图片包含 游戏机, 物体, 钟表, 画

描述已自动生成

![卡通人物

描述已自动生成]()

软件工程综合实验

配置管理计划

WeChatterBot

|  |  |
| --- | --- |
| 团队编号 | E 组 |
| 团队成员 | 郭维泽 李书缘 吕江枫 |
|  | 王元玮 王云杰 张延钊 |

**Version 1.1**

版本变更历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本 | 变更时间 | 主要编制人 | 备注 | 审核人 |
| 1.0 | 2020/03/23 | 吕江枫 | 初稿 | 郭维泽，李书缘，王元玮，王云杰，张延钊 |
| 1.1 | 2020/04/01 | 吕江枫 | 调整目录结构。修改章节1，2，3中部分内容。 | 郭维泽，李书缘，王元玮，王云杰，张延钊 |

目录

[版本变更历史 2](#_Toc36766602)

[1. 配置管理说明 4](#_Toc36766603)

[2. Github配置管理 4](#_Toc36766604)

[2.1 目录结构 4](#_Toc36766605)

[2.2 分支使用 5](#_Toc36766606)

[2.3 文件命名规范 5](#_Toc36766607)

[2.4 提交信息规范 6](#_Toc36766608)

[2.5 文件提交 6](#_Toc36766609)

[3. OneDrive配置管理 6](#_Toc36766610)

[3.1 目录结构 6](#_Toc36766611)

[3.2 文档编制要求 6](#_Toc36766612)

# 配置管理说明

配置管理（Configuration Management，CM）是通过技术或行政手段对软件产品及其开发过程和生命周期进行控制、规范的一系列措施。本小组主要使用GitHub作为项目的配置管理工具，来记录软件产品的演化过程，同时完成实验文档的提交。由于本次软件工程综合实验主要以各阶段实验过程文档作为实验结果，而GitHub不适合文档的配置管理，因此，本小组采用OneDrive对实验中的各阶段文档进行配置管理，在OneDrive中编制完成可提交的实验文档后，再统一提交至GitHub。

本文章节结构如下：第二章介绍了本小组如何使用GitHub进行项目总体的配置管理。第三章介绍了本小组如何使用OneDrive进行文档的配置管理。

# Github配置管理

## 目录结构

小组成员向GitHub的成果提交须严格遵守以后目录结构和要求。不得未经许可在根目录下创建新目录或将文件归类于错误的目录下。本组GitHub目录结构如下：

* 0-参考资料：用于存放必要的参考资料，如原项目说明文档、参考教材等
* 1-会议记录：存放每一周会议的记录文件
* 2-课堂报告：存放每周用于课堂报告的PPT文件或其他展示所用的资料
* 3-项目源码：存放WeChatterBot源代码（使用GitHub Submodule链接）
* 4-项目计划：存放项目计划书
* 实验1：软件需求分析：存放软件需求分析相关资料
* 实验2：软件需求评审：存放软件需求评审相关资料
* 实验3：软件产品改进与展示：存放软件产品改进与展示相关资料
* 实验4：软件测试：存放软件测试相关资料
* 实验5：软件测试评审：存放软件测试评审相关资料
* 实验6：软件进度计划与控制：存放软件进度计划与控制相关资料
* 实验7：配置管理：存放配置管理相关资料
* 实验8：工作量估计与统计分析：存放工作量估计与统计分析相关资料

## 2.2 分支使用

小组成员均能够熟练的使用Git进行配置管理，在实验过程中，实验文档及其补充资料的提交由每周选定组员统一提交于主仓库20\_E/master分支。

在代码开发中，本项目将代码仓库设置于WeChatterBot代码仓库中，它是GitHub上新建的个人项目，添加了所有组员为开发者，WeChatterBot作为submodule被连接进20\_E，使得开发过程中的修改与分支管理被封装于WeChatterBot代码仓库中，而主仓库20\_E可随时更新获得WeChatterBot中的最新内容。

在WeChatterBot的开发中，每个功能一个分支，多人完成一个功能，从master分支上的基础上进行开发。在分支开发完成之后，组员提交Pull Request，由除开发者之外至少一名组员进行代码评审，合并到主分支。

## 2.3 文件命名规范

1. 实验报告文件

命名规范：E\_document\_Vx.y.z.ext

例子：E\_软件需求分析说明书\_V1.0.0.docx

修订号更新规则：只做了向下兼容的修正时才递增，即针对不正确的内容进行内部修改。

次版本号更新规则：在有新功能加入或者功能被改进时递增。递增时必须整合之前所有的修订号改动，并将修订号归零。

主版本号更新规则：整合所有当前的次版本改动，并经过确认后，更新一个主版本，并将次版本号和修订号归零。

1. 补充文件

命名规范： filename.ext

例子：UML模型.mdj；第一周展示.ppt

1. 代码文件

命名规范：filename.ext，文件名为英文单词或单词组合，若文件名为单词组合时，使用下划线相连。

例子：response\_selection.py

## 2.4 提交信息规范

命名规范：month.day\_document\_description

例子：3.25\_软件需求分析说明书\_完成概述部分

## 2.5 文件提交

* 实验文档提交：只集中提交每周所要展示的最终版文档，中间结果一律在OneDrive处理。（按文件夹进行提交，一次提交一个文件夹中的实验文档和相关补充文件）
* 补充文件提交：提交本周文档时一同提交与该文档相关的全部补充资料，课堂展示用ppt和会议记录等按文件夹进行提交。
* 代码提交：个人代码提交全部提交到对应的功能分支，在功能完成开发内容和测试后通过贡献请求合并到主分支。

# OneDrive配置管理

## 3.1 目录结构

小组成员向OneDrive的成果提交须严格遵守以后目录结构和要求。本组OneDrive目录结构与上文GitHub目录结构相同，用于存放对应的文档文件和补充文件。

## 3.2 文档编制要求

* 文件命名规范：同GitHub配置管理中的文件命名规范。
* 文件提交：小组成员向OneDrive的文件提交、修改不需要约定提交顺序或要求。