配置管理总结

小组一共在Github上有309次提交记录，通过GitInfo工具，将所有提交记录爬取并分析得到如下统计报告。

# 一、总览

## 1.1. 用户提交情况

表 1 B组Github提交记录总览

|  |  |  |  |
| --- | --- | --- | --- |
| 统计类型 | 负责人 | 次数 | 百分比(%) |
| 用户提交情况 | 胡明昊 | 118 | 38 |
| wangxy381 | 48 | 17 |
| OovEver | 97 | 31 |
| liuye | 41 | 13 |
| buaaliuchao | 5 | 1 |
| 关键词提交情况 | 添加 | 80 | 25 |
| 更新 | 77 | 24 |
| 修改 | 39 | 12 |
| merge | 36 | 11 |
| 需求 | 34 | 11 |
| project | 27 | 8 |
| 提交 | 25 | 8 |
| 会议 | 17 | 5 |
| 评审 | 12 | 3 |
| 幻灯片 | 12 | 3 |
| 实验四 | 11 | 3 |
| 实验三 | 10 | 3 |
| 实验八 | 10 | 3 |
| 删除 | 9 | 2 |
| test | 9 | 2 |
| 需求规格说明书 | 8 | 2 |
| readme | 7 | 2 |
| 实验一 | 6 | 1 |
| 实验五 | 6 | 1 |
| 实验七 | 6 | 1 |
| update | 5 | 1 |
| conflict | 4 | 1 |
| PPT | 4 | 1 |
| 课堂 | 3 | 1 |
| keynote | 3 | 1 |
| change | 2 | 1 |
| 评价 | 2 | 1 |
| 计划 | 2 | 1 |
| 恢复 | 1 | 1 |
| 错误 | 1 | 1 |
| 实验二 | 1 | 1 |
| 实验六 | 1 | 1 |

## 1.2. 提交次数随周数变化情况

图 1 提交次数随周数变化情况

图 1是组员提交的次数随周期变化的情况，从图中可以看出，在实验后期组员的提交次数少有下滑，而在需求和测试的部分显得较为活跃。分析认为，需求阶段提交频繁、测试阶段提交频繁的原因在于该阶段零散文件较多，提升了提交的次数。而后期提交的次数变少的原因在于，对于二进制文件的更新需要极为小心不能产生覆盖，因此均在本地完成后由组员统一提交。

## 1.3 提交次数随天数变化情况

图 2 提交次数随天数变化情况

## 1.4 各个关键词提交次数

图 3 各个关键词提交次数

在关键词分析时，我们发现，在所有的关键词中【添加】、【更新】的出现频率最高，可见在平时的实验中，一直都有新的产物出现，并且对原有的产物也一直保持着更新。但仍然可以确认有些产物提交之后并没有修改行为。

在merge中出现了36次说明在同一个分支下出现提交不同步的概率较高，因此之后仍然应该在不同分支下进行提交。conflicat出现了4次，说明对于任务的划分较为合理，之后的提交只需稍稍注意即可。

# 二、实验一至六变更分析

## 2.1 版本变更情况

表 2 实验主要内容各版本变化情况

|  |  |  |  |
| --- | --- | --- | --- |
| 实验内容 | 版本 | 负责人 | 版本内容 |
| 实验一 | 需求规格说明书v1.0 | 胡明昊 | 初稿 |
| 需求规格说明书v1.1 | 汪晓燕 | 完善需求内容 |
| 需求规格说明书v1.2 | 汪晓燕 | 完善需求内容 |
| 需求规格说明书v1.3 | 穆鹏飞 | 修改文档错误 |
| 需求规格说明书v1.4 | 穆鹏飞 | 修改文档错误 |
| 需求规格说明书v1.5 | B组全体 | 更改非功能性需求，对刘超老师批注进行修改，对DE组批注进行修改 |
| 需求规格说明书v1.6 | 胡明昊 | 补充改进部分内容 |
| 需求规格说明书v1.7 | 穆鹏飞 | 增加改进模块需求分析 |
| 实验二 | B-A\_3\_30评审汇总 | B组全体 | 当期实验内容评审汇总 |
| B-A\_4\_6评审汇总 | B组全体 |
| B-A\_4\_25评审汇总 | B组全体 |
| B-G\_3\_20评审汇总 | B组全体 |
| B-G\_4\_6评审汇总 | B组全体 |
| B-G\_4\_20评审汇总 | B组全体 |
| B-G\_4\_25评审汇总 | B组全体 |
| CD组\_04\_09\_软件评审问题汇总  CD组提交 | B组全体 |
| 刘超老师\_5\_2\_评注汇总 | 刘超老师 |
| 实验三 | 改进方案和设计v0.1 | 胡明昊 | 改进方案的初步设想 |
| 测试模块需求分析文档v0.1 | 胡明昊 | 改进模块的较完整分析 |
| 实验四 | 软件测试说明书\_v1.0 | B组全体 | 初稿 |
| 软件测试说明书\_v1.1 | 汪晓燕 | 添加测试用例 |
| 软件测试说明书\_v1.2 | 刘晔 | 添加测试结果 |
| 软件测试需求说明书 | B组全体 | 描述测试总体内容 |
| Web压力测试工具http\_load | 刘晔 | 描述Blade压力测试工具和结果 |
| 实验五 | 软件测试需求及测试用例汇总表 | B组全体 | 测试需求与测试用例对照表 |
| 软件需求汇总表 | B组全体 | 软件需求汇总 |
| 设计实现汇总表 | B组全体 | 设计实现汇总 |
| 软件测试覆盖表 | B组全体 | 测试与测试点覆盖汇总表 |
| 术语表 | 汪晓燕 | 测试中涉及术语汇总 |
| B-A测试评审 | B组全体 | 对A组测试的评审 |
| B-G测试评审 | B组全体 | 对G组测试的评审 |
| 对A组测试 | B组全体 | 对A组测试的结论 |
| 实验六 | Project管理第一次 | 穆鹏飞 | 调整项目计划 |
| Project管理第二次 | 穆鹏飞 |
| Project管理第三次 | 穆鹏飞 |
| Project管理第四次 | 穆鹏飞 |
| Project管理第五次 | 穆鹏飞 |

# 三、Git管理自评

## 3.1 提交管理

### 3.1.1 通过分支并行开发

小组成员均能够熟练的使用Git进行配置管理，在日常实验中，各小组成员均采取独自分支进行开发。分支是Git等版本控制工具的特色功能，能够保证任务以足够的灵活性进行并行处理，因此小组在各自分支上对各自所属的文件进行处理，并在周五或需要时对分支进行合并，使得主分支上的文件始终处于有效、稳定的状态。

### 3.1.2 提交规范

1）划分文件归属

在配置管理过程中，我们对提交进行了规范，要求不允许同时编辑同一个文件，在拉取、合并之前需要相互通知和声明。若存在冲突，需要告知冲突文件所属成员以决定保存的版本。在实验中所涉及的文件大多为二进制文件，这对Git来说并不友好，因为该类文件无法做到字节级别的合并，只能由新版本覆盖上一个版本，因此分割文件归属是避免冲突的最有效办法。

2）提交描述的限制

在配置管理中，提交描述是分析提交变化内容的主要方式之一。reviewer希望能够通过每次提交的描述即可分析版本、功能变化情况，而不是一定要从代码中检查版本变更内容。对于实验七的内容小组做了分析，认为提供一个描述规范能够更轻易的分离提交和分析版本变化。

根据小组提交的描述和其他组提交的描述，我们给出如表 3所示高频动词。

表 3 动词规范

|  |  |  |
| --- | --- | --- |
| 关键词 | 词性 | 描述 |
| 添加 | 动词 | 添加文件和组件 |
| 更新 | 动词 | 在已有内容基础上增加内容 |
| 修改 | 动词 | 修改内容 |
| 提交 | 动词 | 汇总更新 |
| 恢复 | 动词 | 重新放入之前提交过的内容 |
| 删除 | 动词 | 删除内容或文件 |
| update | 动词 | 同更新 |
| change | 动词 | 同修改 |
| conflict | 动词 | 冲突及解决 |
| test | 动词 | 测试 |
| merge | 动词 | 合并分支 |

除此之外，小组给出有效语义助词用于标记特定内容，例如文件类型、更新重要性等词汇。

表 4 助词规范

|  |  |  |
| --- | --- | --- |
| 关键词 | 词性 | 描述 |
| 评审 | 动词 | 评审内容相关 |
| 需求规格说明书 | 名词 | 需求文档相关 |
| 参考资料 | 名词 | 参考资料相关 |
| 评价 | 名词 | 同评审 |
| 课堂 | 名词 | 课堂内容整理 |
| readme | 名词 | 项目介绍 |
| mpp | 名词 | project文件相关 |
| PPT | 名词 | ppt内容 |
| 幻灯片 | 名词 | 同ppt内容 |
| keynote | 名词 | 同ppt内容 |
| project | 名词 | 同mpp |
| 需求 | 名词 | 需求相关 |
| 错误 | 名词 | 修正错误或发现错误 |
| 计划 | 名词 | 项目计划相关 |
| 有效 | 形容词 | 表示里程碑提交 |
| 重要 | 形容词 | 描述重要修改 |
| 微调 | 形容词 | 描述轻微修改 |
| 实验一~八 | 名词 | 表示提交涉及的实验归属 |
| 会议 | 名词 | 表示会议内容相关 |

## 3.2 Git版本标识

本组的版本标识共分为两类：

* 提交注释

采用上节所示的关键词进行标记，表示是对文档的更新、添加或整理来表达文档的版本变化情况。

* 提交文档

本组对不同的文档命名方式略有不同。

1）预见会存在多版本的文档：采用后缀vX.Y的形式递增版本，若没有结构上发生变化，仅更新内容则仅变更小版本号，若发生结构性的变动，则更新大版本号。

2）与日期相关的文档：评审文件与评审日期相关度较大，因此对于评审文档采用后缀日期的形式进行版本标识。

3）与前一版本关系不大的文档：项目文档mpp文件、会议记录文件，每次版本均与上次内容关系较小，因此采用第X版的形式进行编号。

## 3.5 优点分析

1）较为准确的行为记录

在提交的过程中，我们小组使用了助记词汇进行提交注释，在提交后，能够较为准确的分析组员提交的内容、动作和所属实验。

2）较为完善的项目描述

在Github主页的Readme文档中，我们小组描述了系统的基本原理、Blade的官方主页、项目涉及的文件类型的说明以及工具的使用说明。

除此之外，小组在Readme文档中提供了所有的文件的路径展示，因此，在查看中可以直接搜索到需要查看的文件所属的文件夹。

3）工具支持

采用了工具GithubInfo辅助分析提交信息，能够快速的分辨出组员的提交行为，并对提交的行为按日期、周数的分布情况进行的整理，有助于分析组员的提交行为。

## 3.6 缺点分析

1）提交与文件不能完全对应

我们在分析的过程中发现，提交了很多次，但文件、标题并不能反映出内容的更新。例如，实验六的项目计划文件的更新，在系统内记录有27次提交，但文件仅有5个版本。因此在版本管理中，虽然能够随时恢复至提交前后的状态，但无法直观的获得提交版本的更新、内容的变化。应当单次提交只提交一个文档，并注明文档更新的内容，更新前后文件名变化和版本号变化。

2）文件归属的不统一

我们对于一个系列的文件采用了文件夹的形式存放，例如对于测试覆盖表，我们在实验文件夹下建立了新的文件夹用于存放，能够保证一次性浏览全部文件，也不至于因为文件名排序而混杂到其他文件中。但对于评审文档来说，我们采用文件夹的形式散落在各个实验文件夹下，虽然对应至各个实验的评审，但不能够一次性获得所有的评审文档，显得稍有紊乱。应当建立总的评审文件夹，下分实验1~8文件夹存放评审文档。