01 版本控制系统介绍：为什么要使用git?

更新时间：2019-09-25 16:50:22

> 才能一旦让懒惰支配，它就一无可为。
>
> ——克雷洛夫

一、为什么要学习 Git?

在开发项目的时候，我们可能会不断地去修改代码，但是有时候会遇到，想查看某一时间的代码这种情况，如果没有版本控制器，你可能需要不断地定时备份代码，但这样显然是很麻烦的，而且备份也不一定好用，比如某个时间点并没有修改代码，那么备份就重复了；再比如虽然备份了代码，但你并不知道两个版本有什么区别。

1.1 版本控制系统

为了解决上面的一些问题，一些工程师便尝试开发代码版本控制器系统；每次当你修改完代码想进行备份时，只需要输入简单的命令，版本控制系统便会帮你完成备份操作；

在完成这个备份时候，大体会做这几件事情，首先把当前代码哈希一下，得到一个哈希 (hash) 值，同时把这个哈希值分配一个版本号，比如上一次的版本号是 `2`，那么这次的版本号便会是 `3`，用来保证它的顺序性；接着会比较当前的版本与上次版本的一些差异，包括文件差异，和文件里面的内容差异，并且会把这些差异单独存储起来，当你之后想看某一刻的修改时，可以非常方便地查看。

上面提到的是版本控制系统的最基本功能，应用因为实际需求不同，所以版本控制器也有不同的类型，最为常见的就是分布式版本控制系统和中央版本控制系统。

1.2 中央版本控制系统

中央版本控制系统必须存在两个端，服务端和客户端，当进行代码备份时，客户端会向服务端发出请求，并将此次修改的内容发送到服务器当中去；服务端收到请求后，会将代码存储在服务器当中；同样当客户端想查看某一个版本的修改内容或者想恢复到某一个版本之时，客户端也会发送请求到服务端，服务端再与之相应的响应。

从图中可以看到当提交代码时候，两个客户端都是把数据推送服务器当中去，拉取的时候代码也都是从服务器中获取，可以看出这是非常典型的 C/S 机制；张三和李四的交互都必须通过中央服务器，不能私下直接访问。

1.3 分布式版本控制系统

分布式版本控制器，主要是将备份的代码以及记录完全独立在本地存储，比如说上面提到，当你想将代码恢复到某一个版本的时候，本地版本控制器，不需要依赖网络便可以完成此操作，因为本地版本控制器拥有完整独立的控制系统。

从图中可以看出，张三和李四不仅仅可以向服务器推送代码，服务器也可以向客户端推送代码；并且张三和李四还可以互相推送代码，同样拉取代码也可以从任意一个节点中拉取，而不必须从服务器中拉取。

所以从分布式版本控制系统本身的功能来说，它们是完全平等的，每一个系统都拥有全部的功能；但在实际的工作中我们为了更好地管理代码版本，会人为设置一些规则来限制代码推送，所以在图中的红色线条通常是不会使用的，另外服务端通常也不会去主动向客户端推送和拉取代码。

1.4 GIT 和 SVN

前面提到版本控制系统的区别，我估计不少人会纠结该怎么去选择版本控制系统，这里我提一下我的一些看法：目前主流的版本控制系统主要有 Git 和 SVN，各自分别代表分布式版本控制系统和中央版本控制系统，两个工具各有优势。为了在后面的文章当中方便大家理解，后面将直接使用 Git 和 SVN 来表述分布式版本控制系统和中央版本控制系统，下面我主要从功能、热度、难度上来进行分析 Git 与 SVN 的优势与劣势。

1.4.1 功能丰富

Git 最大的优势在于功能更加丰富，虽然大部分功能在 SVN 上也都有，但是却极不方便，比如说分支功能。在 Git 中每个客户端都可以自由创建分支，而 SVN 中必须由服务端创建，客户端才能拉取，但实际工作中很少会在服务端执行 SVN 命令的；再比如说，在没有网络的情况下，Git 依然可以提交代码，而 SVN 则不行，因为必须连接服务器才可以提交。当然功能上还有很多区别，这里只是举了两个简单的例子，在实际操作中 Git 的功能远远多于 SVN。

1.4.2 流行热度

在选择一个软件的时候，我觉得很多人也会看流行度怎么样。在 5 年以前，可能 SVN 占了大部分市场。不过随着这些年 Git 的优势和越来越多的人对 Git 的熟悉，Git 已经占据了主流市场，并且 Git 的生态也越来越比 SVN 好，这个结论可以从 GitHub 和国内的代码托管平台码云默认只支持 Git 中看出。另外从身边的朋友中我得到一个观点，在一线城市 Git 使用得较多，在二三线城市使用 SVN 的公司较多，我猜测是这个原因主要是操作复杂度导致。

1.4.3 学习难度

可能看起来本地版本控制器比中央版本控制系统简单，潜意识会觉得操作也比较方便；但实际上恰恰相反，中央版本控制系统在操作上会相对简单，复杂的只是搭建过程；而本地版本控制系统则是安装简单，操作相对复杂一些。

因为开发者在使用中央版本控制系统时候，只需要提交代码和拉取代码即可，所能操作的项并不多，理解起来也相对简单；而本地版本控制系统则不同，除了提交代码到版本控制系统中，还需要将修改的记录同步给其他开发者，理解起来也相对复杂，所以结论是：**常规操作时候中央版本控制系统操作较为简单。**

1.5 本章小结

通过前面的介绍，我想大家对版本控制器，有了一定的认识，下面简单归纳概括一下：

- 版本控制系统分为`分布式版本控制系统`和`中央版本控制系统`。`分布式版本控制系统`每一个节点都是独立的系统，中央版本控制系统是典型的 `C/S` 架构，后面的内容分别以 `GIT` 和 `SVN` 来表述两个版本控制系统；
- 从存储空间来说 `Git` 比 `SVN` 更加占用空间，因为 `Git` 在每个端都保留了所有的版本历史，而 `SVN` 只在服务器中保存了历史版本记录，在客户端是没有保存本地版本历史的；
- `Git` 在流行程度、功能丰富方面比 SVN 有优势，在学习难度上， SVN 更占据优势。

}

02 环境准备：在各系统中快速安装 git 环境