

**毕 业 论 文(设 计)**

**论文（设计）题目:**

**加强过程管理的网上项目实践平台的设计与实现**

姓 名 **周钊屹**

学 号 **201600130051**

学 院 **计算机科学与技术学院**

专 业 **计算机科学与技术**

年 级 **2016级**

指导教师 **梁文革**

2020年 5月 10日

**­目 录**

[摘 要 I](#_Toc38890709)

[ABSTRACT II](#_Toc38890710)

[第1章 绪 论 1](#_Toc38890711)

[1.1 选题背景和研究意义 1](#_Toc38890712)

[1.2 Git的国内外研究现状 1](#_Toc38890713)

[1.2.1 Git的国内研究现状 1](#_Toc38890714)

[1.2.2 Git的国外研究现状 2](#_Toc38890715)

[1.3 本文主要工作 3](#_Toc38890716)

[1.4 论文组织结构 3](#_Toc38890717)

[第2章 需求分析 5](#_Toc38890718)

[2.1 系统概述 5](#_Toc38890719)

[2.2 解决问题和基本要求 5](#_Toc38890720)

[2.3 系统目标 6](#_Toc38890721)

[2.4 功能性需求 7](#_Toc38890722)

[2.5 非功能需求 8](#_Toc38890723)

[第3章 系统设计 9](#_Toc38890724)

[3.1 工作流程 9](#_Toc38890725)

[3.2 系统模块设计 9](#_Toc38890726)

[3.3 系统数据库设计 9](#_Toc38890727)

[第4章 实现与测试 10](#_Toc38890728)

[4.1 系统实现 10](#_Toc38890729)

[4.2 系统测试 10](#_Toc38890730)

[第5章 总结 11](#_Toc38890731)

[致 谢 12](#_Toc38890732)

[参考文献 13](#_Toc38890733)

[附录1 代码及相关附件 14](#_Toc38890734)

[附录2 文献英文原文 15](#_Toc38890735)

[附录3 文献中文译文 16](#_Toc38890736)

**加强过程管理的网上项目实践平台的设计与实现**

# 摘 要

不论是在大型互联网公司还是学校平时做的一些实验，一个软件的开发或者课程设计的完成，都需要很多人的分工合作才能完成。而在多人合作开发实现一个项目的时候就会出现很多问题，最主要的问题就是版本管理和如何多人合作开发，多人合作开发的流程模式是怎样的。

本课题针对多人合作的项目式开发进行过程化、系统化的管理。拟结合GitHub和项目实践的特色，解决在多人合作中出现的许多常见问题，最终便于版本管理、多人合作、项目进度管理和历史项目的学习以及持续改进。整个课题的难点在于实现版本管理和多人合作的问题，最终通过Git提供的许多功能实现了上述基本要求。整个项目通过Java Maven创建项目，利用Swing创建图形界面，版本管理和多人合作功能通过Git在Java中提供的jar包JGit实现。整体的工作流程简述如下:项目负责人先创建项目，其他合作开发同伴Fork相应项目。负责人可以对合作开发同伴授权，如果有授权，合作者就不用通过pull request功能先发起请求，才能将合作者完成的功能提交到项目负责人的项目中；如果没有授权，合作者就需要先通过pull request发起合并请求，在项目负责人看过可以合并之后才能合并，如果项目负责人不同意则不予合并。

**关键词：**项目实践平台；多人合作；Git；版本控制；

# ABSTRACT

Whether it is a large-scale Internet company or some experiments which is always done in school, the development of a software or the course design needs a lot of people to work together to complete. Many problems will arise when a project is developed by a lot of people in the same time. The main problems are version control and how to develop by a lot of people at the same time, and what is the process of multiplayer development.

This paper aims at the process-oriented and systematic management of project-based development with multi-person cooperation. It is planned to combine the characteristics of GitHub and project practice to solve many common problems that occur in multi-person cooperation, and ultimately facilitate the version control, multi-person cooperation, project progress management, learning of historical projects and continuous improvement. The difficulty of the whole subject lies in the issue of implementing version control and multi-person cooperation. Eventually, the above basic requirements are realized through many functions provided by Git. The entire project is created through Java Maven, using Swing to create a graphical interface. The jar package JGit provided by java realize the function of version control and multi-person cooperation. The overall work flow is briefly described as follows: the project leader first creates the project, and other cooperator Fork corresponding projects. The person in charge can authorize the partner to develop the project together. If authorized, the collaborator does not need to initiate a pull request before submitting the function completed by the collaborator; if there is no authorization, the collaborator needs to Initiate a pull request, and the project can only be merged after the person in charge of the project has approve the changes. If the person in charge of the project does not agree, it will not be merged anyway.

**Keywords:** Project Practice Platform; Multiplayer cooperation; Git; Version Control.

# 绪 论

## 选题背景和研究意义

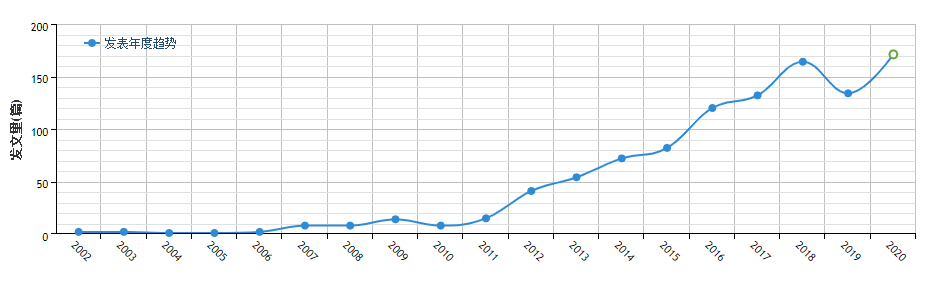
## Git的国内外研究现状

由于整个设计的功能大部分功能以及核心功能都是通过Git为Java提供的JGit包进行实现的，所以主要的研究现状就对Git的国内国外研究现状进行探索。

国内外直接研究现状没有直接文献进行叙述，所以通过一些论文网站的数据，即相关论文的发表数量作为一个研究现状的参考，国内的研究现状通过知网查询全文得到的论文作为研究现状的一个参考，而国外的部分参考ScienceDirect，而直接搜索Git会出现生物上的一些概念，应该是缩写的重复问题，需要用Git版本控制为关键字进行搜索，其中有些结果是单纯讨论Git或者GitHub的应用的文章[[[1]](#endnote-1)-5][[2]](#endnote-2)[[3]](#endnote-3)[[4]](#endnote-4)[[5]](#endnote-5)，利用Git的版本管理的特性实现各种各样不同的需求，还有些结果是一些利用Git嵌入自己的项目中实现一个具有需求功能的软件，和本文所阐述的设计有类似之处[[[6]](#endnote-6),[[7]](#endnote-7)]。

### Git的国内研究现状

国内的研究现状通过在知网用“Git版本管理”进行全文搜索，然后利用知网的计量可视化功能显示出总体趋势分析（见图 1‑1），主题分布太多太杂乱没有太多参考价值，在此不做引用，发文量的多少有一定参考价值，至少可以说明在这方面的研究或者使用量的多少。



**图 1‑1** **文献发表年度趋势**

从图上还是基本可以看出关于Git版本管理的文献数量从某个时间节点之后就开始一直上涨，到最近几年开始平缓，其中2020年的是预测值。Git这一技术是从2005年才开始有所开发，之前还有一些版本管理系统比如SVN等，但基本都是集中式版本管理，而Git是分布式版本管理。而后来很多相关文献和应用的增长或许与GitHub的出现有关，中国也有Gitee这样的基于Git的代码托管和研发协作平台[[[8]](#endnote-8)]。

### Git的国外研究现状

国外的研究采用在ScienceDirect上进行“Git version control”关键字查询，统计从2002到2020年以来的论文发表数量，最终得到的折线图见图 1‑2。

**图 1‑2 文献发表年度趋势**

在搜索结果中和国内的搜索结果一样可以大致分为两类，一类是直接利用Git或者GitHub进行辅助研究者工作的完成，简述其中的工作流程使其利于更多人使用。另一类是将Git的功能嵌入自己的软件中，利用其中的功能特性，比如基本的push，merge和branch功能，可以实现很多功能，不局限于版本管理，而且利用不同的Gitflow工作流程使得自己的工作更加容易被完成。

## 本文主要工作

文章主要用于阐述软件系统设计思路，从软件产生的背景到软件构思功能，即需求分析，然后说明软件整体需要怎样协调的设计才能完成需要功能，最后进行实现部分的详细说明和测试。本文一方面是对软件的系统化介绍，一方面用于在软件开发过程中的记录。

最终设计的系统将多人合作的项目式开发进行过程化、系统化的管理，结合项目实践的特色，首先实现了四个基本功能:版本管理、多人合作、项目进度管理、历史项目的学习，持续改进。并且为了更好地实现这些功能，还实现了一些比如对本地Git项目的管理操作，比如Push，Merge，Fetch等，以及一些为了安全性考虑并结合GitHub特色所应用的SSH Key管理和Pull request，Fork功能。

## 论文组织结构

论文第一章绪论主要介绍文章研究对象所研究的背景，由于设计是一个工程性项目，没有专门对某个学术部分进行研究，应用比较新的技术是Git的版本管理以及Git所提供的一些基本功能所实现的多人合作功能，所以绪论部分的很多背景研究都是探索的Git这个技术的背景研究。

论文第二章主要描述设计的系统基本概述，详细阐述系统设计想解决什么样的问题，以及需要具备什么样的基本功能去解决这样的问题，利用设计出来的基本功能用什么样的步骤去实现，即相应产品的工作流程。

论文第三章开始阐述系统如果需要实现之前需求分析所说的功能，需要如何去进行设计。软件一开始进入写代码阶段就比较需要有明确的目标，能够知道当前需要实现什么功能，并且实现功能的顺序也有所不同，这样有利于一边写一遍进行一些小的测试用来确保当前完成的小功能能够正常工作，如果最后一起再进行测试就会容易找不到具体错误的地方。软件如果进入代码阶段，写到后面发现该功能不能这样去实现，进行修改就会很麻烦，因为功能之间大都是有所联系的，而且修改起来需要重新测试，过程繁杂，因此第三部分的系统功能的具体到每个板块的设计还是很有必要的。

论文第四章正式阐述论文具体每个功能是如何实现的，和需求分析的功能进行对比就可以评估软件的完成程度。该章主要分成系统具体实现部分和项目测试部分。具体实现结合关键代码对细小的功能进行阐述，这些功能不一定直接就是最后的要求的功能，更有点类似数学的公理化方法，从一些定理进行推导得到最后的结论。项目测试功能利用一些设想的使用软件的操作流程，将其作为输入，然后用软件正式进行运行，最后测试这样的功能是否完成，完成情况是否符合预期。

论文第六章是总结部分，这部分主要用于会看论文设计过程遇到的问题和一些闪光点，并且对实现的功能做一些评估，是否可以用别的更好的方式将这些功能实现，如果还有时间之后可以进行的一些改进的方向。

# 需求分析

## 系统概述

项目设计主要分成两部分。第一部分是非本地部分，非本地部分主要用来实现一些远程服务器实际进行的一些操作，以及模拟一些网站才会进行的操作，比如创建仓库，pull request和fork，clone之类的一些和不同用户之间交互有关的功能。另外一部分就是本地部分，进行本地git仓库的一些操作。比如本地有一个git仓库，现在需要对这个仓库内容进行一些修改，然后才将其上传到远程服务器用于多人操作，所以有了打开本地git仓库，以及一些多人合作所需要的基本功能，比如add，commit，push到远程分支，创建删除切换本地分支，以及fetch和回到历史的某些版本功能。

## 解决问题和基本要求

项目需要对多人合作的项目式开发进行过程化以及系统化的管理。解决在多人合作中出现的许多常见问题，最终便于版本管理、多人合作、项目进度管理和历史项目的学习以及持续改进。最基本的要求就是实现上面的四个功能，每个功能初期的设想实现方式简述如下，最后详细实现会在本文后面章节详细阐述。

* + - 1. 版本管理

版本管理手动实现可以利用一个日志文件，手动记录每次Commit的内容以及修改的文件，文件改变的内容需要想办法保存下来。这一点并不好实现，设想可以通过一些第三方工具来实现，版本管理工具中有很多出色的三方工具，比如集中式管理的SVN和分布式版本控制系统Git，所以最后可以实现不用自己手动去实现版本管理这样的操作，而是直接借助于三方工具来实现。而GitHub和Gitee这样的网站也都是用了Git作为版本管理的工具实现的在线代码托管平台，并且用于多人合作，所以可以借鉴一下这些平台的实现方式，以及结合项目实践的特色完成版本管理。

* + - 1. 多人合作

多人合作需要解决如果多人完成一个项目，开始可以考虑的有腾讯文档那种直接多人进行合作，但是鉴于真正代码开发项目时的多人合作不太可能存在多人同时编辑一个文档的情况，都是分开完成相应功能之后再将各自完成的功能整合起来。另外多人合作还需要解决仓库存储问题，即需要模拟多用户创建属于自己的仓库，这样就必然需要一个空间去存储这些仓库，利用Gitee网站自己创建一个账号，命名特殊来模拟多人的一个情况是一种可能的选择，但是由于项目每次需要创建一个新项目，即需要在网页点击相应内容，所以需要网站爬虫的一些内容，结果是不太好爬。最后可行的办法就是自己租一个服务器，在服务器上用与用户名称同名的文件夹模拟用户，里面就可以用于自己存储仓库。

然后还需要如何实现多人合作的一个工作流程，这样的工作流程需要按照实际需要进行设定，不需要完全按照基本的gitflow流程，但是需要能够安全并且完全地实现多人合作的流程。

* + - 1. 项目进度管理

一个软件最好还可以实现一些进度管理之类的工作。需要知道整个项目的进程，不仅有利于项目进度的明示，而且有利于让开发不同功能的合作同伴知道当前自己的工作开发到什么步骤了，而且参考接下来可能的进度还可以让当前完成功能的人有意识地留下一些接口为了后来的功能实现更加方便。

* + - 1. 历史项目的学习以及持续改进

除了当前正在完成的项目之外，还需要将软件打造成类似于一种在线社区的 氛围，这样有利于软件开发的交流与借鉴，并且历史项目的展示也是多人软件管理的一种需要，只有将历史项目进行展示，才能让更多人可以知道这个项目，从而进行共同开发。

## 系统目标

系统的整体目标是在完成版本管理、多人合作、项目进度管理、历史项目学习，持续改进这四个主要的基本功能的前提下，为了更好地实现多人合作的项目式开发进行过程化、系统化的管理并结合一些已经运用Git技术的平台比如GitHub，更多地去实现一些方便的功能。

## 功能性需求

在正式实现软件上诉四个基本功能之前需要将功能具体细化，细分成更多可以直接明确实现的功能，然后下面具体阐述需要实现的功能。

* + - 1. 登录注册功能

需要管理历史用户和新建用户信息，数据库管理用户账号和密码对应信息并且对注册账号形式进行限制，即不规范字符不允许作为用户名。该功能重点在于实现之后代码的安全性以及鲁棒性。

* + - 1. 项目展示功能

这个功能其实是为了历史项目学习功能做铺垫。想要学习历史项目首先得能够知道历史项目有哪些，并且这个功能还可以用来让不同的用户相互交流借鉴项目。基本要求应该是需要可以展示项目目录的，在基本功能上还可以进行提升的部分就是，可以做成一种富文本，直接就可以点击查看想要查看的文本内容，至少二进制文件都可以直接查看。

* + - 1. 版本管理功能

版本管理至少应该可以按照一些历史版本的信息使得用户可以对历史版本有选择性地返回到该版本。如果在基本要求上对功能进行扩展就是在可以在返回版本前对该版本提供更多的信息，最好就是直接可以模拟回到该版本之后文件状态时怎么样的，文本中的内容状态是否是想要回到的版本，如果是再确定回到该版本，如果不是就不选择回到该版本。

* + - 1. Git基本操作功能

这是最难实现的一个功能，因为多人合作首先需要自己构思设计一个多人合作的流程，不一定需要标准的gitflow流程，但是需要一个可以方便并且安全实现多人合作的流程。但是有一点可以肯定的是，不论用什么样的流程去实现多人合作功能，必须要实现Git的一些基本操作:Clone，Push，Merge，Commit，Add等。这些功能的实现有些需要再远程服务器上实现，有些需要在本地实现。远程服务器可以考虑直接用java对远程服务器执行命令行操作实现，而本地就需要jgit提供的接口实现相应功能。

Git的基本功能还需要对本地仓库可以进行操作。一个合作者在工作时毕竟只能在本地进行代码的一些编辑，但是最后还是需要add并且commit和push修改过的内容，所以需要对本地仓库需要专门有一个模块用来实现上述功能，并且图形界面显示出来，完全不用借助于用户自己在命令行中运行git功能就实现本地仓库的管理功能。

* + - 1. 项目进度管理功能

项目进度管理功能可以简单实现，但是必须自己定义什么是项目进度，然后在完成相应任务或者功能的时候，可以对项目进度进行同步更新。

* + - 1. 界面美化功能

由于整个系统的图形界面是利用Swing来实现的，做网页版本前端需要花费很多时间，为了先把功能实现出来，所以最后还是采取了Swing来实现图形界面。而Swing本身的界面并不好看，所以在完成基本功能之后需要对Java Swing的界面进行一定程度的美化。

* + - 1. 安全性功能

Git本身在进行传输时，例如Push或者Clone，都是使用的SSH或者HTTP协议进行传输，所以如果在用SSH协议进行传输时需要对用户的SSH Key进行相应的管理，并且在每次连接的时候都采用RSA产生的Key进行确认的形式进行连接远程服务器，从远程服务器获取或者传输相应内容。

## 非功能需求

* + - 1. 正确性

基本实现所要求的功能，并且自己模拟用户实现起来时不会出现明显功能上或者明显卡住的问题。

* + - 1. 稳定性

如果软件出现一些多人同时的访问或者数据库在运行时网络中断能够保证数据不被影响，或者可以保存进度之后继续，不过前者实现起来更简单。

* + - 1. 可靠性

尽量做到软件的结果要么是正确的，要么就直接给出错误信息，不能程序还在正确运行，但是给出或者展示出的结果却是错误的。这样会给用户错误的误导，自己还认为程序运行正确，得到的结果也是正确的。

* + - 1. 性能、效率

软件在运行时不能点击之后要等很久才能执行完一个功能，需要对代码进行不断的优化，尽量使得用户等待程序响应时间很短，程序运行起来需要很流畅。

* + - 1. 安全性

数据库的安全性，不能被恶意连接，以及数据库内容需要备份等。多人项目之间需要有权限的区分，别人的项目只能进行读取，如果没有收到项目负责人的授权不能直接进行修改操作。远程服务器安全以及SSH连接安全需要有一定保障。

* + - 1. 可移植性

软件在不同操作系统以及不同电脑上可以同样进行运行，并且数据库中的内容需要保持一致性，这样需要在

* + - 1. 兼容性

对于软件中用到的接口之类，需要针对不同版本尽量做到更多的兼容，如果可以还可以单独写一些函数类用于提供接口。

* + - 1. 鲁棒性

软件功能虽然是这么设计的，但是用户可以进行很多匪夷所思的操作，不管用户进行了操作，系统程序尽量做到不能崩溃，如果用户进行了非常错误的操作，需要程序能够继续正常运行并且给用户提供提示信息表明刚才的操作不是正确操作，在之后用户进行正确操作之后还能正确返回结果。

# 系统设计

## 工作流程

## 系统模块设计

## 系统数据库设计

# 实现与测试

## 系统实现

文章该部分详细对基本功能进行阐述，每个功能主要分成对功能实现的描述和界面效果展示以及关键代码示例。

* + - 1. 登录界面以及新项目的创建

1. 功能实现

登录界面是软件的基本功能，主要涉及对数据库中用户的账号密码的一些管理，用户注册和登录需要在数据库中进行相应的查找和插入。这个部分还需要进行整体构思，之后用户的合作项目需要存储在什么地方。最开始的思路是不租用服务器，直接利用Gitee提供的空间来模拟有服务器的情况，即自己注册一个Gitee的账号，然后在自己软件创建用户时，只在数据库中进行记录，真正需要创建项目时，直接通过网络爬虫的方式发送请求，在Gitee中创建一个用户名加项目名称作为项目名称的项目，只是由于Gitee或者GitHub每次创建项目时都需要手动点击创建项目，网络爬虫写起来有点繁琐，所以还是租借了服务器。

整个流程为，在软件中创建用户时，除了在数据库中插入用户和密码信息，还需要通过Java对远程服务器进行操纵，在远程服务器根目录下创建一个和创建的用户名同名的目录，这个目录之后就用于存放该用户创建的所有新项目。并且创建的项目都直接用裸仓库实现，即不让用户可以在远程服务器中进行任意的更改，在目录中只能看见git运行所需要的文件，而没有用户自己创建的文件，无法当作工作仓库。

1. 界面展示
2. 关键代码

## 系统测试

# 总结

# 致 谢

不需要“花团锦簇”，不需要“无病呻吟”，需要的是：洗尽铅华、朴素是真。

# 参考文献

# 附录1 代码及相关附件

# 附录2 文献英文原文

按前述规范排版！

# 附录3 文献中文译文

按前述规范排版！

1. [] 包永红. Git在 JavaWeb技术实践教学课程中的应用与探索[J]. 内蒙古农业大学学报，2019，21(107):32-36. [↑](#endnote-ref-1)
2. [] 周伟，陈柳. Git 在软件版本管理实验教学中的应用[J]. 信息技术与信息化，2016，5:97-100. [↑](#endnote-ref-2)
3. [] 徐娅. Gitb版本控制工具在团队协作项目中的应用[J]. 智能计算机与应用，2019，9(5):341-343. [↑](#endnote-ref-3)
4. [] 曾少宁. 基于GitHub平台的协同式实验教学方法[J]. 计算机教育，2016，(12):144-148. [↑](#endnote-ref-4)
5. [] 任维，李伟，袁海娣. 基于GitHub平台的高校软件类课程教学方法研究[J]. 电脑知识与技术，2019，15(2):81-82. [↑](#endnote-ref-5)
6. [] 陶健. 基于git托管平台的代码存储系统的设计与实现[D]. 武汉:华中科技大学. 2018. [↑](#endnote-ref-6)
7. [] 侯效永，李良伟，孙召. 基于Git的代码托管平台JLUCODE[J]. Computer Era，2016，(12):32-38. [↑](#endnote-ref-7)
8. [] 字凤芹. 基于Git的协作小组学习资源库的建设与研究[D]. 云南:云南大学. 2016. [↑](#endnote-ref-8)