际上提交的是测试脚本而不是测试代码，所以在执行测试时，需要根据脚本数据生成测试代码，即以测试用例脚本数据以及对应测试案例作为输入参数，以测试用例代码文件为输出结果。通过对比目前主流模板引擎Freemarker和Velocity，Beetle性能以及操作上都要优于它们，所以代码生成采用Beetle模板引擎，具体实现方式如下：

第一步：编写模板文件（\*.template）。

|  |
| --- |
| <%  for(head in bean.heads){  %>  import ${head}  <%}%>  /\*\*  \*  \* 测试类${bean.className}  \* @author ${bean.author}  \* ${bean.createTime}  \*  \*/  。。。。。。。。。。。。 |

第二步：加载模板文件目录，获取模板。

|  |
| --- |
| GroupTemplate gt = new GroupTemplate(resourceLoader, cfg);  Template t = gt.getTemplate(FilePath); |

第四步：封装Bean。

将输入参数数据封装成Bean，作为下一步的绑定参数。

第四步：绑定Bean执行。

|  |
| --- |
| t.binding("bean", bean);  String code = t.render(); |

### 5.2.2消息通信

为了提高系统的高可用性，防止测评时出现问题导致系统死机的情况，本系统将测评机作为一个独立的整体。测评机与其他部分的通信通过MQ消息服务器来实现，具体实现过程为：学生提交测试用例后，首先将测试用例先关属性信息存入数据库中，然后向测评机发送提交创建消息，测评机收到消息后，首先根据提交ID获取提交用例信息，然后创建测评任务（此部分为了避免线程同步问题，采用synchronized来保障一次只有一个测评任务在执行），执行测试用例的测评，完成测评后，将测评日志信息存入、结果存入数据库，同时将测评进度、结果信息打包成消息反馈给Web端，向用户展示测评结果。先关实现点如下图



图 5- 2测评机消息通信实现

### 5.2.3 测评机

测评机是在线测评系统的核心，其主要功能就是接收测评请求、创建测评任务（执行前准备、执行测评任务、覆盖率解析）、结果处理及反馈。接收测评请求的实现是通过MQ消息服务器来实现；测评任务的创建，测评请求消息体包含由提交ID，创建测评任务时可以根据提交ID来获取提交的测试用例以及对应的测试案例信息，根据获取的信息完成相应的文件创建等前期准备工作，执行测评任务通过Java的Running.exec()来执行相应语言的测评指令，执行完成之后会生成对应的覆盖率报告（针对目前支持的语言Java和Python，来说报告是html形式），通过html解析工具（Jsoup）完成覆盖率报告的解析；测评完成后将测评的过程日志、结果存入数据库，根据测评的覆盖率选出有优秀的测试用例存入数据库中，

在测评的过程中每完成一个阶段都会给Web端发送相应的消息，告知测评的进度。由于测评机是独立的，所以可以同时部署多个测评机，以此来提高系统的效率，减轻测评机的负载。其实现流程如下图5-3：



图 5- 3 测评机执行

## 5.3 系统测试

### 测试环境与方案

系统测试环境：

系统：window7

Java：jdk1.8

Python：3.6

数据库：MySQL

服务器：tomcat 8.0

Ant：apache-ant-1.9.9

Apache ActiveMQ: 5.14.3

测试方案：

系统测试大致分三块进行测试：

系统的核心功能测试

描述：测试用例的测评，主要测试系统的测评功能，能否正确的测评结果。

流程：

第一步：选择测试案例，根据类型或名称筛选测试案例。

第二步：编写测试用例并提交，查看测评结果及测评日志。

第三步：查看提交列表。

系统相关实体的管理功能测试

描述：测试数据库实体的添加、修改、删除，查询。

流程：

第一步：输入操作实体参数。

第二步：执行操作。

第三步：结果对照。

数据服务接口测试

描述：测试数据层的数据操作接口。

流程：

第一步：测试参数环境配置。

第二步：编写测试类。

第三步：执行分析执行结果。

错误处理测试

描述：测试程序对错误输入参数，错误操作的处理。

流程：

第一步：分析可能出错的操作。

第二步：执行错误操作。

第三步：查看错误处理是否与设想的一致。

### 系统测试数据与过程

数据

通过sql脚本完成数据库的初始化，涉及的数据表如下：

表格5- 1测试需要初始化的表

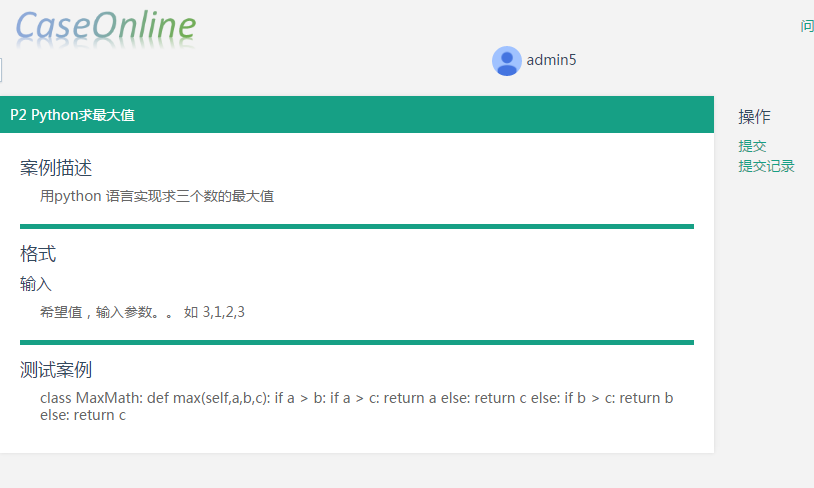
|  |  |
| --- | --- |
| 名称 | 说明 |
| coj\_user\_role | 角色 |
| coj\_user | 用户 |
| coj\_usecase\_format | 用例格式 |
| coj\_testcase\_category | 案例类型 |
| coj\_option | 系统属性 |
| coj\_language | 语言 |
| coj\_judge\_result | 测评结果 |

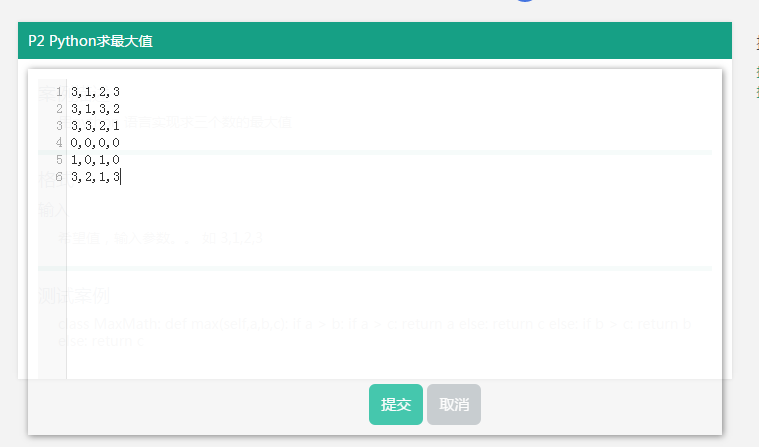
测试使用测试数据库testcaseonline，使用设计好的脚本完成数据初始化。

受篇幅限制，本文中仅展示部分测试过程。

功能测试

提交测试用例测试：







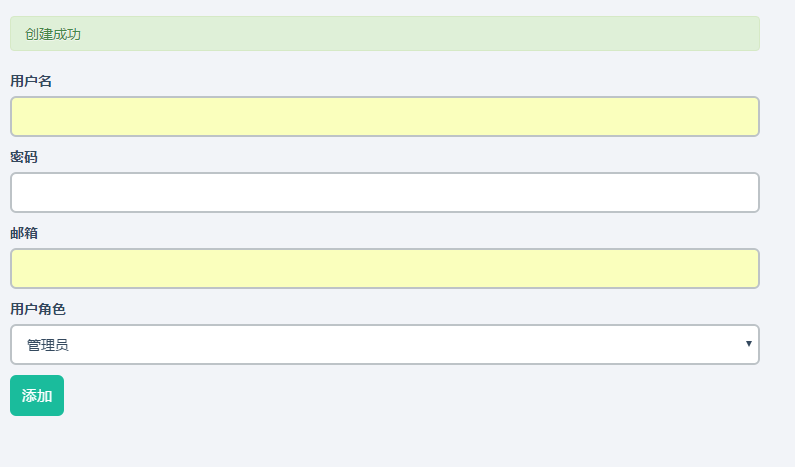
系统元素信息管理

用户：

添加：



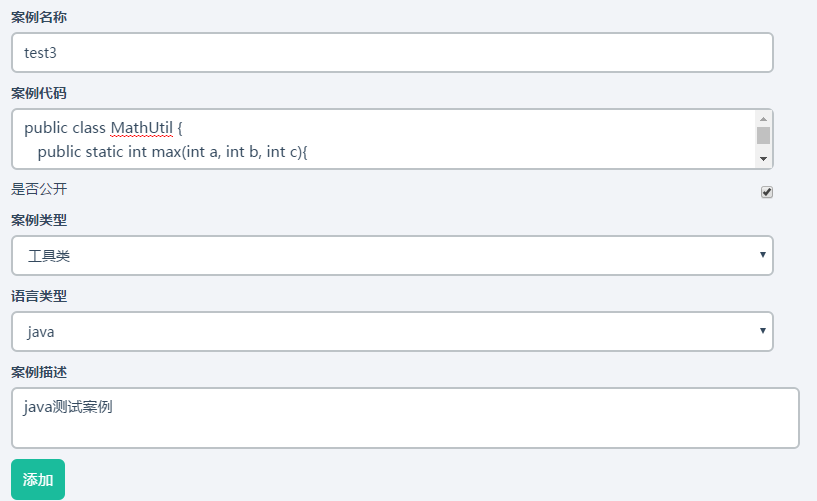
添加成功



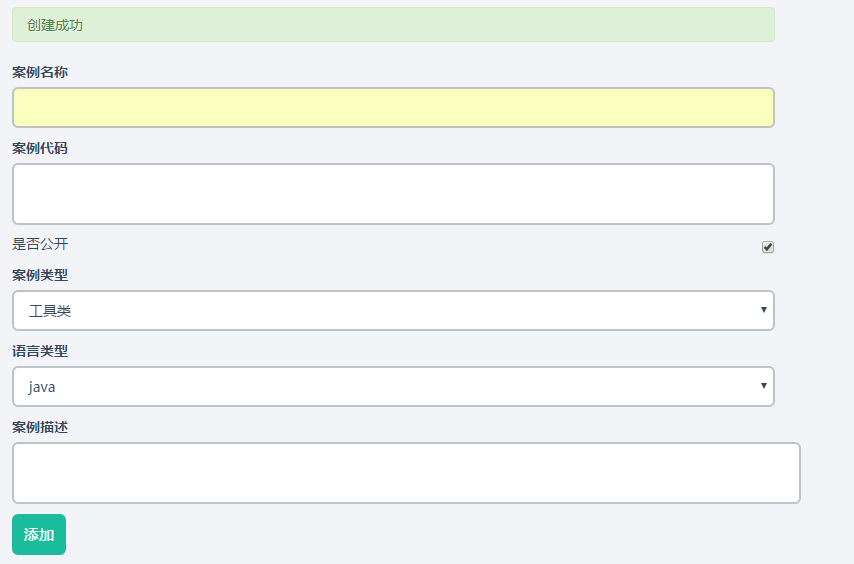


测试案例：

添加：



添加成功





接口测试：



用户角色接口：

|  |  |
| --- | --- |
|  |  |

测试案例接口：

|  |  |
| --- | --- |
|  |  |

### 系统测试结果与分析

功能测试：

|  |  |
| --- | --- |
| 测试功能点 | 测试结果 |
| 测试案例的查询筛选 | 正常 |
| 测试用例的提交获取测评结果 | 正常 |
| 测试用例提交的查询筛选 | 正常 |
| 用户的管理（添加、删除、修改、查询） | 正常 |
| 测试案例的管理（添加、删除、修改、查询） | 正常 |
| 测试用例提交（查询、删除） | 正常 |
| 系统设置 | 正常 |
| 用户操作限制 | 正常 |
| 验证（测试案例代码、用户注册信息等） | 正常 |
| 提交用例排名 | 正常 |
| 用户的登录注册 | 正常 |

数据接口测试：

|  |  |  |
| --- | --- | --- |
| 接口 | 说明 | 测试结果 |
| ExcellenceUsecaseMapper | 优秀测试用例提交接口 | 正常 |
| JudgeResultMapper | 测评结果接口 | 正常 |
| LanguageMapper | 支持语言接口 | 正常 |
| OptionMapper | 系统属性接口 | 正常 |
| SubmissionMapper | 测试用例提交接口 | 正常 |
| TestcaseCategoryMapper | 测试案例类型接口 | 正常 |
| TestcaseMapper | 测试案例接口 | 正常 |
| TestcaseRelaCateMapper | 测试案例类型关系接口 | 正常 |
| UsecaseFormatMapper | 测试用例格式接口 | 正常 |
| UserMapper | 用户接口 | 正常 |
| UserRoleMapper | 用户角色接口 | 正常 |

就当前测试，系统基本上可以正常工作，实现所设计的功能。但是测试上难免有所疏漏，或许系统还有一些没有测试出的Bug，这就有待以后使用过程中发现了。

## 5.3 本章小结

在这一章中主要讲解了系统的整体实现以及系统中关键技术点的实现方式，包括如何根据测试用例数据代码的生成、Web是如何与测评机进行消息通信的以及测评机实现细节。然后介绍了系统的测试，包含了系统测试环境与测试方案，测试需要的数据环境，简要的测试过程展示，最后是对测试过程中的结果进行统计与分析。

# 第六章 总结和展望

## 6.1 总结

本文主要根据在线判题系统的原理，结合软件测试工具、Web开发技术，开发出一款能够进行软件测试用例的在线测评系统。在线判题系统基本原理就是通过Web页面选择试题提交，具体对题提交答案的判定由判题引擎来完成。所以本文主要研究内容是如何实现对测试用例的评判。

目前确定的评判标准是单元测试的覆盖率，单元测试覆盖率统计工具有java语言的JaCOCO，python 语言的nose-cover等等，这些工具以源码及单元测试类作为输入参数，以覆盖率报告作为输出结果。所以测评机需要做的就是：根据测试任务请求创建文件，然后使用指令执行覆盖率统计工具，得出覆盖率结果。同时考虑到测评机对支持语言的扩展性，将测评的工作抽象封装成对应的三步，文件创建、执行、结果解析。添加语言支持，需要继承测评接口实现这三个函数。

## 6.2 展望

目前系统可以完成单元测试测试用例的覆盖率的测评，受覆盖率统计工具的限制，支持的覆盖率标准较少，如Python的nose-cover 仅仅支持分支覆盖率的统计，所以可以考虑自己设计实现一种单元测试覆盖率统计工具[17]。另一方面在支持语言扩展上，目前需要手动去配置新语言单元测试覆盖率统计环境，且需要改动测评机代码，一种理想的扩展方案是，将新语言的测评实现进行封装做成一个插件，直接在系统管理页面上配置实现扩展，或者是将测评实现打包，通过页面添加扩展时，向测评机发送下载包的指令，将新语言的测评包下载到服务器，并关联到测评机，以实现对新语言的支持。另外在前端功能上可以添加班级管理的功能，教师可以按班级布置测试任务，班级间的结果数据对比等等。

# 致谢

本系统设计过程中，得到了许多的帮助。很是感谢张剑波老师的指导，从系统设计到论文的编写，张老师都给出了不少的建议。另外张老师严格的教学方式和耐心的指导以及做事的严谨性对我的提高有不少帮助，。还要感谢研究生学长在系统设计上的帮助。

# 参考文献

[1] 王秀. 谈软件工程中软件测试的重要性及方法[J]. 天津成人高等学校联合学报,2004,(02):76-78+81.

[2] 张盈. 在线考试评判系统的设计与实现[D].四川大学,2005.

[3] 卢依宁. 基于静态分析的Java单元测试教学反馈系统[D].南京大学,2016.

[4] 王迎. 自动化测试服务平台的设计与实现[D].北京交通大学,2015.

[5] 荀江萍. 浅谈基于云计算的软件测试[J]. 无线互联科技,2014,(05):45.

[6] 熊娇. 新型软件测试技术研究与实现[D].电子科技大学,2008.

[7] 焦安涛. 测试用例综合评价模型的研究[D].昆明理工大学,2009.

[8] 黄晓华,沈健,常晋义,张明新. 基于Online Judge与HTML批注技术的实验教学平台设计[J]. 计算机与现代化,2014,(11):117-121.

[9] 王腾,姚丹霖.Online Judge系统的设计开发[J]. 计算机应用与软件,2006,(12):129-130+137.

[10] 何迎生,罗强. Online Judge评判内核的设计与实现[J]. 吉首大学学报(自然科学版),2010,(06):37-39.

[11] Huanda Lu,Liangzhong Fan,Zhenyu Shu Laboratory of Information and Optimization Technologies Ningbo Institute of Technology, Zhejiang University Ningbo, Zhejiang Province, ChinaXin Yu,Qiyu Liu,Xinjie Yu Laboratory of Information and Optimization Technologies Ningbo Institute of Technology, Zhejiang University Ningbo, Zhejiang Province, China. Online Judge System under a Cloud Computing Environment[A]. Hubei University of Technology, China.Proceedings of 2010 Third International Conference on Education Technology and Training(Volume 2)[C].Hubei University of Technology, China:,2010:3.

[12] SSM框架笔记http://www.kancloud.cn/digest/lunaticssm/122056[OL]

[13] JaCoCo：分析单元测试覆盖率的利器https://www.ibm.com/developerworks/cn/java/j-lo-jacoco/[OL]

[14] nose introduction http://pythontesting.net/framework/nose/nose-introduction/[OL]

[15] Jsoup中文文档 http://www.open-open.com/jsoup/parsing-a-document.htm[OL]

[16] Beetl中文文档 http://ibeetl.com/guide/#beetl[OL]

[17] 魏光新,苏丽. 逻辑覆盖测试工具的设计与实现[J]. 计算机工程与应用,2000,(05):106-109.