# 版本控制器SVN

# 第1章：SVN概述

## 1. 为什么要使用版本控制器？

* 这个项目中需要一个很复杂的功能，老王摸索了一个星期终于有眉目了，可是这被改得面目全非的代码已经回不到从前了，什么地方能买到哆啦A梦的时光机啊？

——需求之一：备份还原！版本回退

* 小刚、小强同时修改了项目中的同一个index.js文件。小刚在index.js文件中的第30行声明了一个方法count()，随后进行保存，与此同时小强也在index.js文件中的第30也行声明了一个方法，叫sum()，也随后保存了，那么问题来了，当汇合文件时，在第30行以谁的为准呢？

——需求之二：协同修改！

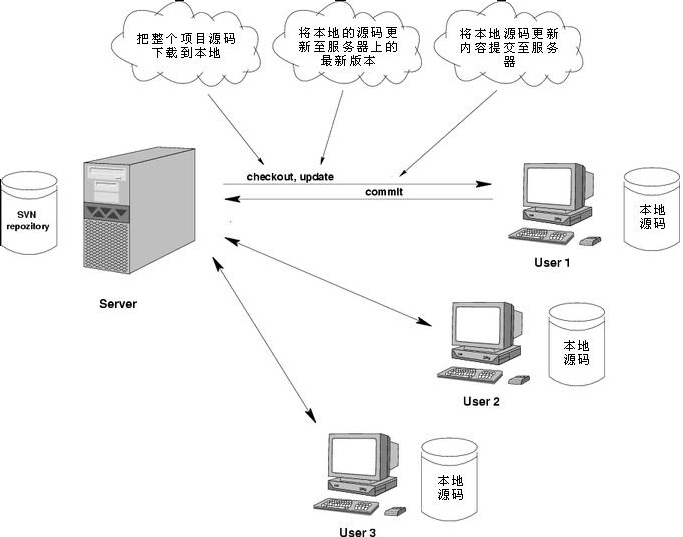
* 小张很郁闷，这几天总是在bug平台上收到测试小组的通知，要求他修正程序中存在的Bug，可他反复检查自己电脑上的代码，发现根本没有测试组说的bug，他心里默默的想：“难道我的代码被别人给改了？”是的，测试组的一个“实习生”私自动了小张负责的两个文件中的代码

——需求之三：权限控制！

## 版本控制是什么？

* 版本控制，最初来源于工程设计领域，是维护工程蓝图的标准做法，能追踪工程蓝图从诞生一直到定案的过程。是一种记录若干文件内容变化，以便将来查阅特定版本修订情况的系统。
* SVN [subversion]，就是一款实现集中式的版本控制工具软件，通常也称为版本控制器。

## 3. SVN的操作流程



## 4. SVN的安装

* 安装服务端：VisualSVN-Server
* 安装客户端：TortoiseSVN

备注：此处参考安装截图文件夹

## 5. SVN版本文件的常用状态(不一定显示)

|  |  |
| --- | --- |
| **项 目** | **图标** |
| 新建的文件 | 1 |
| 文件已经被标识，代表后期要交给svn管理 | 2 |
| 本地文件已经和服务器上的仓库一致(仅供参考) | 3 |
| 本地文件修改了, 但还没有提交 | 4 |
| 要提交的文件与服务器端的有冲突 | 5 |
| 当前文件被设置了锁定, 但当前用户没有获得锁, 不能操作 | 6 |
| 当前文件被设置了锁定, 但当前用户获得了锁, 可以操作 | 7 |

冲突

# 第2章：SVN的使用

## 版本控制

* 两个基本区域
  + 服务区
  + 用户区
* 三个基本操作
  + Checkout：检出代码(第一次检出)
  + Commit：提交代码(将自己写的代码推给svn服务器)
  + Update：更新代码(将服务端的代码同步到本地)
* 三个辅助操作
  + show log（查看文件历史记录）
  + revert（撤销）
  + update to version（更新为指定版本。版本回退）

## 2.冲突

* 冲突的原因？
  + 如果在同一个文件的同一个地方，都修改了或新增内容会引起版本冲突。
* 如何解决冲突？
  + 方法一: 手动解决
  + 方法二：借助你所使用的IDE解决
* 如何尽量减少冲突的产生？
  + 技巧一：每次修改文件之前，先更新，再去修改。
  + 技巧二：每次提交之前，先更新，再提交。
  + 技巧三：不要长时间“闭门造车”，多提交，勤提交。
  + 备注：无论是哪一种技巧，都不能完全的避免冲突。

## 3. 提交代码的规范

* 每次提交之前：先更新, 再提交。
* 敏感时间点，一定及时更新文件。
* 多提交，避免“只关注写代码，不关注提交”的现象。
* 每次提交必须书写清晰明了的提交说明。
* 不要提交不能通过编译的代码。
* 不要提交自己不明白的代码。
* 慎用锁定功能（尽量避免使用锁，不轻易解锁上锁的文件）。
* 不要提交本地自动生成的文件、文件夹。

## 6. 使用SVN开发的优势

* 可以掌握所有项目文件的历史纪录。
* 可以掌握团队的开发进度。
* 在任意修改后，可以回退之前的版本。