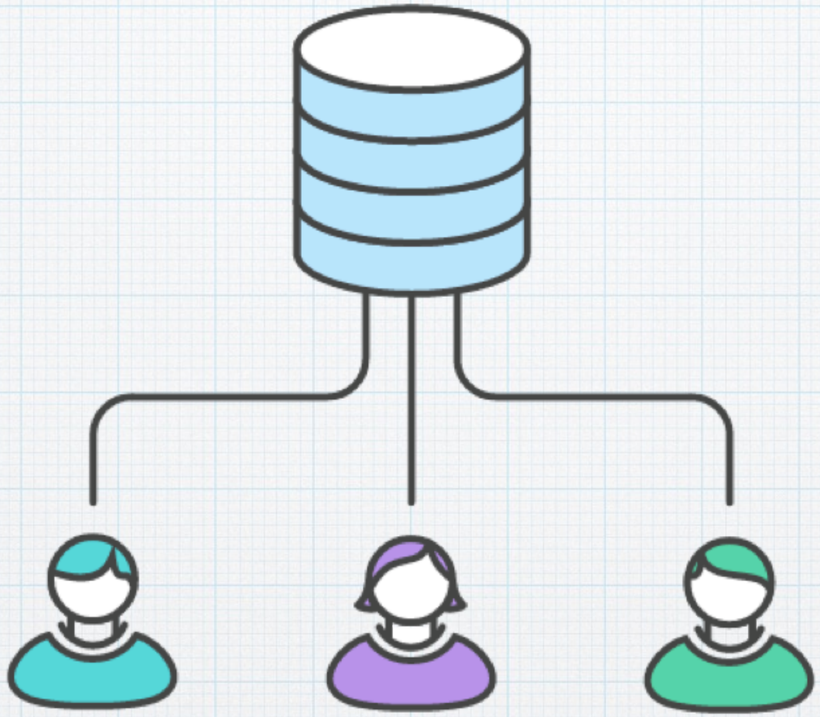
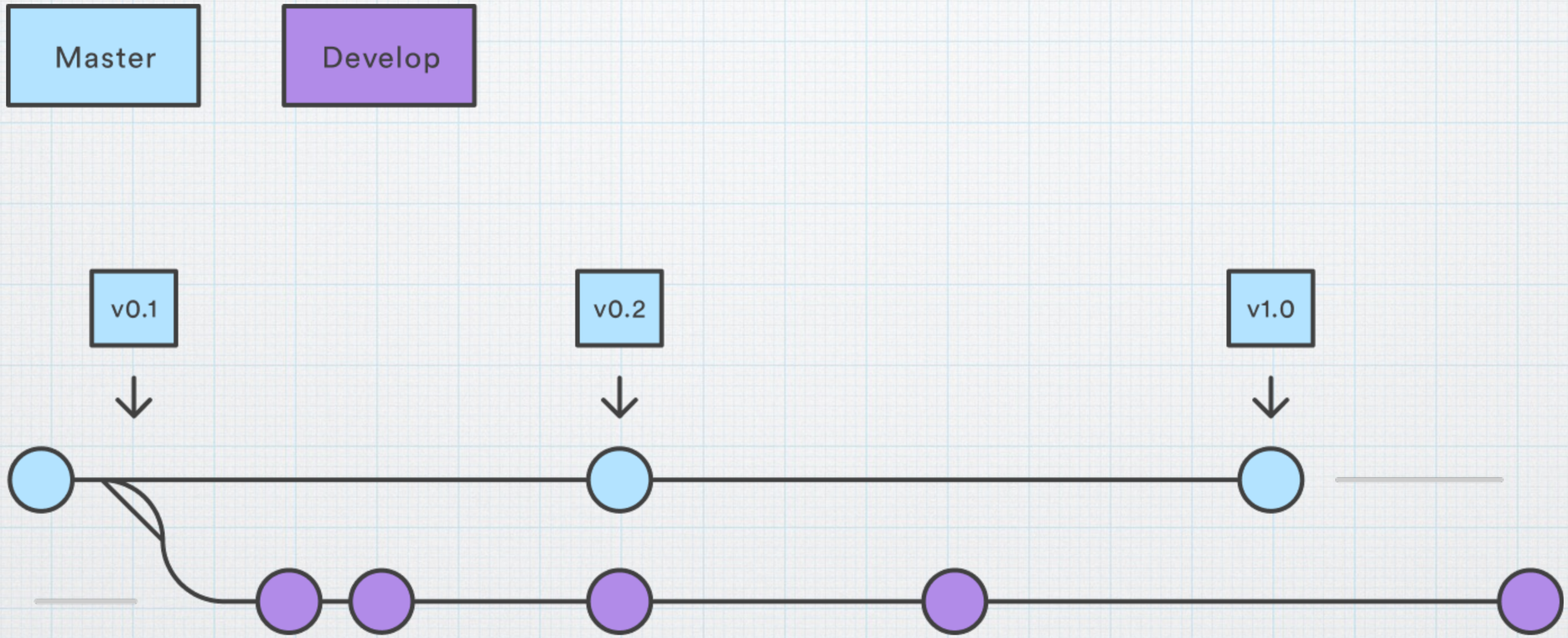
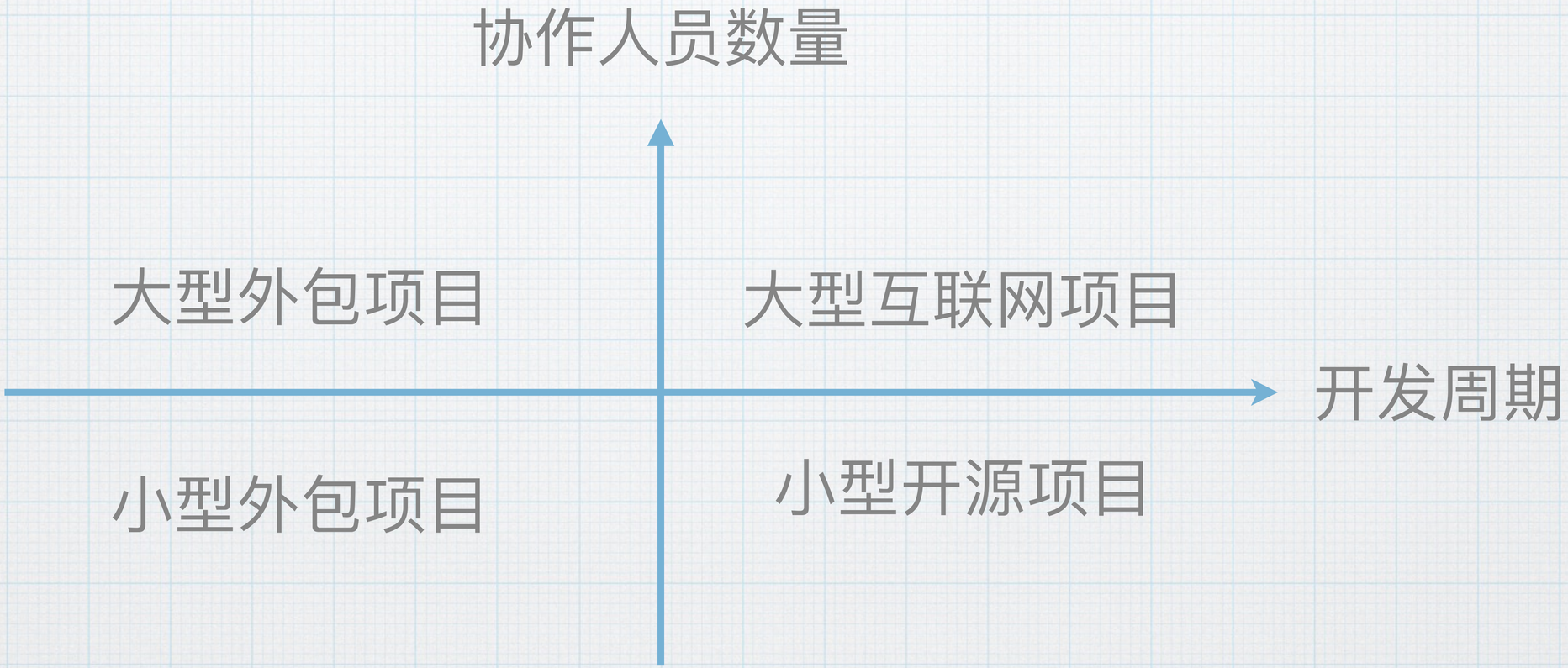


场景式解读 Git 工作流

Git Working Flow



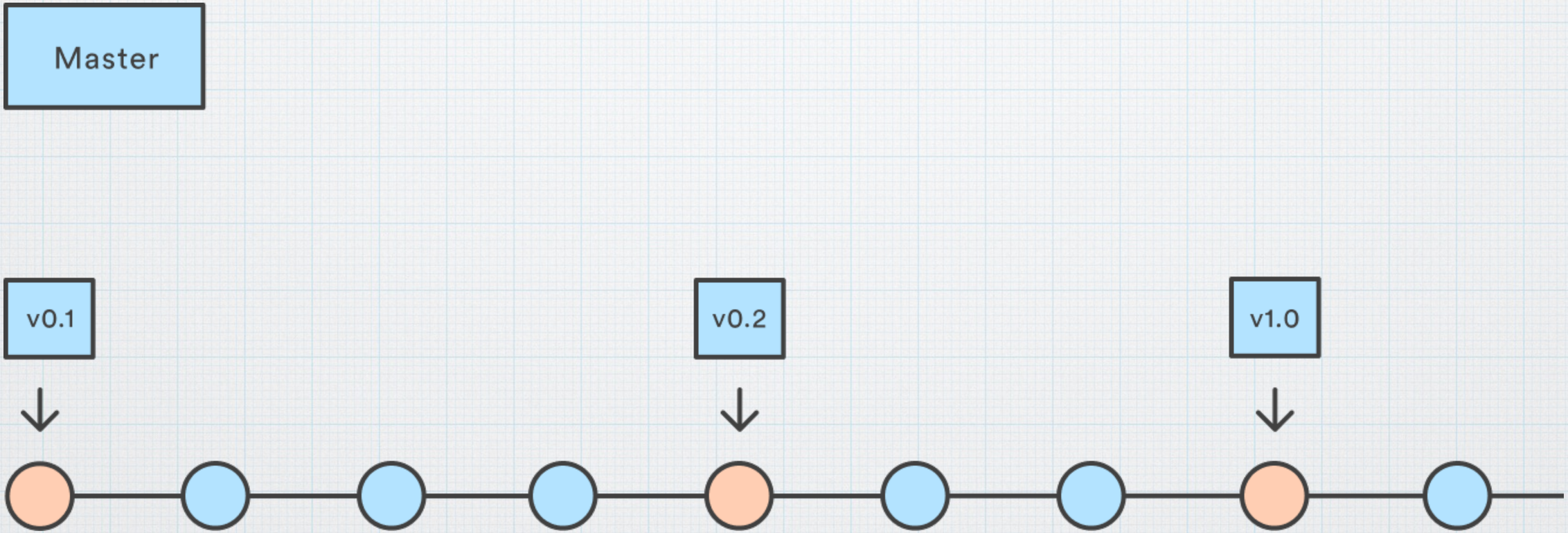
场景式解读



小型外包项目

* 项目周期短
* 团队沟通方便
* Code Review 需求不强

小型外包项目

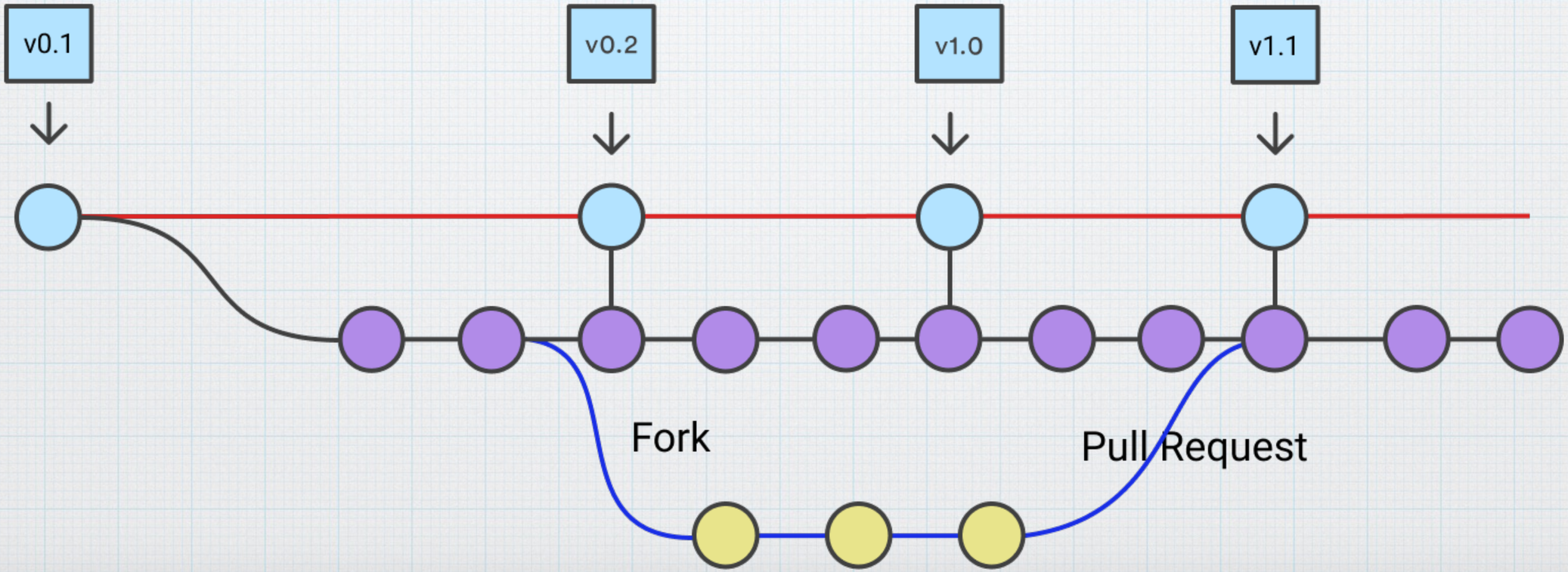


小型开源项目

* 项目周期长
* 要接受第三方贡献
* Code Review 需求强烈
* 可能有维护多个版本需求

小型开源项目

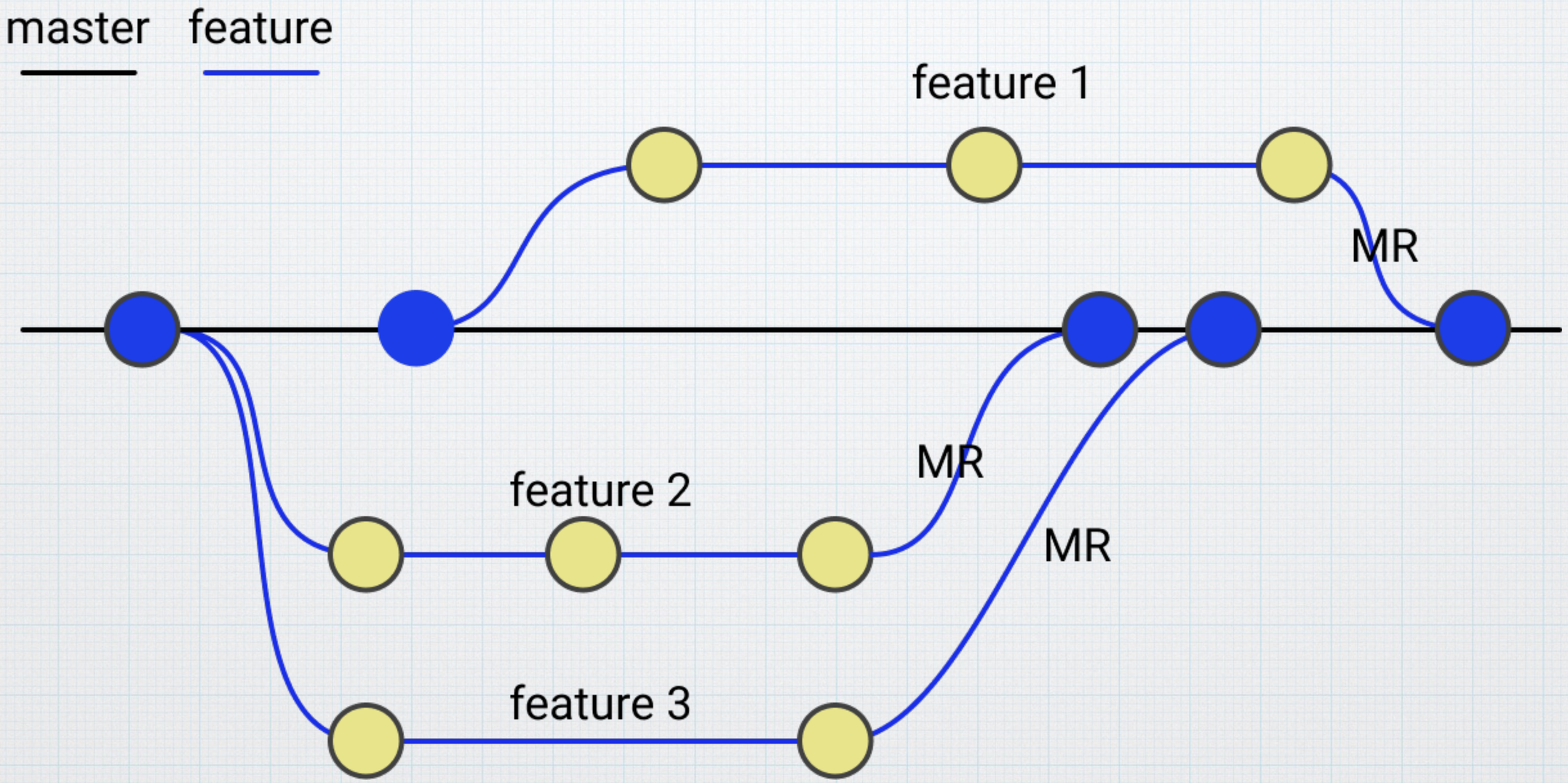
master contributor release



大型外包项目

* 开发人员多
* 分模块并行开发
* Code Review 需求强烈

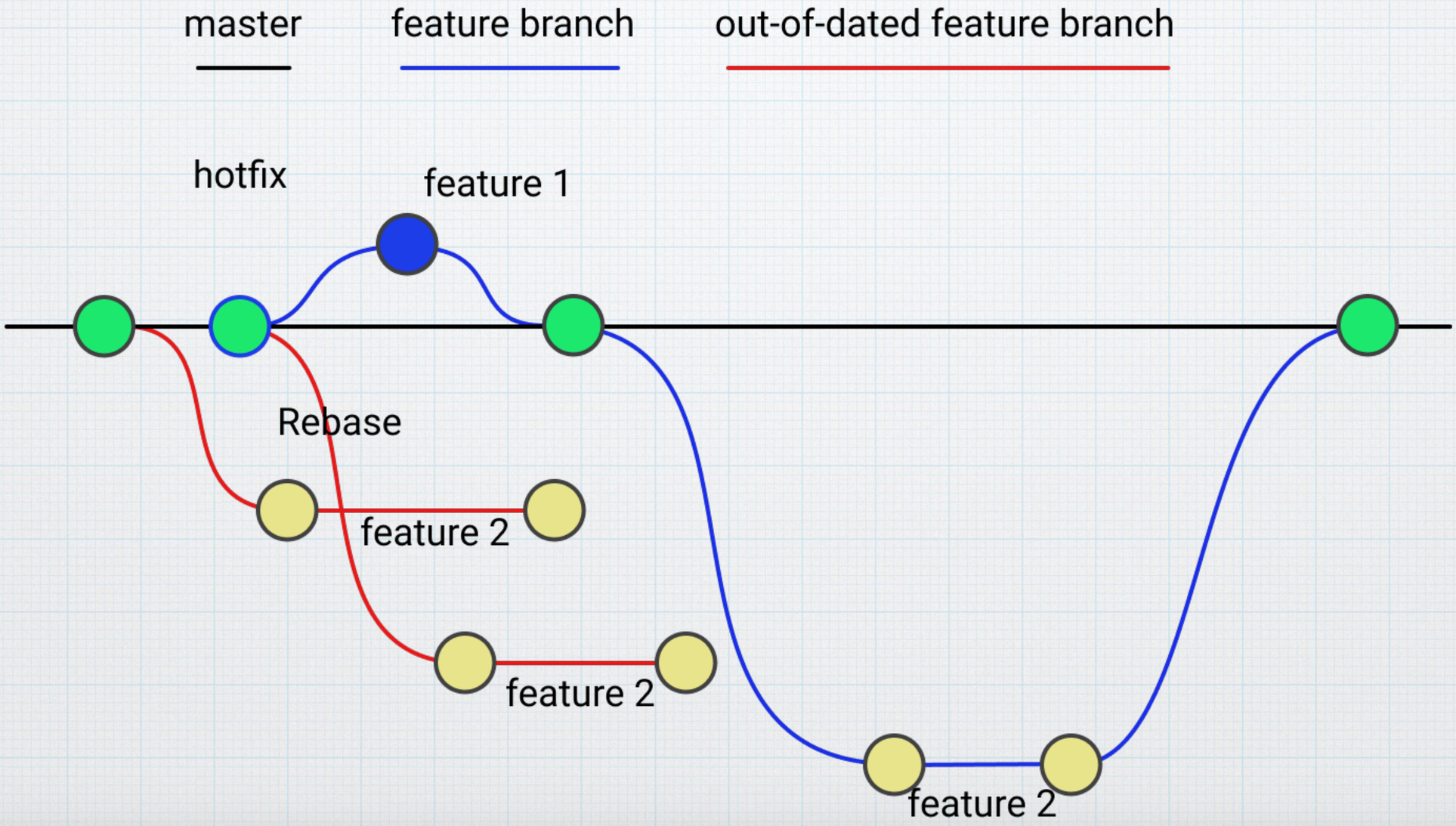
大型外包项目



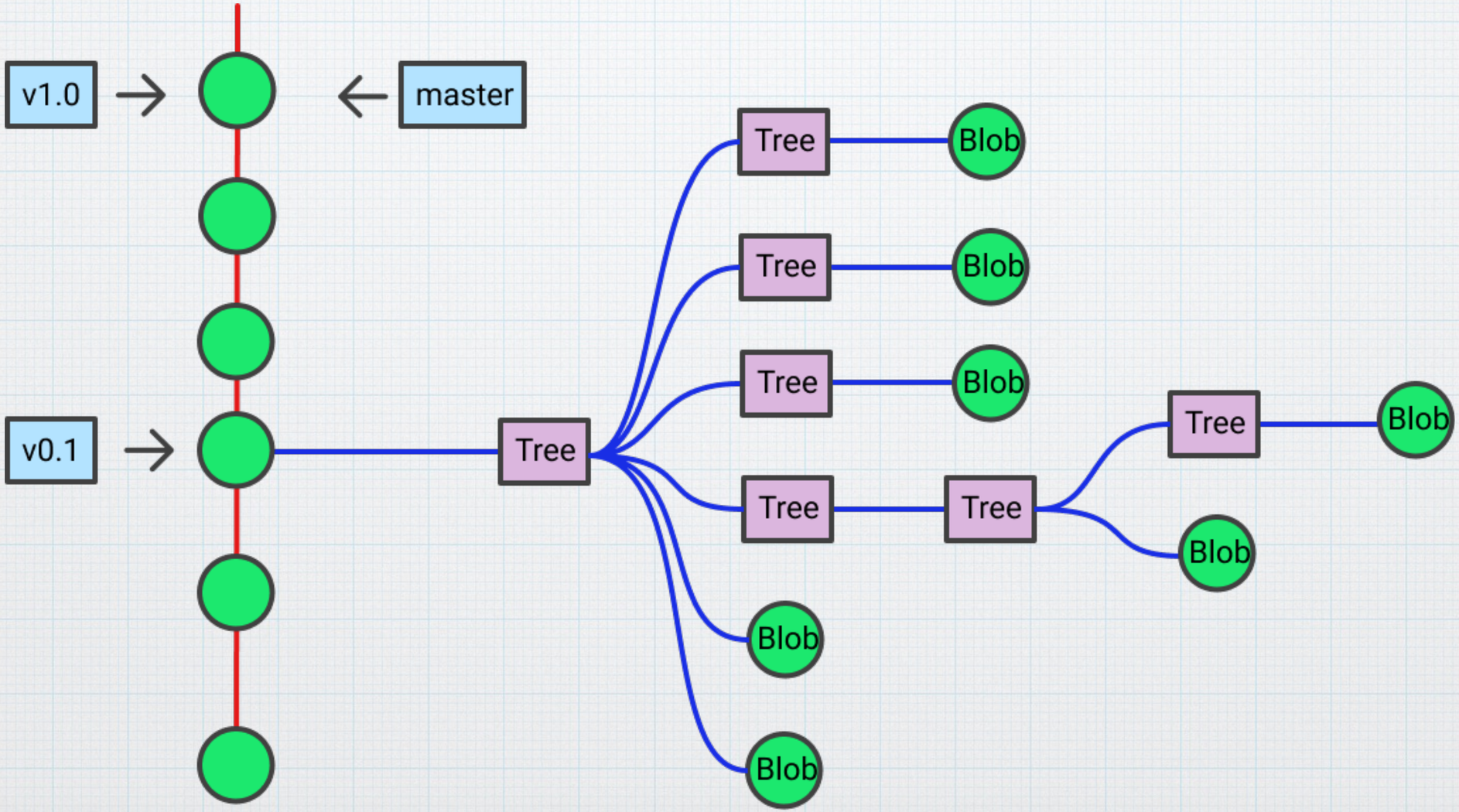
大型物联网项目

* 开发人员多
* 随时维护线上版本
* 分模块并行开发
* Code Review需求强烈
* 有紧急修复和紧急上线需求

大型互联网项目

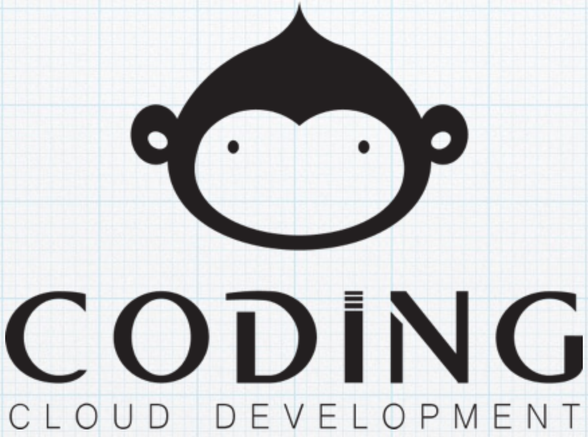


一张图透视 Git 原理



小贴士

* 冲突往往是因为代码结构划分不合理导致不同的人并行修改了同一段代码
* 大型文件会严重拖慢 Git 性能，推荐使用 Git LFS
* 可使用 Git Bisect 功能查找历史中的罪魁祸首
* 使用 Git Work-tree 可以为一个仓库检出多个版本



Q&A