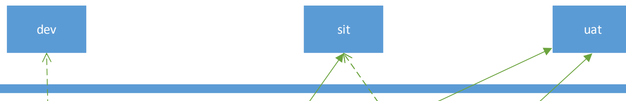
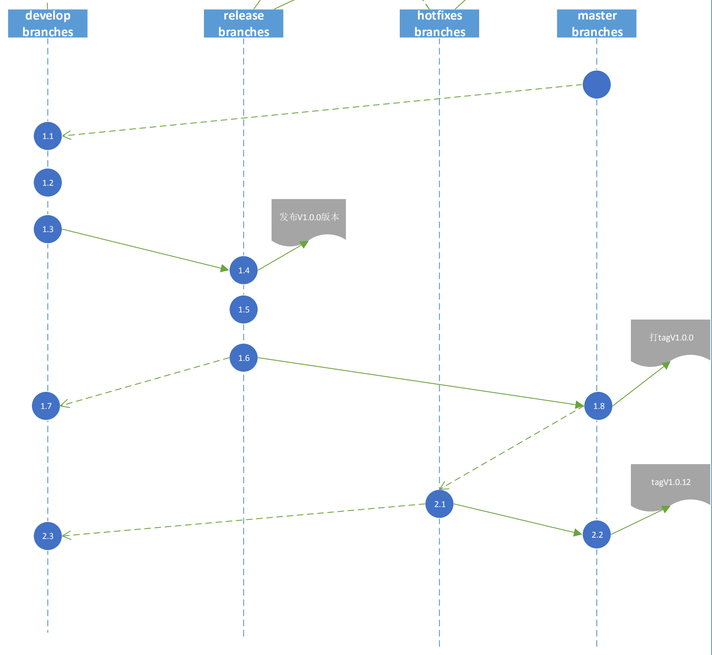
应用服务

1 dev环境用于研发内部问题验证，提供公共的开发数据库（研发人员本地不搭建数据库）。该环境部署使用的代码来自git远程仓库的develop分支。该环境由研发内部维护。
2 sit环境用于研发内部冒烟测试。该环境部署使用的代码来自Git远程仓库的release分支或者hotfixes分支。该环境有研发内部维护。
3 uat环境用于测试部门的冒烟测试和用例测试。该环境部署使用的代码来自Git远程仓库的master分支。考虑到运维部门目前采用的SVN，源码存放采用的目录结构也存在sit和uat。我们只需将Git远程仓库的release分支上传到SVN的sit目录下，剩余其他工作是由运维完成。该环境由运维部门维护。

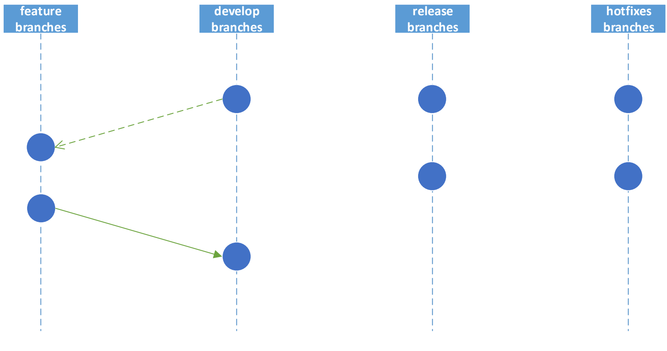


Git远程仓库



1 Git远程仓库分为master、develop、release、hotfixes四个分支。
2 master主分支只用来分布重大版本，日常开发应该在另一条分支上完成。我们把开发用的分支，叫做develop。
3 release是预发布分支，它是指发布正式版本之前（即合并到Master分支之前），我们可能需要有一个预发布的版本进行测试。预发布分支是从Develop分支上面分出来的，预发布结束以后，必须合并进Develop和Master分支。它的命名，可以采用release-*的形式。某个迭代周期完成编码和研发内部测试后即可创建release分支，使用该分支部署到uat，测试过程中发现的bug也是在该分支上完成修复。测试阶段，开发人员任然可以在develop分支上开发新的功能，互不影响。待测试人员完成测试后再切换到release分支修改bug。
4 hotfixes是修补bug分支。软件正式发布以后，难免会出现bug。这时就需要创建一个分支，进行bug修补。修补bug分支是从Master分支上面分出来的。修补结束以后，再合并进Master和Develop分支。它的命名，可以采用fixbug-*的形式。

Git本地仓库



1 Git本地仓库分为develop、release、hotfixes、feature四个分支。
2 本地仓库的release分支是从git远程仓库的release分支clone到本地，在该分支进行变更后，更新内容也只push到git远程仓库的release分支。本地仓库的release分支和本地仓库的其他分支不存在关系。该分支用于预发布阶段（测试阶段）的问题修复。版本正式发布后，可以删除该分支。
3 本地仓库的hotfixes分支是从git远程仓库的hotfixes分支clone到本地，在该分支进行变更后，更新内容也只push到git远程仓库的hotfixes分支。本地仓库的hotfixes分支和本地仓库的其他分支不存在关系。该分支用于正式环境的问题修复。正式环境问题修复后，可以删除该分支。
4 feature分支只存在本地仓库，远程仓库中不存在该分支。该分支是从本地develop分支上面分出来的，开发人员可以在feature分支上开发新的功能，完成开发和内部单元测试后再合并到本地的develop分支。这样做的目的是保持本地develop分支是可以正常运行的。如果没有feature情况下，所有功能都在develop分支上开发，当功能还未开发完成就push到远程服务器，有可能导致develop分支无法正常运行。