**软**

**件**

**项**

**目**

**管**

**理**

**上**

**机**

**报**

**告**

班级：16软工 1 班

学号：140、152、132、139

姓名：王润峰、周伟杰、莫家汉、唐伟武

周次：第 12 周 星期五

上机时间：2019-05-17 第1/2节

目录

[1. CVS版本管理工具 3](#_Toc9456500)

[2. SVN版本管理工具 3](#_Toc9456501)

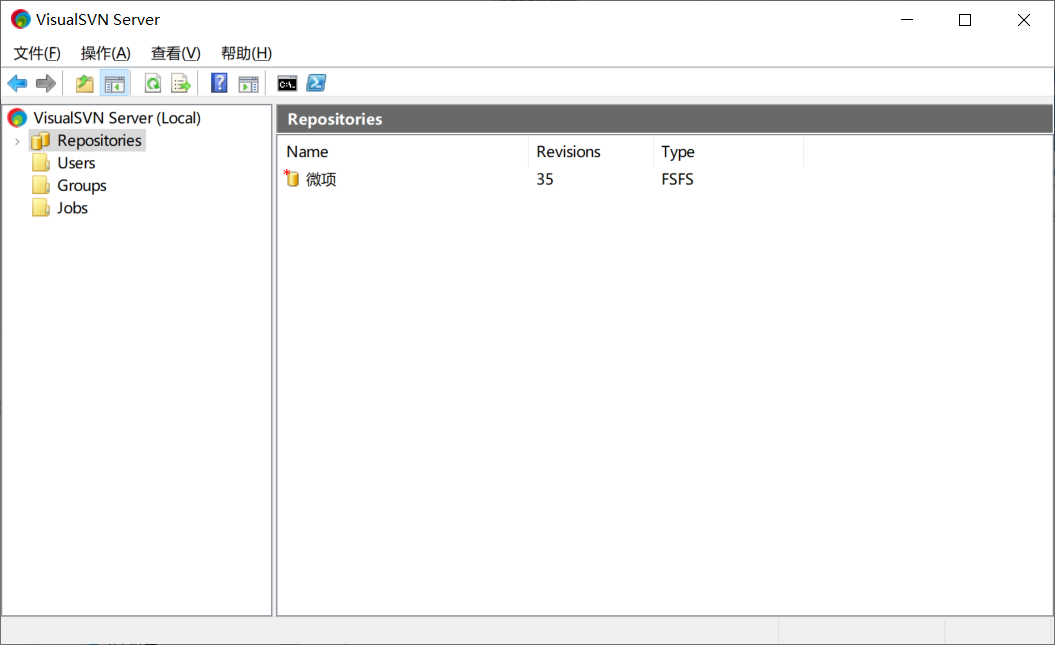
[3. SVN与CVS异同点 12](#_Toc9456502)

[4. 本次分工 14](#_Toc9456503)

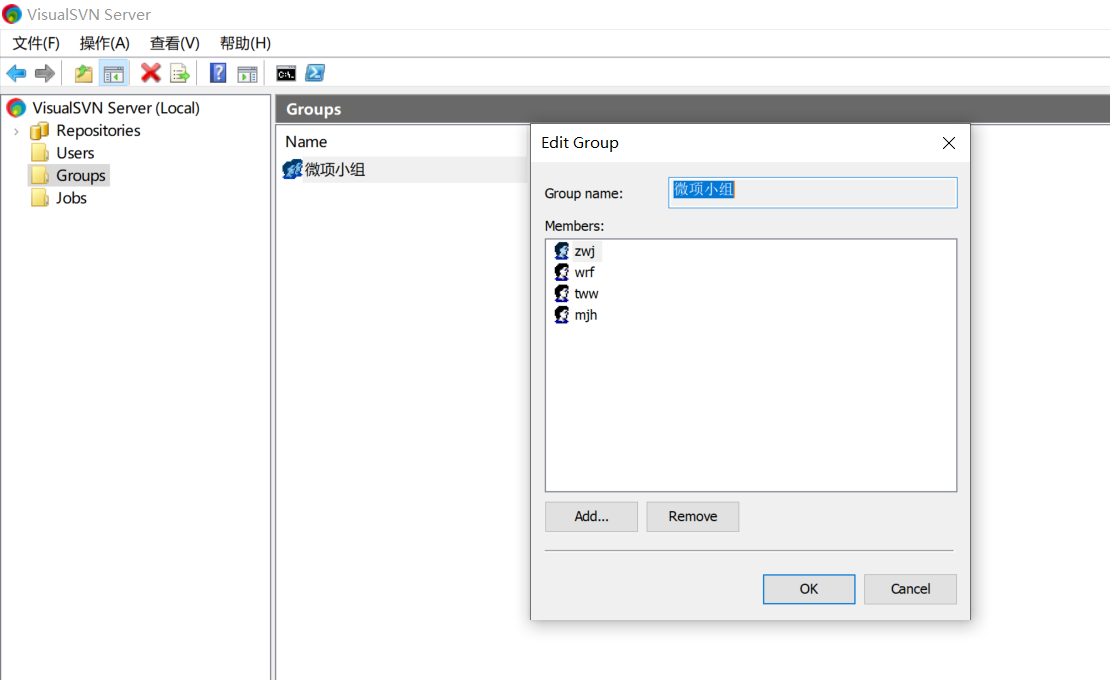
[5. 心得体会 14](#_Toc9456504)

1. CVS版本管理工具
2. SVN版本管理工具

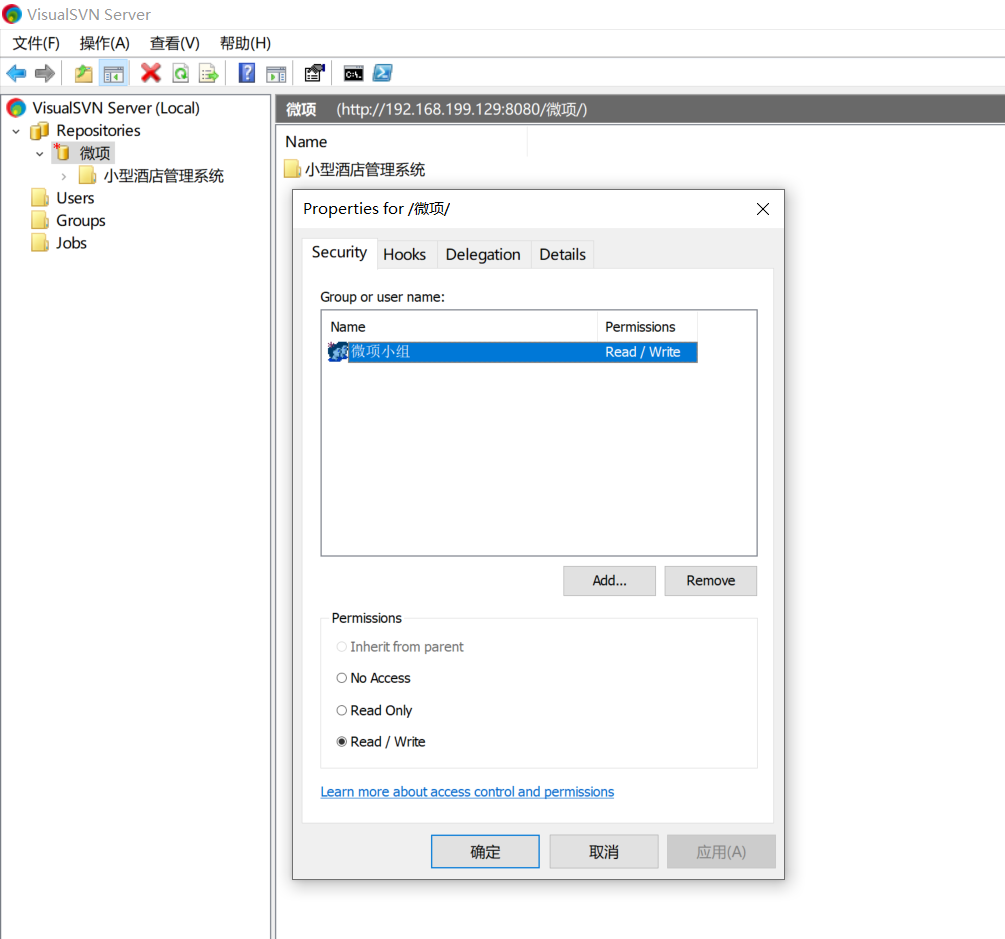
SVN建立小组项目



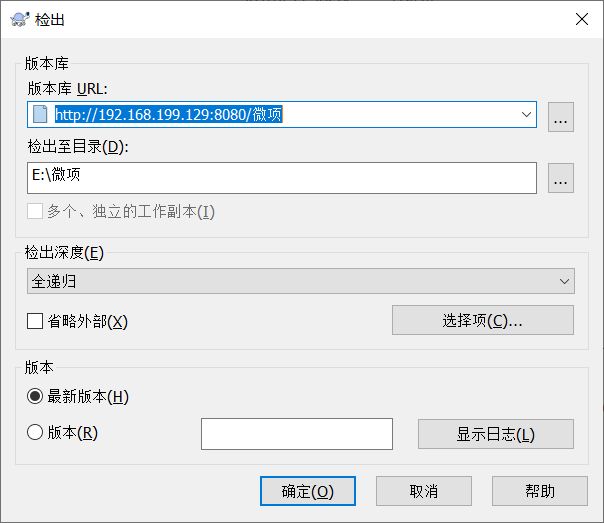
项目小组及成员



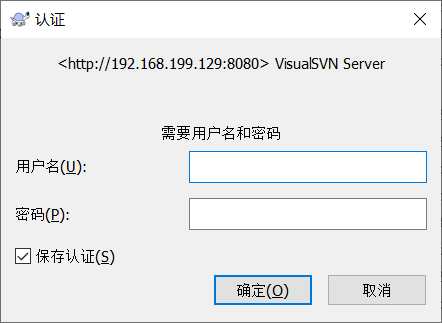
项目成员授权



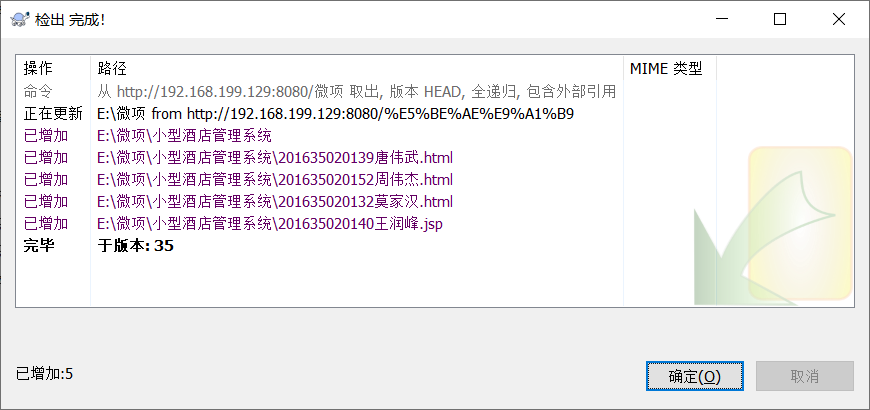
项目检出

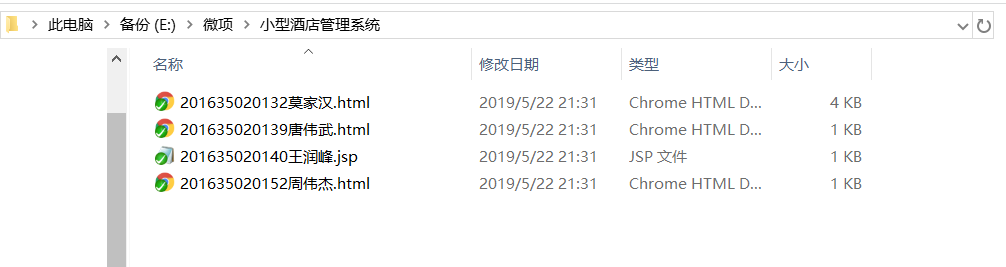


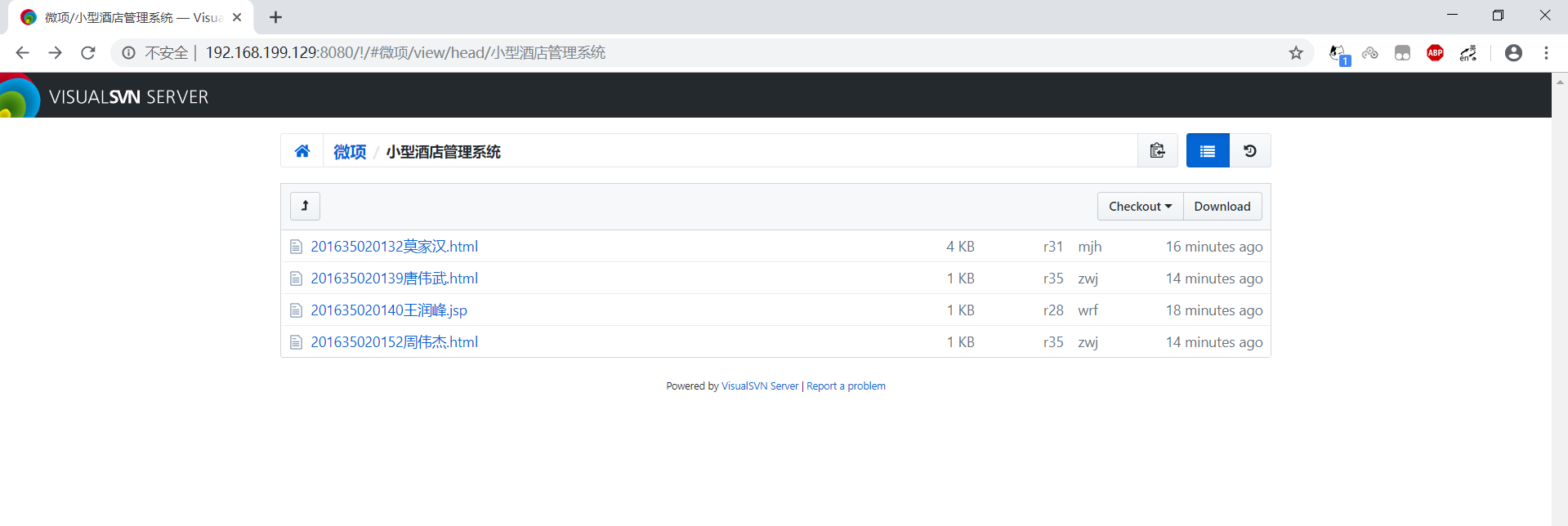
项目成员登录



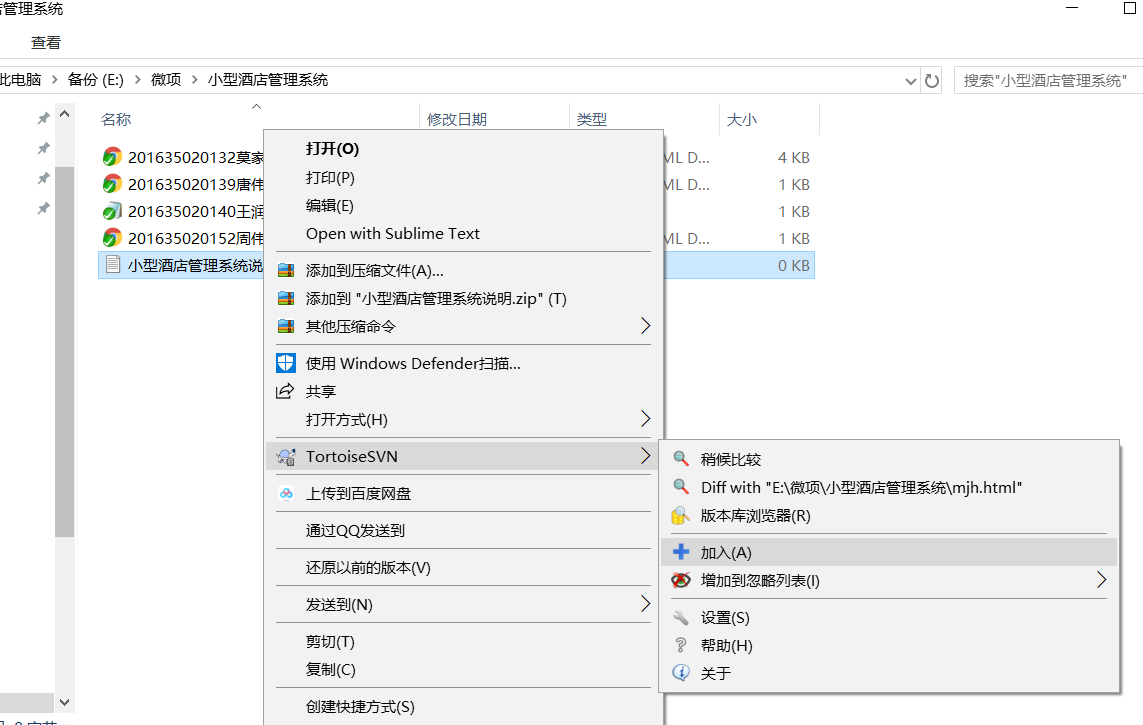
项目检出完成

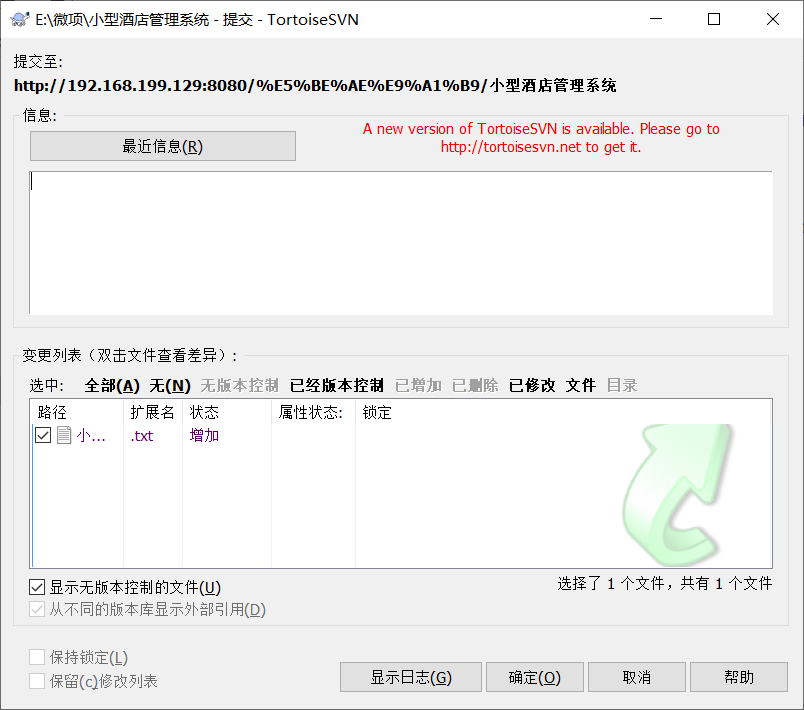


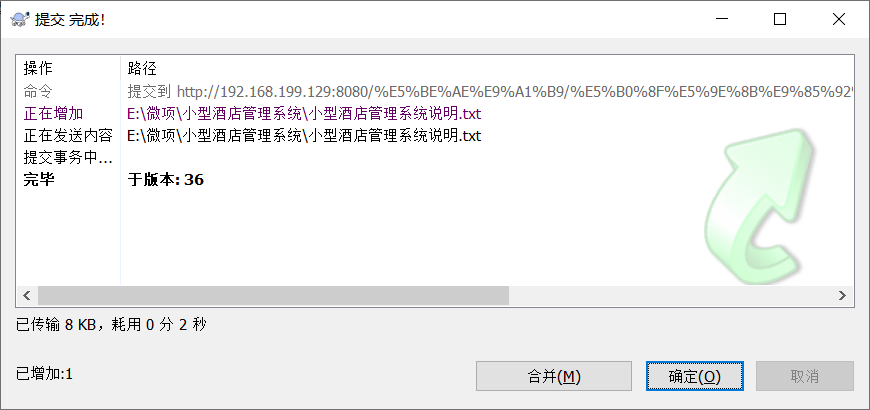


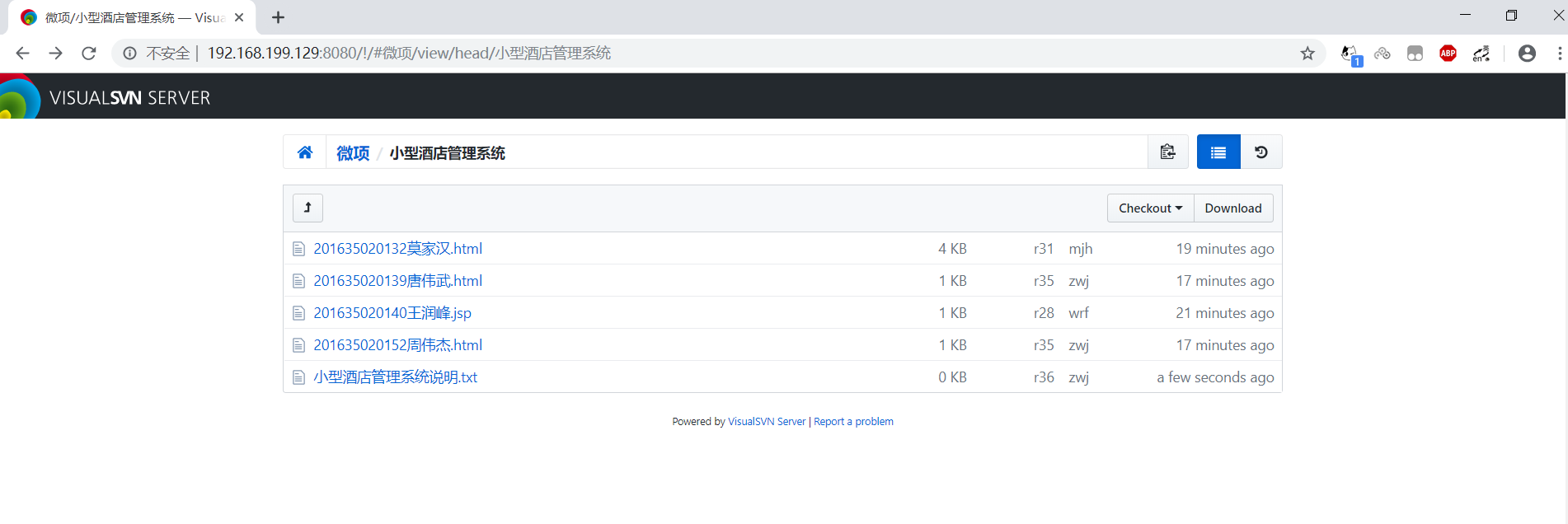


文件加入

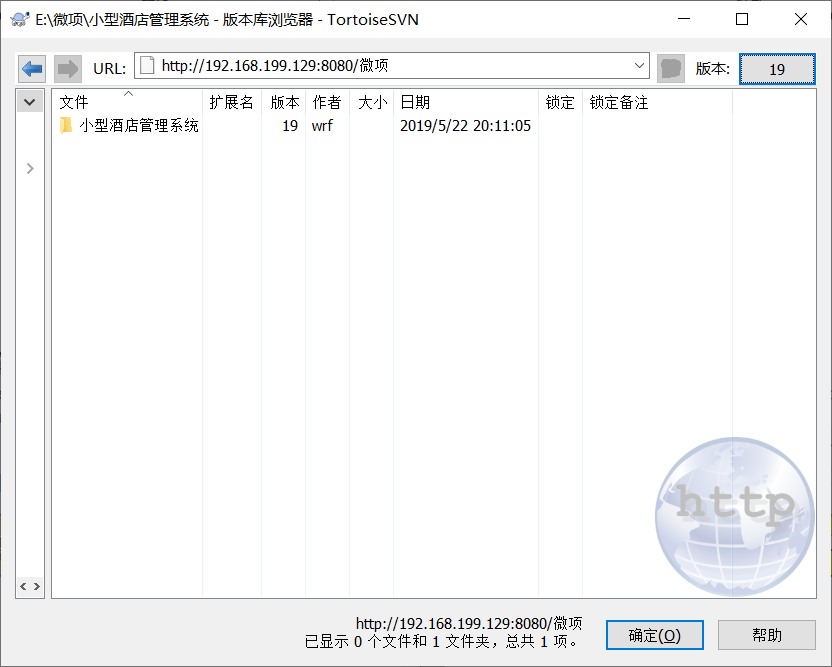


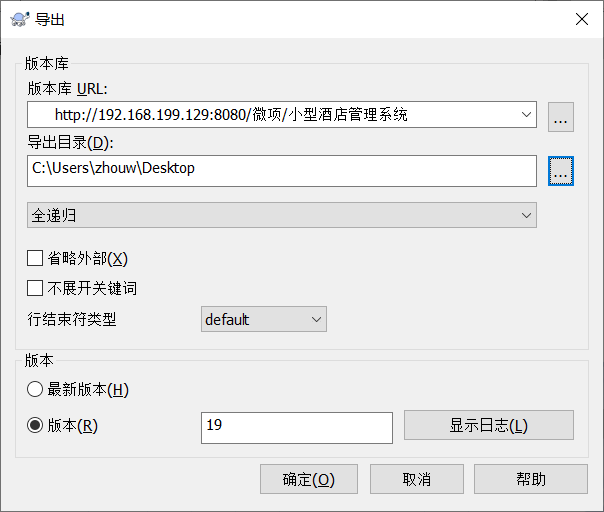


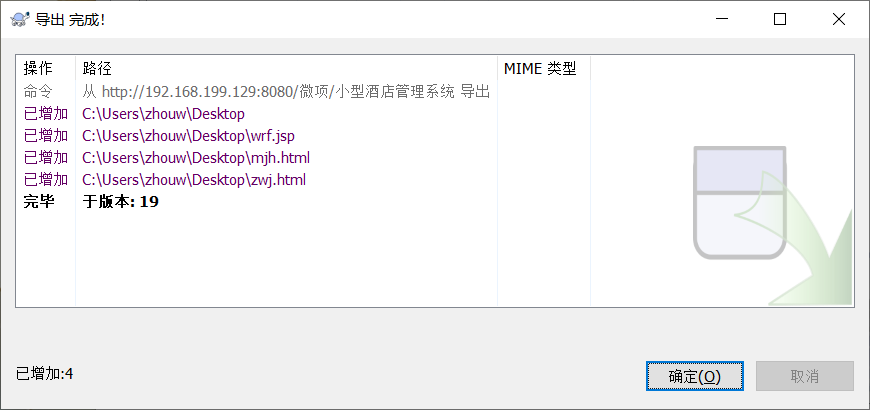




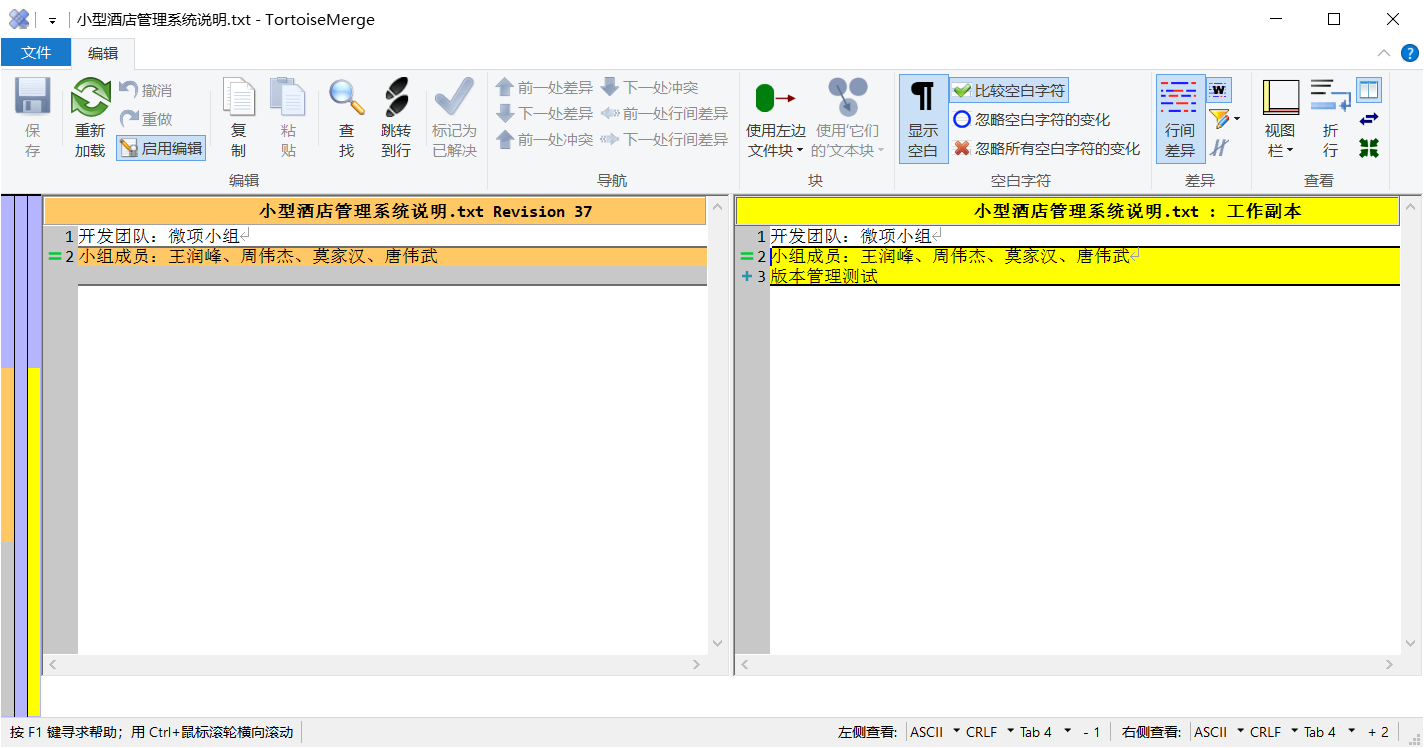
还原





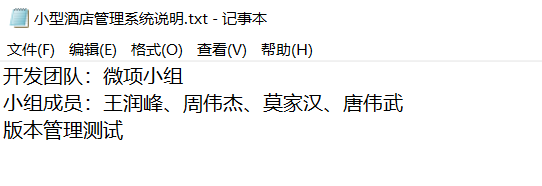


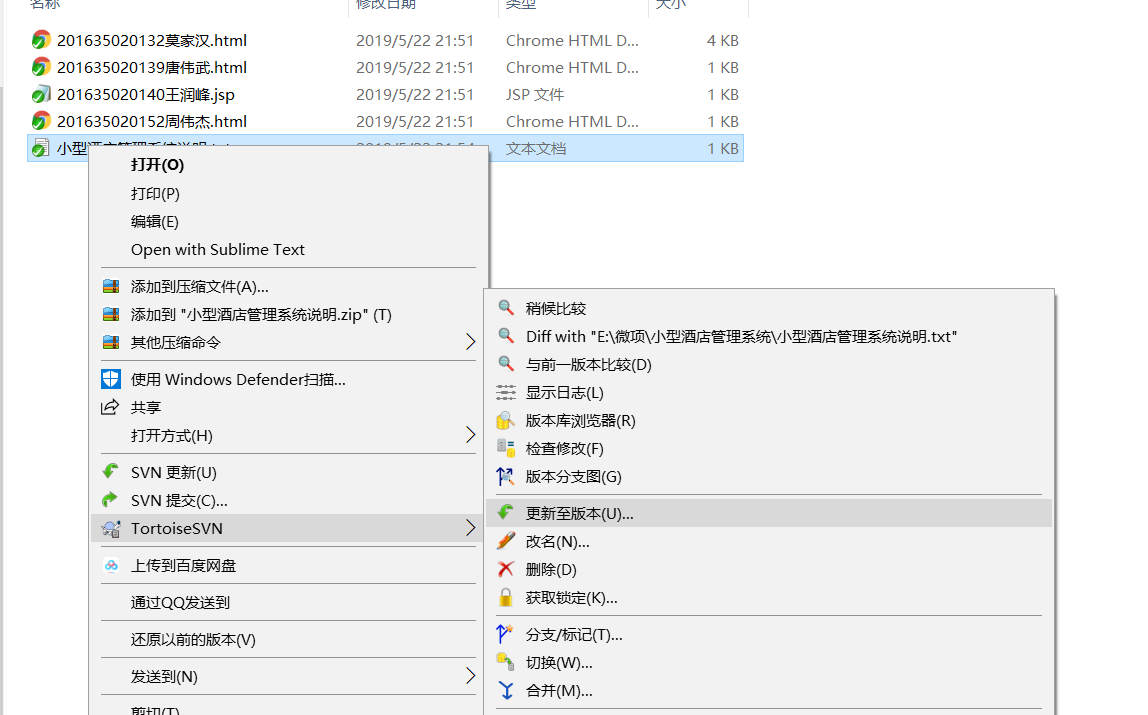
版本比较

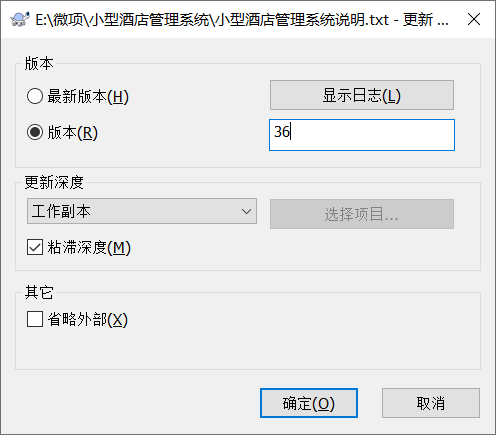


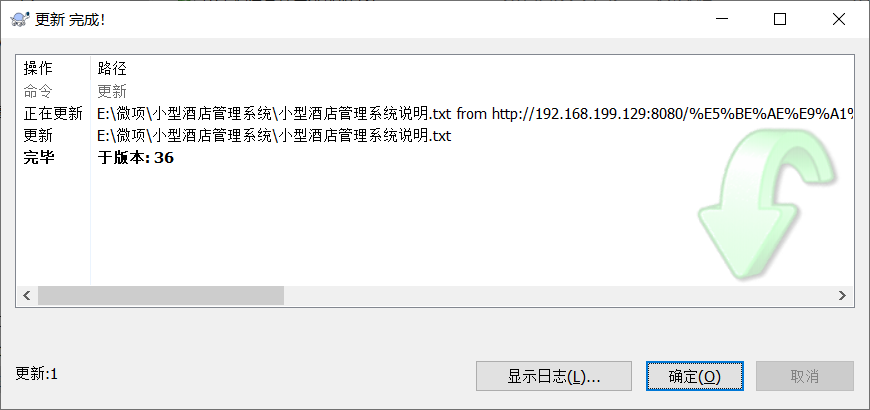
更新特定版本

版本38



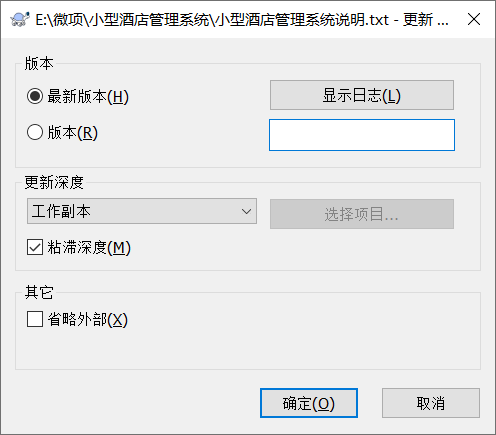


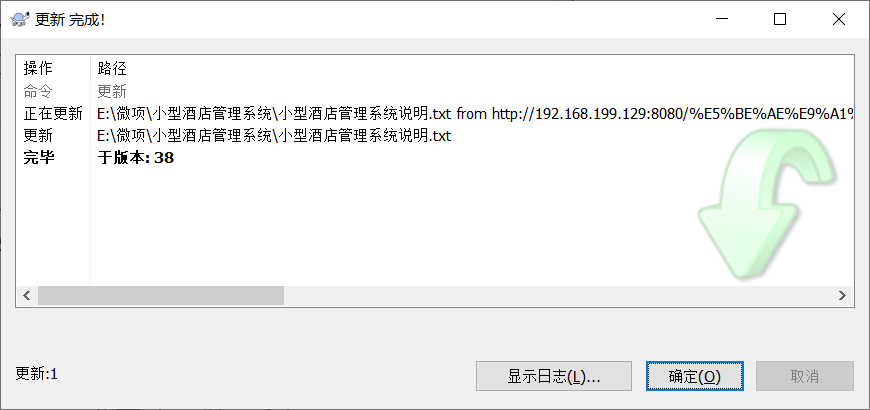




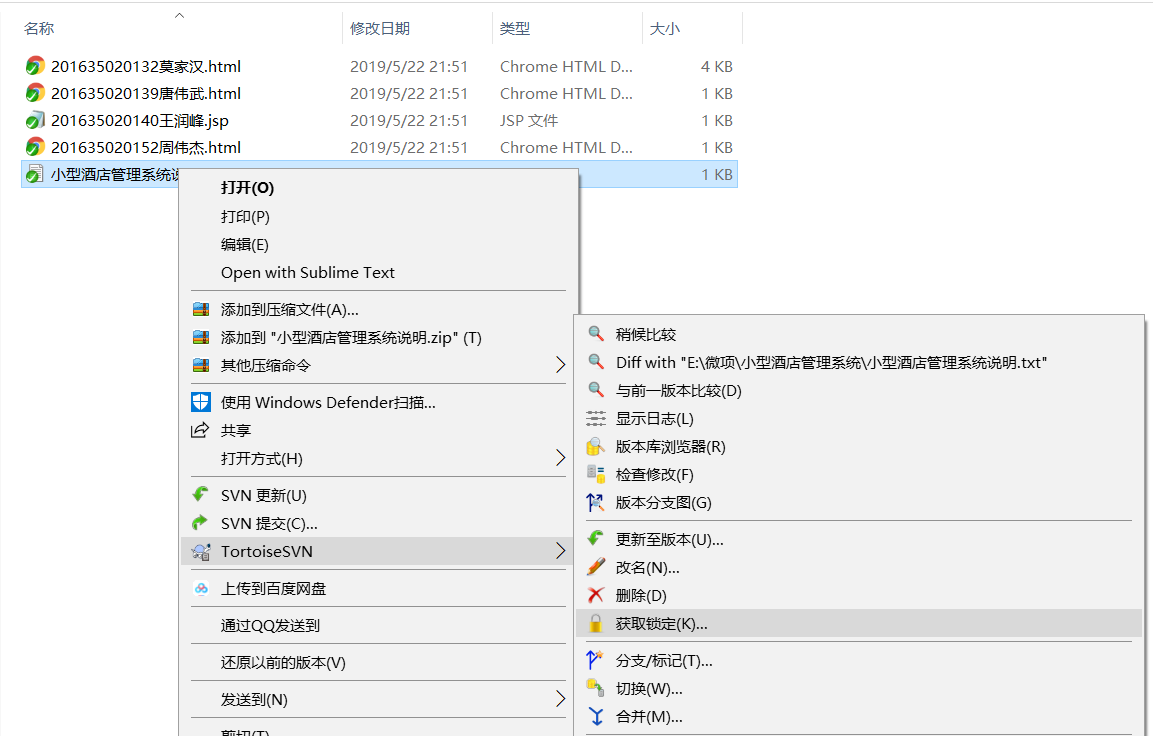


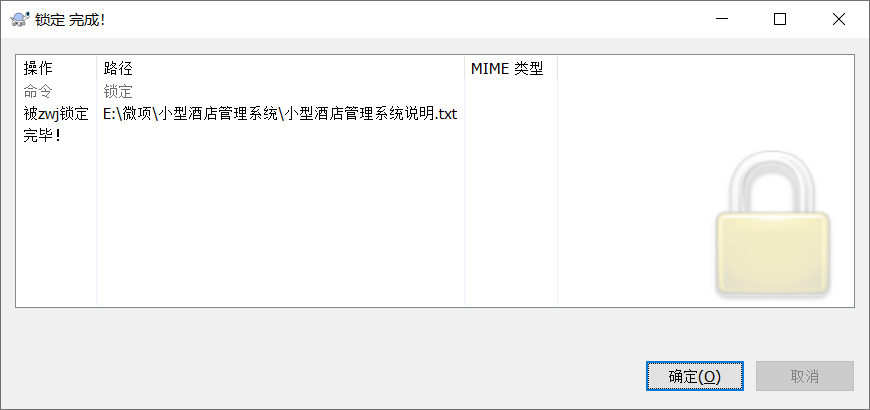
更新最新版本

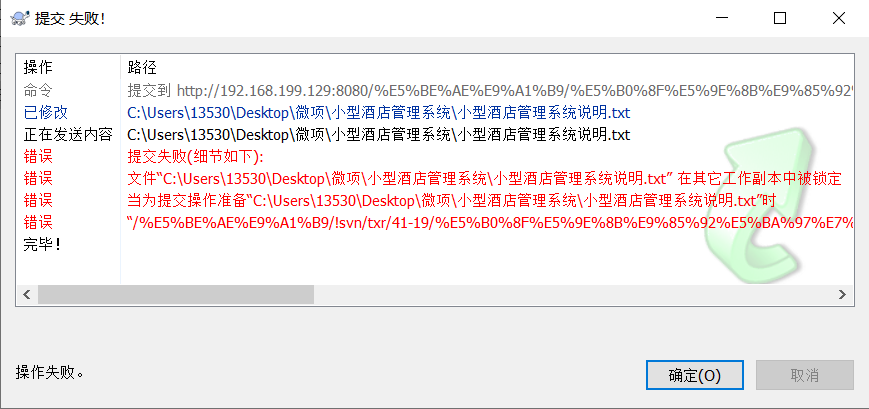




锁定







1. SVN与CVS异同点

|  |  |  |
| --- | --- | --- |
|  | CVS | SVN |
| 关于版本号 | 基于文件的自增序列号。 | 基于全局的自增序列号，而不仅仅针对文件，还包括目录等。 |
| 存储类型格式 | CVS是个基于RCS文件的版本控制系统。每个CVS文件都不过是普通的文件，加上一些额外信息。这些文件会简单的重复本地文件的树结构。因此，不必担心有什么数据损失，如果必要的话你可以手工修改RCS文件。 | SVN是基于关系数据库的(BerkleyDB)或一系列二进制文件的(FS\_FS)。一方面这解决了许多问题 (例如，并行读写共享文件)以及添加了许多新功能(例如运行时的事务特性。)。然而另一方面，数据存储由此变得不透明，或是说并不那么用户友好了。那就是为什么工具软件，对仓库 (数据库)变得那么重要了。 |
| 访问速度 | 比较慢；因为他基于单向（服务端-->客户端）差异化文件传输 | 整体而言,由于架构实现的不同, SVN的确比CVS快很多；因为采用双向差异化文件传输。 在网络上它只传输很少的信息并支持更多的离线模式的功能。但这也是有代价的。速度的代价就是巨大的存储（完全备份所有的工作文件）。 |
| 元数据 | 只允许存储文件 | 允许一个文件有任意的可命名属性。功能十分完全，但不知到底有什么用 |
| 文件类型 | 最初是为文本文件存储而设计的。因此其他文件类型（二进制，统一码）文件的支持几乎没有如需要的话则要有其他信息，并且客户端服务器端都要调整。 | SVN会关心所有的文件类型，不需要你来手工操作；因为它的存储是基于二进制的 |
| 滚回 | CVS允许任意的滚回，在任意一个已递交的版本上，尽管着要华些时间（所有的文件都要分别处理）。 | SVN不允许递交后滚回。我们建议把仓库里好的状态版本加到末尾，覆盖掉损坏的版本。而损坏的版本无论如何也是会存在数据库里的。 |
| 事务 | CVS中的 “零或一”事务原则根本没有实现。如果检入几个文件的话（加到服务器上），很有可能部分文件完成了，而另几个没有。最为一个潜规则，手工纠正这些并且对余下的文件 (而不是所有文件)一一重复检入。这样这些文件将在两阶段中被检入。但至今为止，因为这个功能缺少而导致的数据仓库损坏的案例还没有出现过。 | 支持“零或一”事务原则，这是SVN的一大优势 |
| 架构、代码、可扩展性 | CVS是个古老的系统。起初CVS只是一些运用RCS的脚本文件。后来这些脚本被组成一个单个应用程序，但内部结构仍然有待改进。直到今天，仍有人企图从头开始，重写一遍CVS,但都不成功。我们曾经尝试国重写客户端代码以期更好的集成效果，但是，不成功。现在我们都不认为CVS在功能上能走到多远了。 | Subversion的开发员的确花了许多时间在内部架构上。我们仍然不知道这些决策有多大的正确性等等。但有一点可以肯定，代码有良好的可扩展性，增强工作也在进行着。 |
| 网络层 | 不能与Apache Web Server集成 | 有抽象的档案库存取概念, 可以让人很容易地实作新的网络机制. Subversion “先进” 的网络服务器, 是 Apache 网页服务器的一个模块,它以称为 WebDAV/DeltaV 的 HTTP 变体协议与外界沟通. 这对Subversion 的稳定性与互通性有很大的帮助, 而且额外提供了许多重要功能: 举例来说, 有身份认证, 授权, 在线压缩, 以及档案库浏览. 另外也有小而独立的 Subversion 服务器程序, 使用的是自订的通讯协议, 可以很容易地透过 ssh 以 tunnel 方式使用 |
| 重命名、删除操作 | 不支持本地文件重命名提交； 删除分remove和erase两种前者把本地和库中文件都删除，后者只是删除本地文件； 不能删除文件夹 | 支持文件重命名提交系统会提示删除旧文件，创建新文件 删除本地文件提交 库中文件也被删除 |
| 用户访问权限 | 有read、write、creat、none这四种权限，任何人不能删除文件夹（admin也只能跑到服务器上把相应文件夹残忍删除？我暂时只知道这个方法……） | 只有read、write、none三种权限 creat和delete权限好象和write是捆绑在一起的 |
| 创建分支与基线 | CVS 在创建分支的时候，需要对所有进行分支的文件进行依次的操作，因此分支的建立成本（主要是建立分支所需的时间，或消耗的计算资源）与参与分支的文件数量成比例，项目越大，版本库越大，文件越多，分支的建立成本越高；基线（tag）的建立与此类似。 | SVN 的分支和基线是通过执行“拷贝”来建立的：回想一下在没有引入版本管理工具的时候我们是如何进行所谓的“分支”和“基线”管理的？答案显然是“拷贝” — 我们通过“拷贝”或“备份”来建立基线；同样，为支持多个开发人员可以同时进行开发，我们为每个开发人员创建一份“拷贝”。 |

1. 本次分工
2. 心得体会