

东北林业大学 2021 届本科毕业设计

开 题 报 告

设计题目：_____测试事务管理平台的设计与实现_____

学 生：_____于丽蔷_____

指导教师：_____李莉 副教授_____

专业（年级、班级）：_____软件工程 2017 级 4 班_____

学 院：_____信息与计算机工程学院_____

2020 年 12 月 31 日

选题依据（选题经过，现状动态，初步设想及创新点等）及可行性论述

1. 选题依据

多年来，软件工程一直致力于解决软件危机问题，如软件开发周期长、成本高、质量差、维护困难等，软件工程采用工程的概念、原理、技术和方法来开发与维护软件，把经过时间考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来，以经济地开发出高质量的软件并有效地维护软件。其中，软件测试作为软件生命周期中一个不可或缺的阶段，针对软件质量差的问题提出了解决方案，对应用软件的确认测试工作，测试人员需要从系统级的软件需求规格说明中构造测试用例，并依据测试用例对软件进行确认测试，以发现软件中尽可能多的故障，使软件具有高可靠性，高可信性。

从软件本身的质量保证而言，软件的可靠性主要取决于两方面，一是软件开发的方法与过程，二是软件产品的测试与验证。测试用例是测试工作的实际操作纲要，也是评估测试结果的度量基准，高质量的设计用例能有效的保障软件测试的质量。软件测试用例设计的难点在于：对状态变量繁多，状态变量之间存在复杂的耦合情况的系统，通过人工构造测试用例时，往往出现考虑不周，设计不全的问题，因此无法通过非形式化的人工审查的方式保证设计出来的确认测试用例对需求进行了充分的覆盖。

本选题针对目前软件测试过程中，测试用例的设计和管理进行项目开发，其中测试用例的管理方便测试用例评审和复用，通过人工评审方式尽可能提高覆盖率，同时在后续同类型业务的测试中，通过复用减少工作量，同时也避免因当时考虑不周而遗漏测试点的情况。除此之外，可以根据已有的科学的测试理论，实现基于软件需求的测试用例自动生成功能，将部分人工工作量转化为自动工作量，提高测试工作效率，降低人为因素对测试过程的干扰，降低基本测试用例的冗余性，减少遗漏，排除测试的随机性和盲目性。

2. 现状分析

在国外一些软件行业发达的国家，软件测试技术已经发展了非常长的时间，也相对国内得到更多重视，技术日益成熟。在一些大型软件系统开发公司，测试人员在其员工中占有相当大比重。各种测试软件、自动化测试工具应运而生。软件测试在技术方面也在不断的高提，致力于迈向于通用化、标准化、网络化、自动化的方向。目前国外同类软件主要有 Rational公司的SQA Manager产品，它是SQA Suite测试软件包的一部分，该软件包以测试工具SQA Robot和SQA LoadTest为主。SQA Manager一般用做和测试工具的结合使用，为英文系统，因此产生的各类报告格式西化，没有测试案例具体步骤的管理查询，而且是以客户端的形式呈现的。在产品定位上，面向高端客户，价格昂贵，SQA Manager作为SQA Suite软件包的一部分捆绑出售，不能单卖，用户购买该软件包后经常只需用其中一部分功能，造成不必要的开销。

我国的软件测试技术于上世纪八十年代起源，并伴随着软件工程领域的研究发展而发展。近年来，国内软件行业发展也越来越迅速，但是软件测试技术发展却相对缓慢。国内软件测试水平在国际上也还是属于较年轻类型，具体表现在对软件测试还不够重视，测试单一化，质量监督体系不够完备，自动化程度不够高等方面，软件测试人才缺口也是非常大。一些中小型企业没有形成较为完备的测试管理体系，导致在生产出的产品质量无法达到期望的水准，错误率也比较高。当然现今也有很多企业由起初的“重研发，轻测试”逐渐转变，软件测试的地位也逐渐提高，会有逐步完备的测试管理体系，不断改进的自动化测试工具，日益成熟的测试技术，软件测试将得到更多重视。目前国内使用比较普遍的是i-Test管理系统，是由中科软科技股份有限公司开发的，在笔者实习过程中，公司测试相关工作采用的是 i-Case管理系统。上述系统主要针对测试用例的增删改查、导入导出等功能提供了解决方案，具有一定通用性，但功能单一，且手工录入测试用例工作量大，可能导致覆盖率降低；执行状况记录功能没有得到重视，不能及时更新执行结果，会导致测试的随机性和盲目性。

3. 初步设想

根据现状分析来看，使用Excel表格记录、保存测试用例的传统方法已经不足以满足质量保证需求，开发一个具有可视化界面的测试用例管理系统显得颇为重要，因此本系统是一个测试事务管理平台。

一方面，该系统具备已有的测试用例增加、删除、修改、查询和导入导出功能，完善执行结果的记录功能；另一方面，系统也会考虑减少测试人员工作量，提供基础测试用例集，根据需求文档分词自动生成相应测试用例的功能，涉及自动化测试，会设计类似GitHub功能的自动化框架代码管理模块，增加测试用例覆盖率的自动化分析以及通过深度学习进行自动化生成测试用例的迭代优化。

基于上述需求，计划采用B/S架构完成该系统设计。

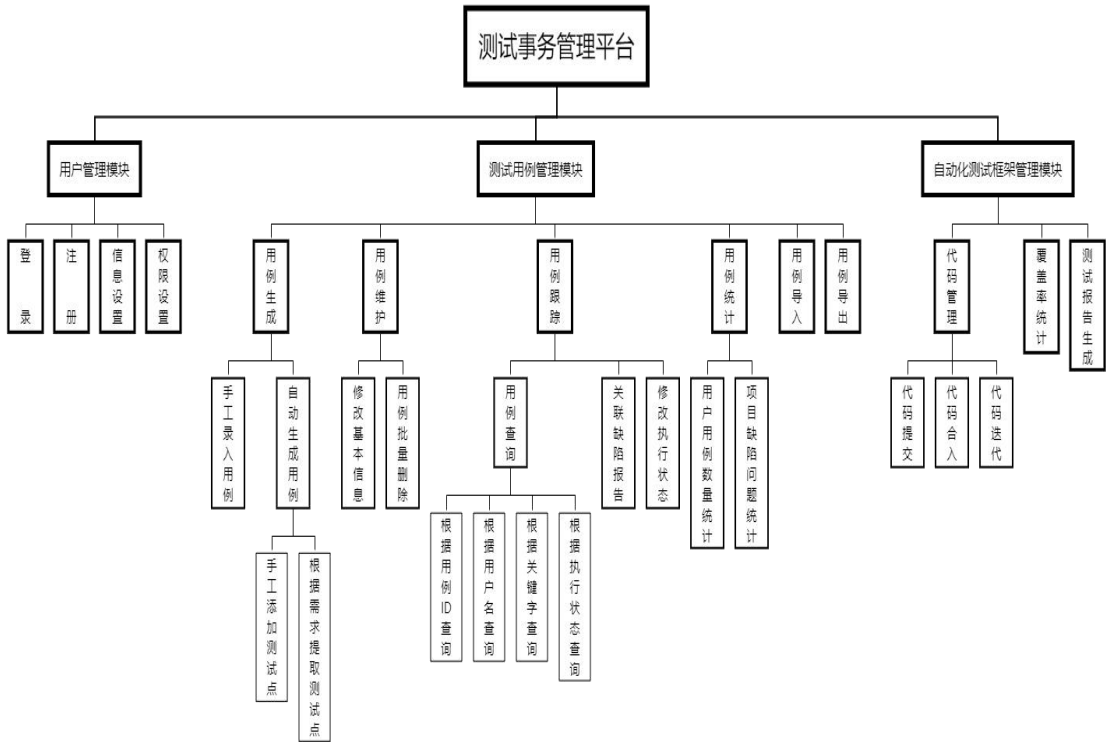


图 1 系统功能结构图

3.1预期系统用户

软件开发自测人员、软件测试人员。

3.2预期业务功能

A. 用户管理模块：

本管理系统主要包括三类用户，管理员、开发人员与测试人员。

基本功能包括：登录、注册、用户信息设置，管理员可以进行权限设置，开发人员可以根据不同的权限跟进不同项目。

B. 测试用例管理模块：

软件工程中的测试用例是一组条件或变量，测试者根据它来确定应用软件或软件系统是否正确工作。测试用例一般包括测试用例编号、用例标题、用例描述、测试步骤、预期结果、相关依赖（脚本、数据等）、测试分类、创建人、执行人等信息，测试完成后还应当追加测试状态，包括是否通过和必要的备注信息以及关联缺陷报告等。

该模块包括测试用例的新建（包括手动创建和自动创建）、批量导入测试用例、测试用例的修改、导出测试用例、删除测试用例、测试用例查询、测试用例执行状态标记、测试用例关联缺陷报告等子功能。

(1) 用例显示:

用户打开测试用例模块，目录中显示为所有项目列表，项目下为测试用例，根目录可以显示所有的测试用例。

(2) 用例生成:

(a) 手工录入测试用例：当新建测试用例时，会按规则自动生成一个唯一的 ID，便于查询和唯一标识，然后依次填入用例标题、用例描述、测试步骤、预期结果、相关依赖（脚本、数据等）、测试分类等信息，并确认新建，信息完整性校验之后，则新建成功。

其中用例信息还包含创建人和执行人字段，创建人为录入基础信息的用户，执行人为修改执行状态的用户。

(b) 自动生成测试用例：根据已有的测试点对应测试用例集自动添加相应基础测试用例，例如：功能测试对于取值范围使用边界值分析法、对于字符串输入采用等价类划分法，对于参数组合采用判定树/判定表法，对操作流程判断采用分支覆盖法等；性能测试提取并发数、并发持续时间、业务类型及业务占比、生产环境基础数据量、预期响应时间、系统其他特殊性能值需求（如net I/O不能占用带宽1/2）等；兼容性测试可根据自动爬取网络上主流的浏览器类型、操作系统、手机端机型的统计数据 and 导入项目组在APP启动时埋点生成的数据分析报表资源，生成推荐测试列表；安全测试关注输入内容的敏感信息加密、批量操作的可行性、密码的SQL注入等情况。

测试点的生成可以直接上传测试点文档，也可以上传需求文档，通过分词算法进行测试点提取后生成。

(3) 用例维护:

(a) 修改基本信息：可以进行上述测试用例基本信息的修改；

(b) 修改执行状态：测试完成后对应编辑执行状态，方便后续查看是否测试通过；

(c) 关联缺陷报告：如测试不通过，测试人员可提交相应缺陷报告，记录对应链接在测试用例中，方便后入回归；

(d) 批量删除用例：如录入有误，可以进行测试用例的删除，测试用例列表采用复选框，可以进行单选或批量删除。

(4) 用例跟踪:

采用字符串匹配算法，根据查询条件进行查询，查询条件包括：根据用例ID查询、根据用户名查询、根据关键字查询、根据执行状态查询。

(5) 用例统计:

(a) 用户用例数量统计：采用SQL语句，根据数据库中测试用例表下的创建人字段，统计用户生成的测试用例数量；

(b) 项目缺陷报告统计：根据缺陷报告字段，统计改项目目录下的缺陷报告数量。

(6) 用例导入和导出:

可导出标准模板Excel表，填写完成各字段后上传至系统；也可批量选中测试用例后，导出测试用例Excel表。

C. 自动化测试框架管理模块:

(1) 代码管理:

实现GitHub功能，GitHub是一个面向开源及私有软件项目的托管平台，只支持Git作为唯一的版本库格式进行托管。测试人员通过Git将本地代码提交至同一个项目仓库，进行合入、更新等代码维护工作，实现合作模式的自动化测试工作。

(2) 覆盖率统计:

使用JaCoCo插件，Jacoco是一款开源的覆盖率工具，可以嵌入Ant、Maven中，也可以使用JavaAgent技术监控Java程序。方便收集测试过程中代码覆盖情况，能够很直观展现哪些代码已经测试过，哪些没有被测试，帮助测试人员有针对性地增加用例，提升测试质量。

Jacoco使用插桩的方式记录覆盖率数据，通过probe探针来注入。插桩模式有两种：

(a) on-the-fly模式: JVM通过-javaagent参数指定jar文件启动代理程序, 代理程序在ClassLoader装载一个class前判断是否修改class文件, 并将探针插入class文件, 探针不改变原有方法的行为, 只是记录是否已经执行。

(b) offline模式: 在测试之前先对文件进行插桩, 生成插过桩的class或jar包, 测试插过桩的class和jar包, 生成覆盖率信息到文件, 最后统一处理, 生成报告。

相比之下on-the-fly更方便简单, 无需提前插桩, 无需考虑classpath设置问题。但是以下情况不适合使用on-the-fly模式:

- ①不支持-javaagent;
- ②无法设置JVM参数;
- ③字节码需要被转换成其他虚拟机;
- ④动态修改字节码过程和其他agent冲突;
- ⑤无法自定义用户加载类。

基于系统应用场景, 采用on-the-fly模式。

(3) 测试报告生成:

根据JaCoCo执行生成的jacoco-client.exec文件, 生成覆盖率报告index.html文件, 保存至对应自动化测试用例代码目录下, 形成测试报告。

3.3 预期架构实现

网络技术的日益成熟, 使得C/S架构似乎已经达不到现今信息化的水准了。B/S架构作为C/S架构的改进与升级, 已经呈现出要取而代之之势。B/S应用越来越广, 目前大部分的网站, 特别是电商网站, 都采用了B/S架构而构建。B/S架构使我们不再需要开发出一个客户端软件, 这便利了系统的维护和版本的升级; 可跨平台操作, 不再考虑不同操作系统开发的不同程序, 只要安装浏览器软件, 就可以作为客户端来访问系统; 安全性好, 防火墙保证了系统的安全性。在B/S体系结构系统中, 用户通过浏览器向分布在网络上的许多服务器发出请求, 服务器对浏览器的请求进行处理, 将用户所需的信息返回到浏览器, B/S结构简化了客户机的工作, 客户机上只需配置少量的客户端软件。结构模型如下图所示:



图 2 B/S 两层结构模型图

鉴于以上优势, 本设计选择了B/S架构。

4. 创新点

该系统基于GUI可视化界面, 打造一款面向测试人员和开发自测人员的测试用例管理平台。创新点如下:

(1) 业务功能上, 从当前实际情况出发, 为减少测试工作量, 可自动生成基础测试用例, 用户在节省录入测试用例的时间的同时, 不会遗漏基础测试点; 开发人员可以通过基础测试点进行自测, 通过后进行提测。提供对测试用例执行结果的统计分析。测试结果以图表形式生动、直观地反映测试用例的执行结果, 为下次回归测试筛选测试用例做好充分准备。

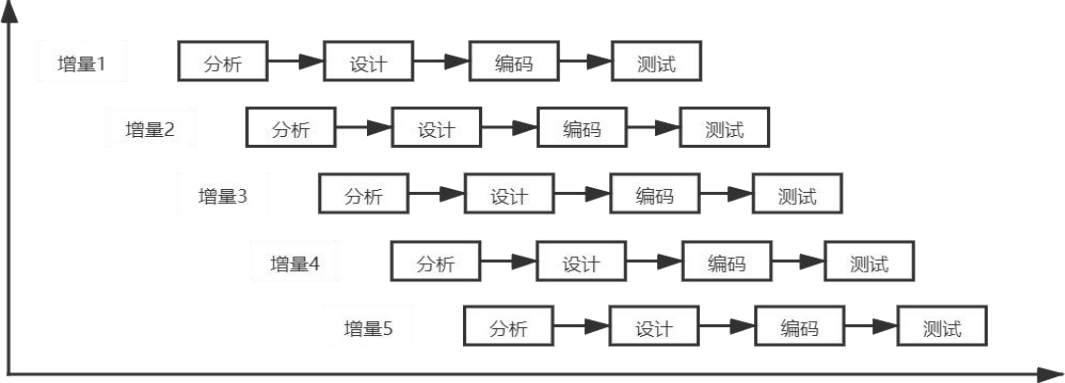
(2) 模块上, 系统包含完整、规范的系统业务流程, 制定了规范的软件测试流程, 包括测试用例的设计、评审、补充、执行、记录、回归等, 引导项目参与人员进行有效地软件测试, 达到提高软件质量的最终目标。针对不同用户开放不同的功能权限, 满足管理员开发人员、测试人员的需求, 使系统易于管理和维护, 项目进度可以精准跟进。

(3) 核心算法:

(a) 分词算法HanLP: 用于文档分析, 根据分词结果生成测试点集。

构建词网, 将词网转化为词图, 词图含有各条边以及边上的权值, 接下来采用维特

	<p>比分词器—基于动态规划的Viterbi算法，算法可以概括为下面三点：</p> <p>①如果概率最大的路径经过网络的某点，则从开始点到该点的子路径也一定是从开始到该点路径中概率最大的；</p> <p>②假定第i时刻有k个状态，从开始到i时刻的k个状态有k条最短路径，而最终的最短路径必然经过其中的一条；</p> <p>③根据上述性质，在计算第i+1状态的最短路径时，只需要考虑从开始到当前的k个状态值的最短路径和当前状态值到第i+1状态值的最短路径即可，如求t=3时的最短路径，等于求t=2时的所有状态结点x2i的最短路径加上t=2到t=3的各节点的最短路径。</p> <p>(b) 字符串匹配算法KMP：用于关键字查询、匹配测试点集自动生成测试用例。</p> <p>KMP算法一种改进的模式匹配算法，它的改进在于：每当从某个起始位置开始一趟比较后，在匹配过程中出现失配，不回溯i，而是利用已经得到的部分匹配结果，将一种假想的位置定位“指针”在模式上向右滑动尽可能远的一段距离到某个位置后，继续按规则进行下一次的比较。</p> <p>算法流程：</p> <p>①规定i是主串S的下标，j是模式T的下标。现在假设现在主串S匹配到 i 位置，模式串T匹配到 j 位置；</p> <p>②如果j = -1，则i++，j++，继续匹配下一个字符；</p> <p>③如果S[i] = T[j]，则i++，j++，继续匹配下一个字符；</p> <p>④如果j != -1，且S[i] != T[j]，则 i 不变，j = next[j]，此举意味着失配时，接下来模式串T要相对于主串S向右移动j - next[j] 位。</p> <h2>5. 可行性分析</h2> <h3>5.1. 技术可行性分析</h3> <p>(1) 本系统的主要作用是进行测试用例管理和根据文本生成用例，对于测试点的挖掘可以使用哈工大提供的HanLP库的API对大量需求文本进行分词处理，再从中提取出常见的测试点，对测试用例的存储可以使用MySQL。</p> <p>(2) 系统本身基于Java实现，平台基于Web实现，不存在技术盲区。</p> <h3>5.2. 用户可行性分析</h3> <p>基于针对身边开发人员和测试人员的调查，众多用户期待能够有一款易于使用，使用直观，界面友好，且具备一定自动化便捷功能的测试用例管理平台。</p> <p>系统的用户群体为软件测试开发工程师，因此界面只需要清晰便捷即可，使用Vue.js和Bootstrap等前端主流框架可以完成。</p> <h3>5.3. 经济可行性分析</h3> <p>(1) 硬件成本</p> <p>正常的具有开发环境的笔记本电脑的性能足够完成该系统。</p> <p>(2) 人力成本</p> <p>本项目的实现阶段由设计者个人独立完成，并且有足够条件在预期的时间内完成。</p> <p>(3) 经济成本</p> <p>系统开发阶段无经济成本，在试运行阶段则需要租借服务器保证项目正常使用，不过该经济成本在可接受的范围之内，不影响开发。</p> <h3>5.4. 结论意见</h3> <p>该研究在技术、经济、用户都具备足够的可行性，系统可进行开发。</p>
设计撰写过程中拟采取的设计方	<h2>6. 技术路线</h2> <p>开发语言：Java、JavaScript、Python</p> <p>开发框架：Spring Boot、Vue.js、Bootstrap、Nginx、Mabatis</p> <p>数据库：MySQL</p>

案、方法、手段或实验方案、实验路线等	<p>开发工具：IntelliJ IDEA、Pycharm、MySQLWorkbench</p> <p>软件周期实验路线：根据进度安排，优先开发核心模块，选用增量模型。</p> <p>增量1：用户管理模块开发</p> <p>增量2：测试用例管理模块基础框架开发</p> <p>增量3：测试用例自动生成功能开发</p> <p>增量4：自动化测试框架管理模块基础框架开发</p> <p>增量5：测试用例关联测试报告功能开发</p>  <p style="text-align: center;">图 3 软件周期实验路线图</p> <p>可行性研究：基于用户需求进行初步的可行性分析与调研，例如调查是否存在相应数据源、技术上是否可行、是否符合上线运营规范等。</p> <p>需求分析：可行性研究完成后，若可行，则针对具体的用户群体进行用户调研，确定系统的最终需求。</p> <p>概要设计：基于用户的需求对平台的架构与功能模块等进行设计。</p> <p>算法实现：针对较为独立的模块，基于相关技术对相关算法与评价体系进行独立研究。</p> <p>详细设计：基于核心功能模块算法，对主要功能进行详细设计。</p> <p>系统开发：对所有功能进行集成、编码和开发。</p> <p>系统测试：对系统的关键算法与模块进行测试，重点测试相关算法的精确性，改进算法的改进程度、应用模块的逻辑严谨性等。</p> <p>试运行与部署：所有开发步骤完成后，基于阿里云服务器对项目进行部署并试运行。</p>
计划进度及其内容	<p>2020 年 12 月 01 日~2020 年 12 月 31 日 撰写开题报告。</p> <p>2021 年 01 月 01 日~2021 年 01 月 10 日 需求分析。</p> <p>2021 年 01 月 11 日~2021 年 01 月 20 日 概要设计与详细设计。</p> <p>2021 年 01 月 21 日~2021 年 02 月 28 日 测试用例管理系统基本框架搭建。</p> <p>2021 年 03 月 01 日~2021 年 03 月 20 日 核心业务的技术分析与实现。</p> <p>2021 年 03 月 21 日~2021 年 03 月 25 日 软件测试。</p> <p>2021 年 03 月 26 日~2021 年 03 月 31 日 软件上线试运行与调试。</p> <p>2021 年 04 月 01 日~2021 年 05 月 15 日 完成毕业设计（论文）撰写。</p> <p>2021 年 05 月 16 日~2021 年 06 月 准备进行毕业答辩。</p>
写作提纲	<ol style="list-style-type: none"> 1. 绪论：论述本文研究问题提出的背景、目的和意义，突出本文所要研究和解决的问题 2. 技术分析：自动生成测试用例实现难点分析及研究 3. 系统需求分析：进行系统功能需求分析、可采用的相关技术分析等。 4. 系统设计：本部分主要包括功能模块概要设计、详细设计、数据库设计以及算法设计。 5. 系统功能实现：包括系统采用的开发工具、典型程序代码和典型系统运行界面。 6. 测试：包括单元测试、集成测试、系统测试以及压力测试。 7. 结论：对系统开发工作效果的评价与展望。

指导教师意见	<div>签名：年 月 日</div>
专业意见	<div>签名：年 月 日</div>

注：纸张填写不够可另加附页。