**Gitea的分析与应用**

**项目计划**

Version 1.4

小组成员：

SY1906915 麦梓健

SY1906505 孙维华  
SY1906306 洪治凑  
SY1906432 王子璇  
SY1906117 王伟民  
SY1906434 郑锋

# 1.项目概述

## 1.1 背景

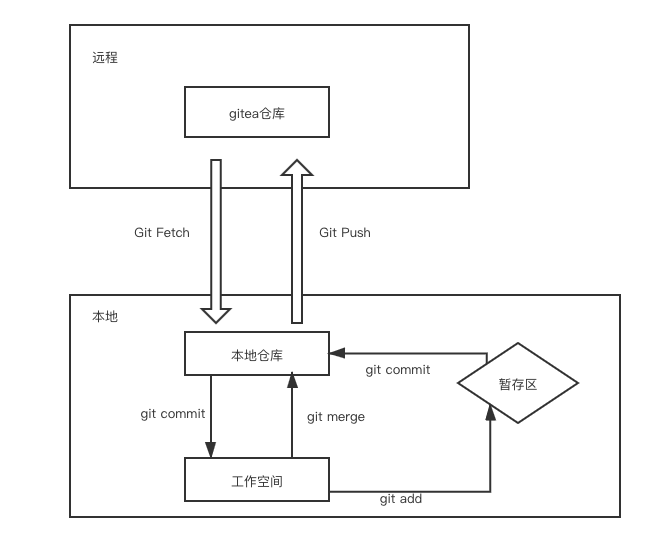
Linux 内核开源项目有着为数众多的参与者。 绝大多数的 Linux 内核维护工作都花在了提交补丁和保存归档的繁琐事务上）。 到 2002 年，整个项目组开始启用一个专有的分布式版本控制系统 BitKeeper 来管理和维护代码。到了 2005 年，开发 BitKeeper与Linux 内核开源社区的合作关系结束，这就迫使 Linux 开源社区（特别是 Linux 的缔造者 Linus Torvalds）基于使用 BitKeeper 时的经验教训，开发出自己的版本系统。 他们对新的系统制订了若干目标： 速度、简单的设计、对非线性开发模式的强力支持（允许成千上万个并行开发的分支）、完全分布式、有能力高效管理类似 Linux 内核一样的超大规模项目（速度和数据量）。自诞生于 2005 年以来，Git 日臻成熟完善，在高度易用的同时，仍然保留着初期设定的目标。 它的速度飞快，极其适合管理大项目，有着令人难以置信的非线性分支管理系统。

GitHub、GitLab等基于Git的代码托管平台，尽管功能强大，但是过重、过于复杂了，对于许多项目而言并不需要如此庞大的体系，因而轻量级的框架gogs等相继出现。Gogs简单、轻便、运行快，只需要廉价的树莓派即可满足需求。但是gogs由于“独裁”性质的维护，使得由社区维护的gitea出现了。

## 1.2 简介

Gitea由go编写，由社区维护，安装简便，运行需求极低。同时gitea的功能非常强大，具有大量的可扩展组件：

* 支持活动时间线
* 支持SSH以及HTTP/HTTPS协议
* 支持用户、组织和仓库管理系统
* 支持仓库和组织级别Web钩子
* 支持仓库Git钩子和部署密钥
* 支持仓库工单、合并请求以及Wiki
* 支持迁移和镜像仓库以及它的Wiki
* 支持自定义源的Gravatar和Federated Avatar
* 支持MySQL、PostgreSQL、SQLite3、MSSQL和TiDB



## 1.3 术语

|  |  |
| --- | --- |
| Git钩子 | Git钩子是在Git仓库中特定事件发生时自动运行的脚本。可以定制一些钩子，这些钩子可以在特定的情况下被执行，分为Client端的钩子和Server端的钩子。 |
| Wiki | Wiki是一种在网络上开放且可供多人协同创作的超文本系统，由沃德·坎宁安于1995年首先开发，这种超文本系统支持面向社群的协作式写作，同时也包括一组支持这种写作 |
| 镜像仓库 | 用来保存制作好的容器的一个仓库，只不过这里的容器被做成了镜像，所以我们称其为镜像仓库。 |
| SSH | SSH 为建立在应用层基础上的安全协议。SSH 是较可靠，专为远程登录会话和其他网络服务提供安全性的协议 |
|  |  |

## 1.4 标准

参照如下标准：

* GB/T 13702-1992 计算机软件分类与代码
* GB/T 19003-2008 软件工程
* GB/T 5538-1995 软件工程标准分类法
* GB/T 9385-2008 计算机软件需求规格说明
* GB/T 5532-2008 计算机软件测试规范
* GB/T 18221-2000 信息技术程序设计语言
* GB/T 8567-2006 计算机软件文档编制规范

# 2 项目内容

## 2.1 项目准备

1. 首先利用go命令或是直接到github上进行下载，获取源文件。

2. 其次取保自己的终端配置了go，node和make，这样就可以对项目进行编译

3. 其余工作就是仔细阅读gitea源码，深入了解项目。

## 2.2 工作内容

1. 通过阅读代码和文档分析gitea功能和模块结构，
2. 对gitea进行需求分析
3. 对gitea实现进行建模
4. 分析gitea实现的不足和缺陷
5. 对发现的不足与缺陷进行测试修改
6. 开发模块：自定义通知特性：仓库状态发生改变时（比如新的PR、issue），发送自定义通知给项目组成员。通知类型可以是微信、邮件
7. 开发模块：GitLab迁移支持：从Gitlab进行的自托管或托管迁移添加了新的迁移接口。在UI上为自托管git服务添加选择

# 3 项目计划

## 3.1 人员分工

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 编号 | 项目确定及介绍 | 工期 | 开始时间 | 完成时间 | 前置任务 | 负责成员 |
| 1 | 个人工作记录 |  | 2020年3月9日 |  |  | 全体成员 |
| 2 | 会议记录及任务调度分配 |  | 2020年3月10日 | 2020年3月20日 |  | 郑锋 |
| 3 | 项目前期调研、资料收集，汇总 |  | 2020年3月10日 | 2020年3月20日 |  | 孙维华 |
| 4 | 项目资料整理、精炼，项目介绍内容撰写 |  | 2020年3月11日 | 2020年3月20日 |  | 麦梓健 |
| 5 | 常见相似框架对比，找出当前框架优缺点 |  | 2020年3月11日 | 2020年3月20日 |  | 王子璇 |
| 6 | 框架本地搭建运营，编写小程序测试 |  | 2020年3月11日 | 2020年3月20日 |  | 洪治凑 |
| 7 | 对Go语言进行学习，实现Git的环境搭建并对组员指导 |  | 2020年3月11日 | 2020年3月20日 |  | 王伟民 |
| 8 | PPT制作 | 1个工作日 | 2020年3月11日 | 2020年3月11日 | 4、5、6 | 麦梓健 |
| 9 | 软件需求分析 | 14个工作日 | 2020年3月14日 | 2020年3月27日 |  |  |
| 10 | 软件需求评审 | 7个工作日 | 2020年3月28日 | 2020年4月3日 | 9 |  |
| 11 | 软件需求复评审 | 7个工作日 | 2020年4月3日 | 2020年4月9日 | 10 |  |
| 12 | 软件产品改进与展示 | 21个工作日 | 2020年4月10日 | 2020年4月30日 | 11 |  |
| 13 | 软件测试分析 | 7个工作日 | 2020年5月7日 | 2020年5月13日 | 12 |  |
| 14 | 软件测试评审 | 7个工作日 | 2020年5月14日 | 2020年5月20日 | 13 |  |
| 15 | 软件测试复评审 | 7个工作日 | 2020年5月21日 | 2020年5月27日 | 14 |  |
| 16 | 软件演示与测评 | 7个工作日 | 2020年5月28日 | 2020年6月3日 | 11、14 |  |
| 17 | 工作量分析与统计分析 | 99个工作日 | 2020年3月9日 | 2020年6月15日 |  |  |
| 18 | 配置管理 | 99个工作日 | 2020年3月9日 | 2020年6月15日 |  |  |
| 19 | 综合实验分析和总结 | 7个工作日 | 2020年6月9日 | 2020年6月15日 | 8、9、10、11 |  |

## 3.2 组织形式

1. 会议：每周1次，进行工作总结汇报和检查
2. 群聊：随时进行，进行工作情况的交流和问题的互助。
3. 使用git管理项目进度
   1. 以 master 作为团队主分支，其上所有commit是团队工作合并后的稳定版本
   2. hotfix 分支用于问题反馈的修复，向 master 和 develop 合并
   3. 组长在develop分支上开始新的实验内容
   4. 各个团队成员通过从develop上创建新的个人分支进行自己的工作
   5. 团队成员需要通过pull request将自己的分支合并到develop分支，团队成员审核后组长操作
   6. hotfix 分支合并完成后应被删除
   7. 团队生成的最终文档通过release发布

## 3.3 人员分工

1. 开发人员：王子璇、孙维华、郑锋
2. 测试人员：洪怡凑、麦梓健
3. 文档人员：王伟民、麦梓健

# 4 关键问题

1. 现在正处于疫情时期，不能面对面交流，线上交流多多少少会存在一些问题。
2. 对项目不够了解和熟悉，不能按预期效果开发，难以保证项目完成。
3. 团队网上协调可能会出现问题，导致无法及时完成任务

# 参考资料

1. <https://docs.gitea.io/en-us/>
2. https://discourse.gitea.io/