iOS小组Git工作流

目的：良好的git工作流可以帮助我们解决项目中多个版本并行迭代冲突、规范开发流程、codereview。

方案：gitflow＋merge request

目前只能使用公司gitlab，本着利用最大价值的目的，采用gitflow＋merge requst的方式进行代码的版本管理，借助merge request进行codereview。

Git 基础

SSH key

SSH key 可以让你在你的电脑和 GitLab之间建立安全的加密连接。

你可以按如下命令来生成sshkey

```
// Creates a new ssh key using the provided email
ssh-keygen -t rsa -C "xxxxx@xxxxx.com"
# Generating public/private rsa key pair...
查看你的public key，并把他添加到 GitLab
cat ~/.ssh/id_rsa.pub
# ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC6eNtGpNGwstc...
```

添加后，在终端（Terminal）中输入 `sh ssh -T git@git.zhubajie.la` 若返回Welcome to GitLab, yourname!则证明添加成功。

添加成功即可通过ssh key 访问GitLab

关于权限

访客

对于公有项目： - 创建issue

- 评论
- Clone 和 Pull 项目
- 打包下载代码

- Fork 项目
- 创建 merge request

报告者

- 继承访客的权限
- 私有项目：不能查看代码
- 私有项目：不能下载代码
- 私有项目：不能fork代码

观察者

- 继承报告者权限
- 创建wiki
- 打包下载代码
- 不能push代码
- 私有项目：可以fork

开发者

- 创建 issue
- 评论
- Clone 和 Pull 项目
- 打包下载代码
- 创建 pull request
- 创建分支
- 推送分支
- 删除分支
- 创建标签（里程碑）
- 创建 wiki

管理员

- 创建 issue
- 评论
- Clone 和 Pull 项目
- 打包下载代码
- 创建 pull request
- 创建分支
- 推送分支

- 删除分支
- 创建标签（里程碑）
- 创建 wiki
- 添加项目成员
- 强制推送分支
- 编辑项目属性
- 项目组管理员
- 编辑项目组属性
- 增加成员
- 添加 / 删除项目
- 设置项目组管理员
- 删除项目组
- 更改成员项目权限

常用几本命令

1. git init
2. git add .
3. git push
4. git pull
5. git commit -a
6. git commit -m
7. git reset
8. git revert
9. git status
10. git log
11. git push
12. git merge
13. git rebase

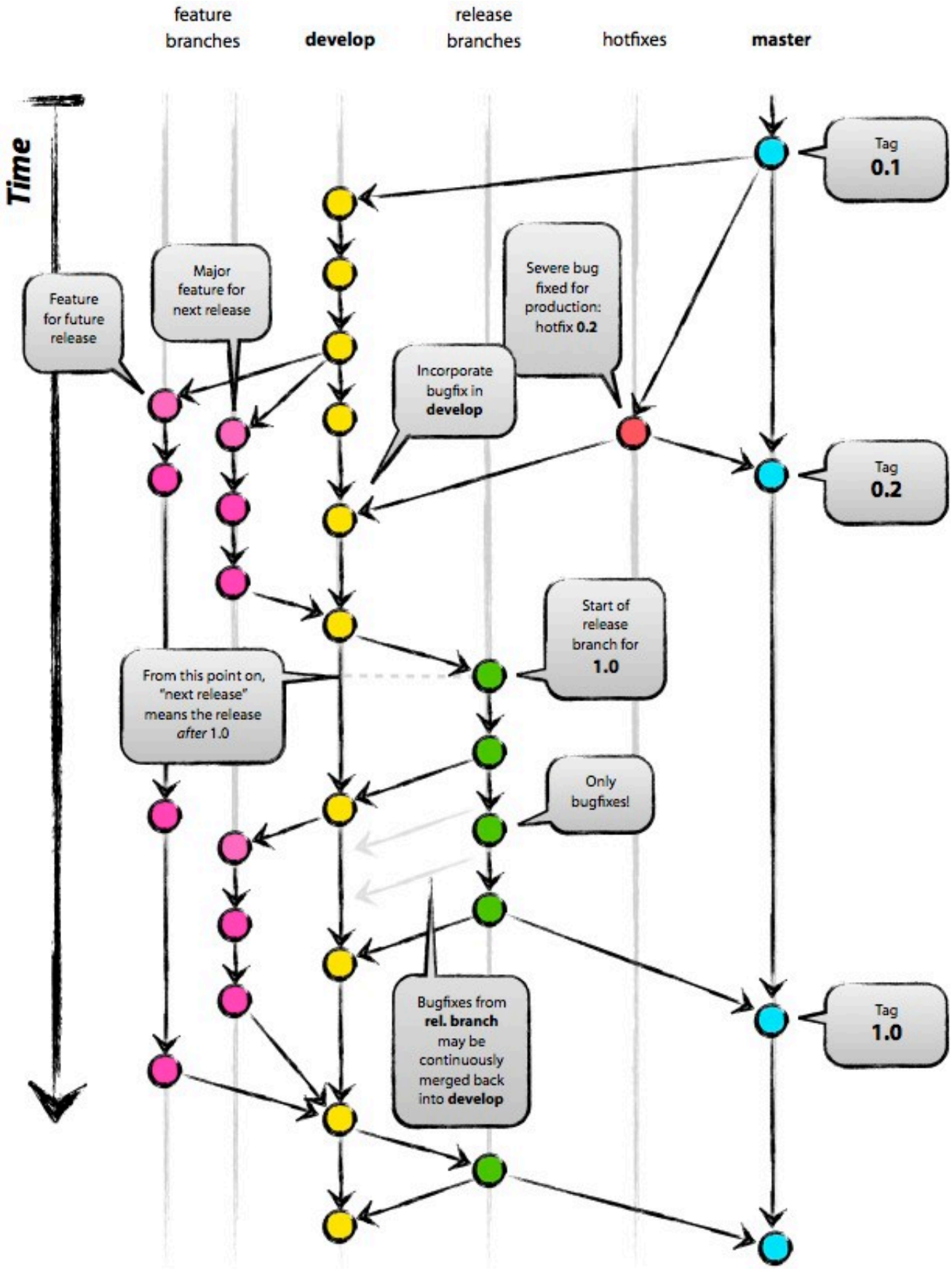
Git常用操作

GUI工具－Tower

GUI工具－SourceTree

Git Flow

Reference:原版Git Flow 、中文版 Git Flow是 一套使用git管理源代码进行软件开发的行为规范最佳实践模

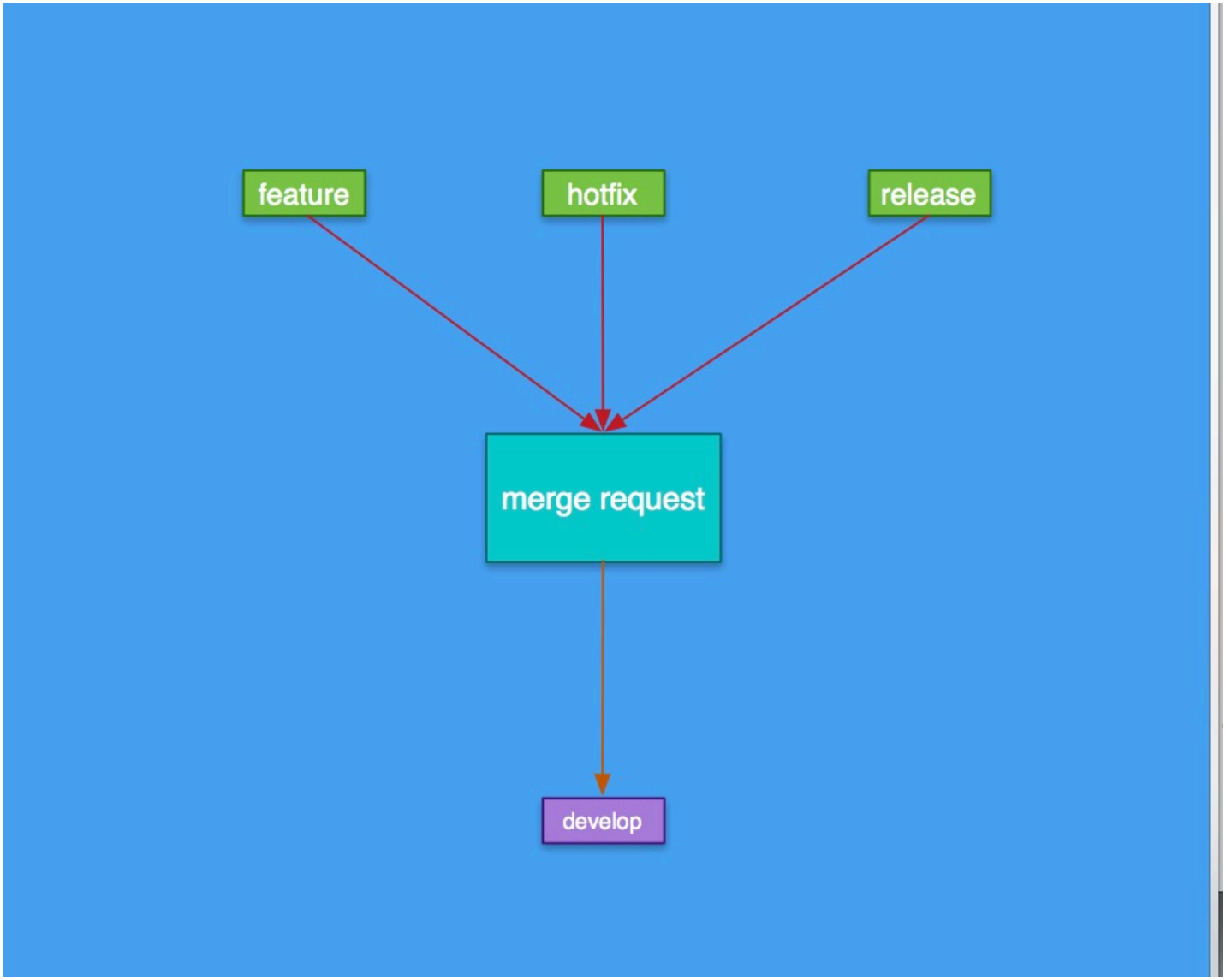


型。

Merge Request-Pull Request

Reference：Merge Request Pull/Merge Request并不是为了替代任何基于Git的协作工作流，而是它们的一

个便利的补充，让团队成员间的协作更轻松方便。主要是便于codereview

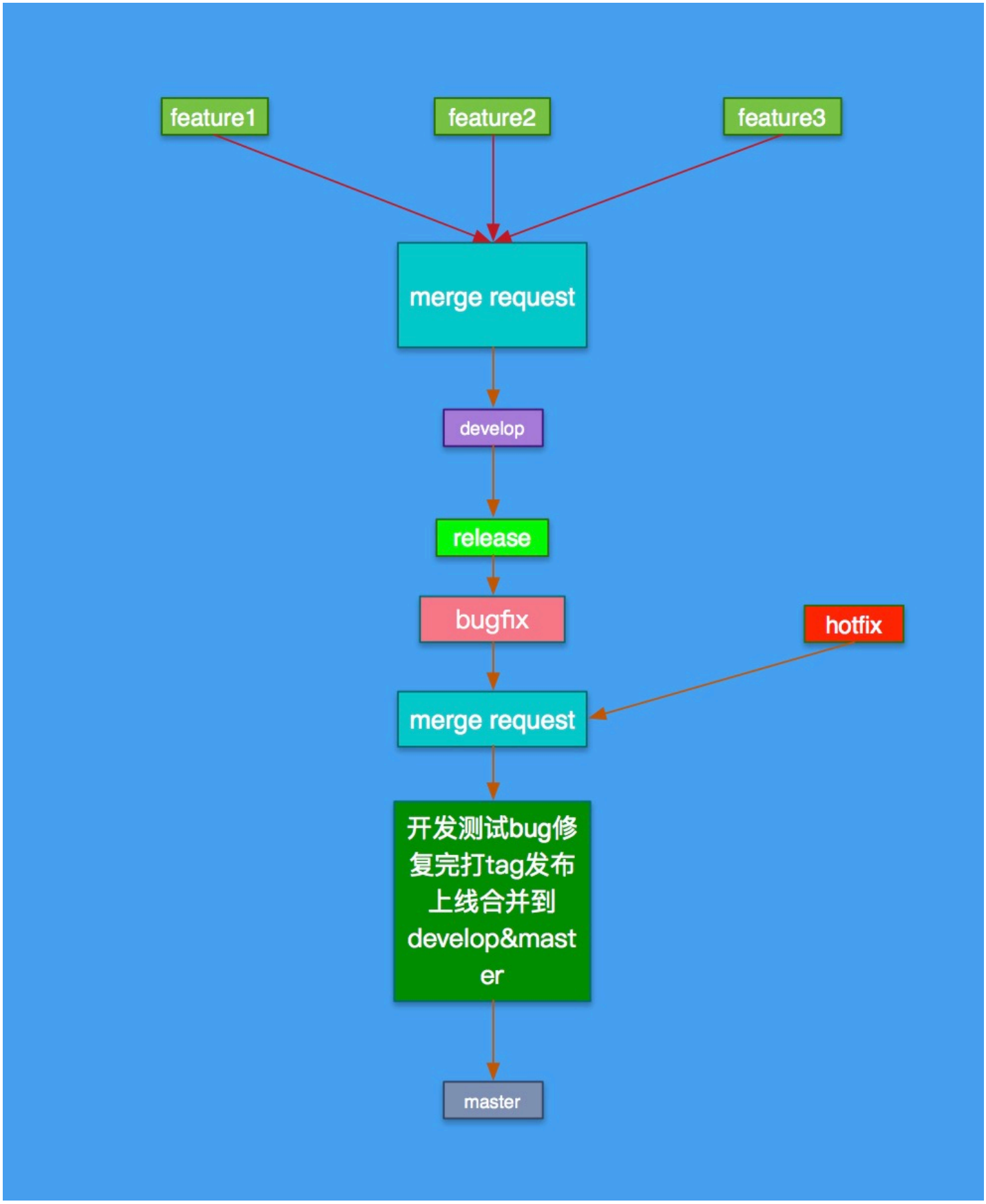


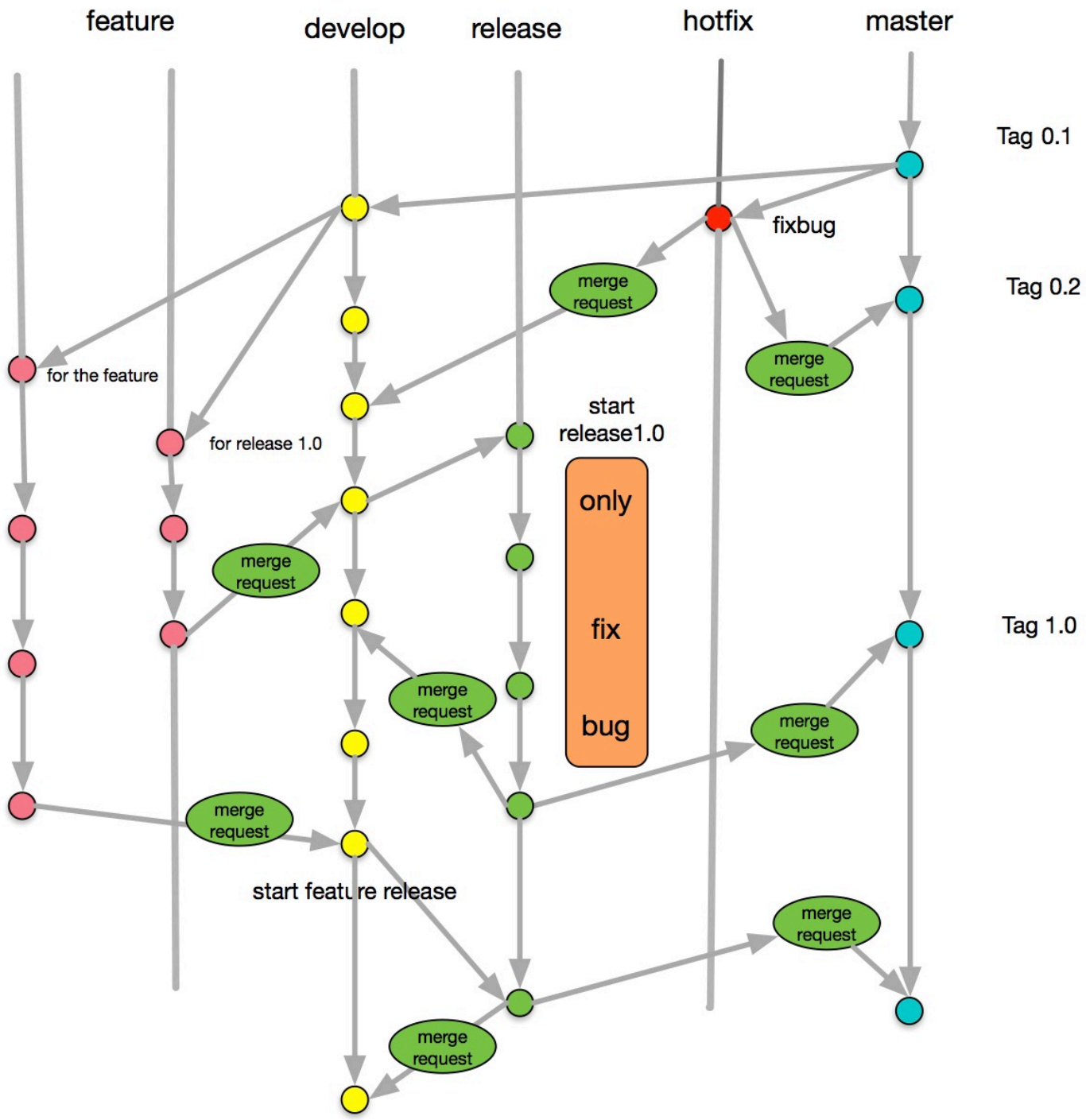
Git Flow ＋ Merge Request

在原有git flow 工作流中每个合并到develop或者master 的环节插入merge request的环节。

实际应用场景：

单个版本迭代

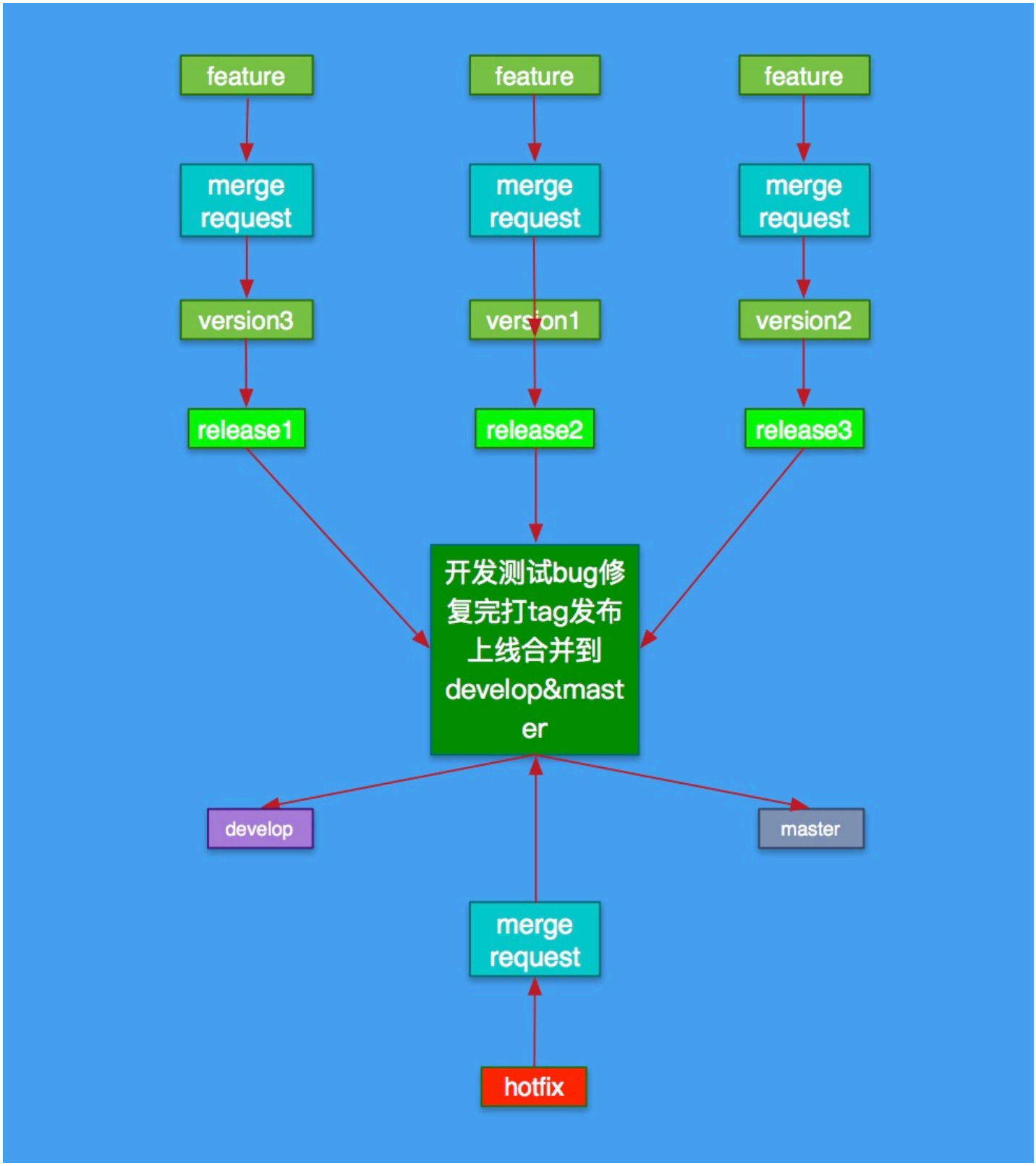

feature1
feature2
feature3
merge request
develop
release
bugfix
hotfix
merge request
开发测试bug修复完打tag发布上线合并到develop&master
master



1. 所有Task为单独featrue分支(包含bugfix)－完全符合gitflow工作流
2. 开发阶段所有Task 的 feture分支 start point为develop分支
3. 从develop分支中pull 最新代码。
4. featreu开发完成，自测完成，从develop分支中pull最新代码。查看时候否有冲突，有冲突解决，然后提交mergerequest。
5. feature分支 mergerequest通过之后合并代码到develop中（目标分支为develop分支）
6. 所有Task feature开发完成，需求完结，,开始release分支。release 分支starpoint 为develop分支
7. release只进行bug修复。跟develop一样，每个bugfix Task 为一个feature 分支。
8. bugfix Task feature 完成提交merge request（目标分支为release）

9. merge request 通过合并代码到release分支。
10. 所有release 分支bugfix Task feature完成。finish release 合并代码到develop&master

多版本并行



在单版本迭代的基础上。开始每个version迭代分支。start point分支为develop分支。

每个版本的工作流跟单个版本迭代一样。

核心概念：

source branch：个人工作分支，只能从 `target branch` pull代码

target branch：版本分支

所有 `target branch` 不允许团队开发人员直接push代码，所有 `source branch` 只能以提交mergre request 的方式合并代码到目标分支。其他所有流程根gitflow工作流一致。