Version Control

V. Paúl Pauca

Department of Computer Science
Wake Forest University

CSC 331-631
Fall, 2013

Version Control

Overview

- Tracks changes over source code
- Provides control over these changes
- Allows multiple versions to be deployed while developers work on updates
- Most systems allow multiple developers to edit the same file at the same time

Version Control

Overview

- Tracks changes over source code
- Provides control over these changes
- Allows multiple versions to be deployed while developers work on updates
- Most systems allow multiple developers to edit the same file at the same time

Some well-known systems (according to Wikipedia)

- **Bazaar**, used by Ubuntu and MySQL
- **Git**, used for the Linux Kernel and others
- **Mercurial**, used by Mozilla, NetBeans, GoogleCode, and others
- **Subversion**, used by Apache, FeeBSD, Ruby, SourceForge, and others

Subversion

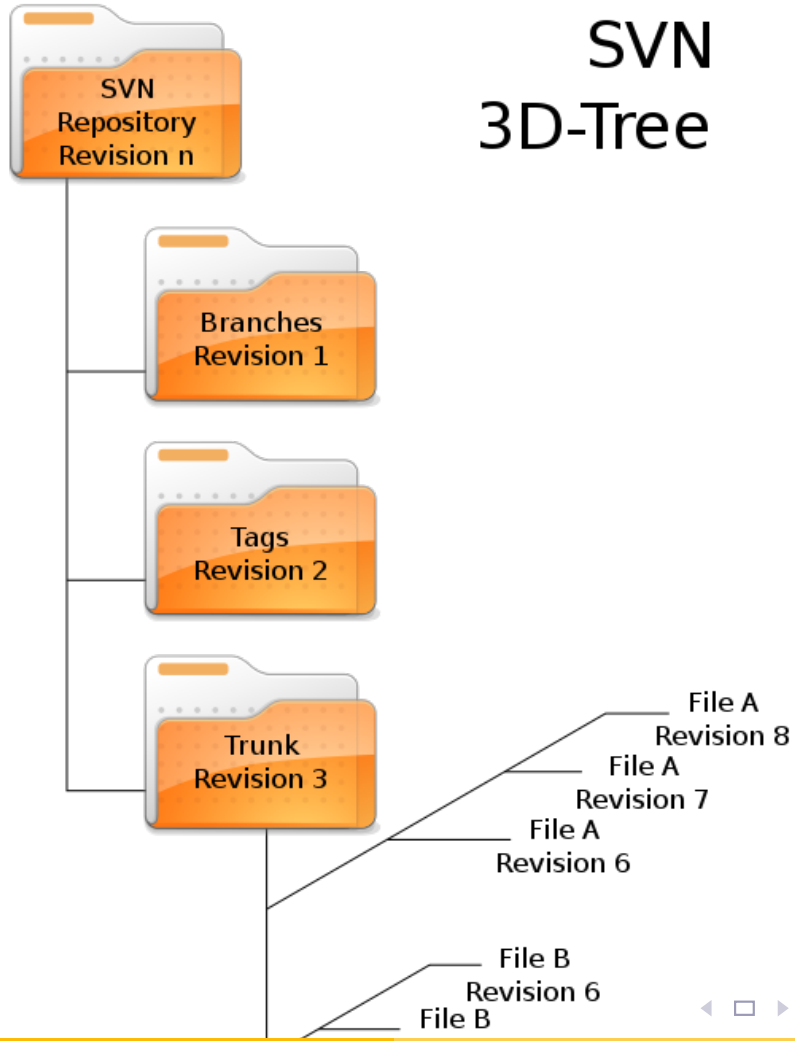
Developed in 2000 as an improvement to CVS

Key features

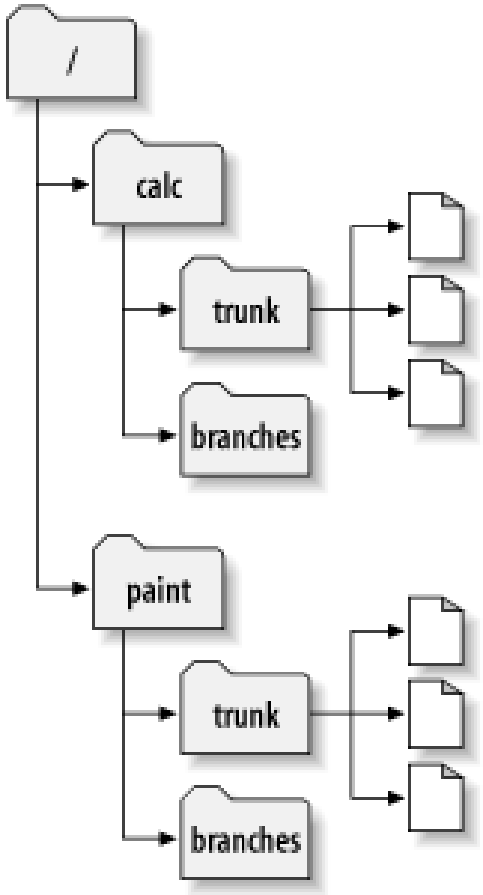
- Atomic commit (won't cause repository inconsistency if interrupted)
- Versioning of directories, renames, and removed files
- Native support for binary files
- HTTP server with WebDAV (allows https access), in addition to SSH
- Reserved checkouts (file locking)

Filesystem

3-D tree
(from Wikipedia)



Suggested Repository Layout



- Each project directory should contain subdirectories named `trunk` and `branches`

Creating the Repository Layout for Project `MyApp`

Will use `svn-se-group` as example repository

- Create the repository layout and move the contents of `MyApp` into `trunk`

```
$pwd
  /Users/username/MyProjects/MyApp
$mkdir ../trunk
$ mv * ../trunk
$ mv ../trunk .
$mkdir branches tags
```

- Your directory should look like this:

```
$pwd
  /Users/username/MyProjects/MyApp
$ls
  trunk
  branches
  tags
```

Importing an Existing Project

We now want to import project `MyApp` into the repository.

- First remove the `build` directory

```
$cd trunk
$ rm -rf build  # remove all object files and executables
```

- Now import your `MyApp` folder into the repository

```
$pwd   #make sure I am inside MyApp
   /Users/username/MyProjects/MyApp
$svn import https://theservername/svn-se-group/MyApp \
  -m "initial import of MyApp"
```

- Rename `MyApp` as `MyApp-old` or remove if you like.
  This copy is **not** under version control.

```
$cd ..
   /Users/username/Myprojects
$ mv MyApp MyApp-old
```

The Sample Repository

The repository now looks like this:

```
svn-se-group/

   MyApp/

      trunk/

         Classes/
         Images/
         Info.plist
         MainWindow.xib
         main.m

      branches/
```

Checking out `MyApp`

We now want to checkout a copy of `MyApp` from the repository for controlled development under Subversion.

- `cd` into the directory where you want to place `MyApp`

```
$pwd
   /Users/username/MyProjects/
$ svn checkout \
   https://theservername/svn-se-group/MyApp/trunk MyApp
```

- You know should have a copy of `trunk` saved in your local machine as directory `MyApp`

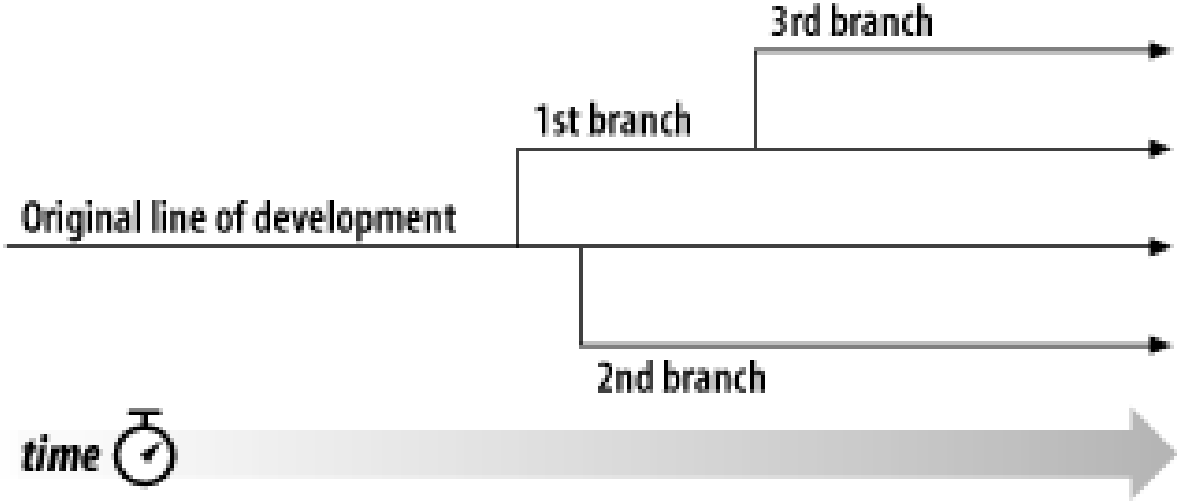
```
$cd MyApp
$ls
   Classes/ Images/ Info.plist MainWindow.xib main.m
```

- You can tell a directory is under version control if it has `.svn` in it

```
$ls -a
  ./ ../ .svn/ Classes/ Images/ Info.plist MainWindow.xib main.m
```

Branch

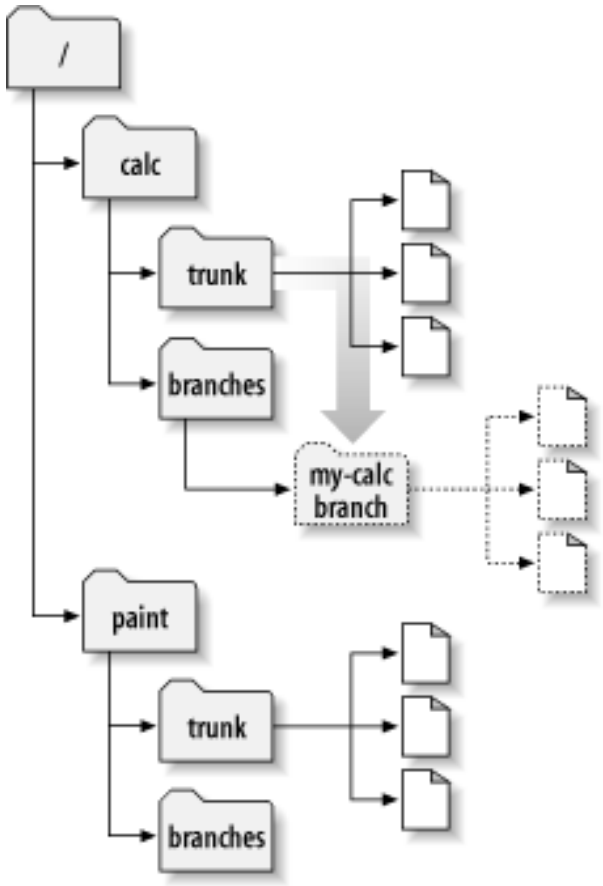
- An independent line of development



- Begins life as a copy

```
$svn copy https://theservername/svn-se-group/MyApp/trunk \
   https://theservername/svn-se-group/MyApp/branches/MyAppBranch1 \
   -m "Creating a private branch"
```

Branching



- New directory created in the next revision
- Branching happens in the server and not in the client
  Constant time operation in the server

Branching I

Checking out a branch

Must start in a directory **not** under version control

```
$pwd
   /Users/username/MyProjects/
   $ls -a  # should not have .svn/ directory
   ./ ../ MyApp/
```

Checkout the branch

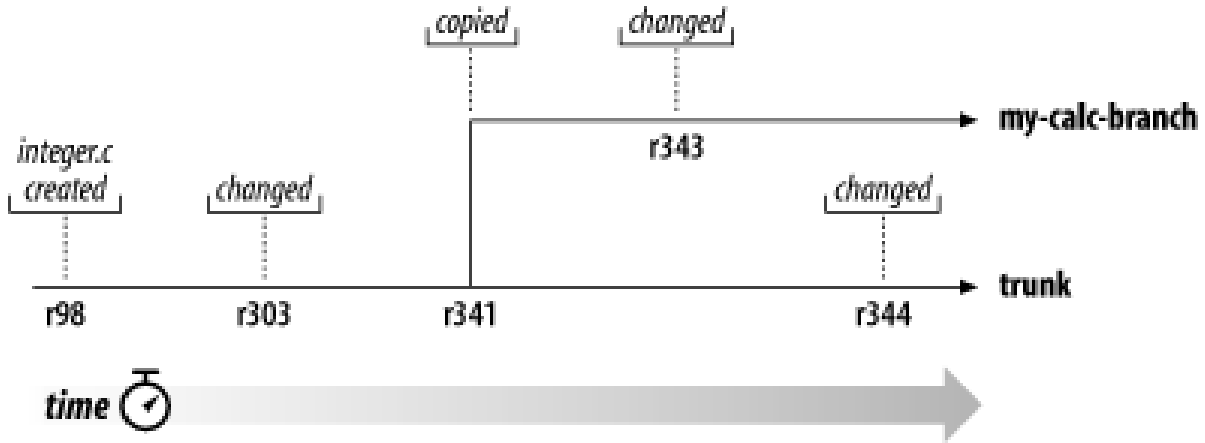
```
$svn checkout \
   https://theservername/svn-se-group/MyApp/branches/MyAppBranch1 \
      MyAppBranch1
$ls -a
   ./ ../ MyApp/  MyAppBranch1/
```

`cd` into `MyAppBranch1` to start developing this branch

Now commits to this branch don't affect the trunk

Branching II

View of two independent lines of development



Merging

Overview

- One can selectively copy changes between branches
- An entire branch can also be copied into the trunk (finished branch)

From trunk to branch

```
$cd MyAppBranch1
$svn merge https://theservername/svn-se-group/MyApp/trunk
```

- Copies all recent changes in `trunk` to branch `MyAppBranch1`
- Should be followed by resolution of conflicts and `commit`

Merging

From branch to trunk

```
$cd MyApp  # my local copy of the trunk
$svn update  # update to most recent copy
$svn merge --reintegrate \
   https://theservername/svn-se-group/MyApp/branches/MyAppBranch1
```

- Copies all recent changes in `MyAppBranch1` to my copy of the trunk
- Follow by conflict resolution and commit