如果多个版本频繁并行，此时完全参照GitFlow是不现实的853769620

develop分支里会混合多个版本的代码，同时在进行集成测试，如果一个版本先测试完，要先进入QA测试环节，是不可以直接基于develop分支去拉release分支的，因为release分支会混合多个版本的代码一块儿测试

基于GitFlow工作流去一点点改进

整个依赖的稳定的和基准的分支只有一个，就是master分支，全部以master分支为基准和基础

（1）比如说启动一个版本

v1.0，涉及3个功能，投入了3个RD（resourch developer，研发工程师），直接从master分支拉3个feature分支下来，不是从develop分支拉了，也没有develop分支了；

同时要做一个v1.1，涉及5个功能，投入了5个RD去开发，直接从master分支拉5个feature下来，每个人就基于自己的feature去开发，搞

同时的话呢，每个版本，刚开始启动，除了拉一堆feature分支，还需要同时从master分支拉一个develop分支下来，专门用于这个版本的集成测试

（2）一个版本ready之后，要进入集成测试了

很多时候，v1.1.0这样的版本，会比v1.0.0版本先上线

经历过很多个大型互联网公司不同时期的项目，稳定时期的项目，我也做过

（1）微型项目，集中式工作流

（2）小型项目，功能分支工作流

（3）中小型，中大型项目，维护，版本迭代缓慢，基于控制，一个版本一个版本的迭代，GitFlow工作流

快速发展的业务和项目，不断的招人，不断的进新人，不断的开启各种版本，经常出现v1.5.3先上线了，但是v1.2.0还没上线

基准代码只有master

feature和develop都从master拉取，以master为准

每个版本的RD各自迭代，各自独立开发，独立测试，加速开发和测试的进度

直接跟master分支拉出来的staging分支去合并，release分支跟master分支的代码，以staging分支为基础进行合并，到这一步才进行多个版本之间的代码集成

到这一步，master代码一定是稳定的，即使master分支混合了别的版本的代码，但是代码基本上都已经趋向于稳定了，所以此时进行多个版本的代码集成，风险是比较低的

staging代码，回归测试，QA仔细回归一遍，要把集成在一起的多个版本的代码对应的功能，全部回归测试，确保每个版本的代码对应的功能都正常运行

staging代码和master分支合并在一起，打tag，准备上线，这个版本就实现了上线

如果在前期，develop分支就开始进行跨版本，跨团队的代码集成，就是一场噩梦