CSCI 2132: Software Development

Subversion (and Git)

Norbert Zeh

Faculty of Computer Science
Dalhousie University

Winter 2019

Version Control Systems

A version control system allows us to

- Record the history of changes to the source code of some software we are writing (and many more)
- Maintain multiple versions of the (software) product
- Coordinate the work by multiple team members via
  - Multiple branches
  - Support for merging from different branches

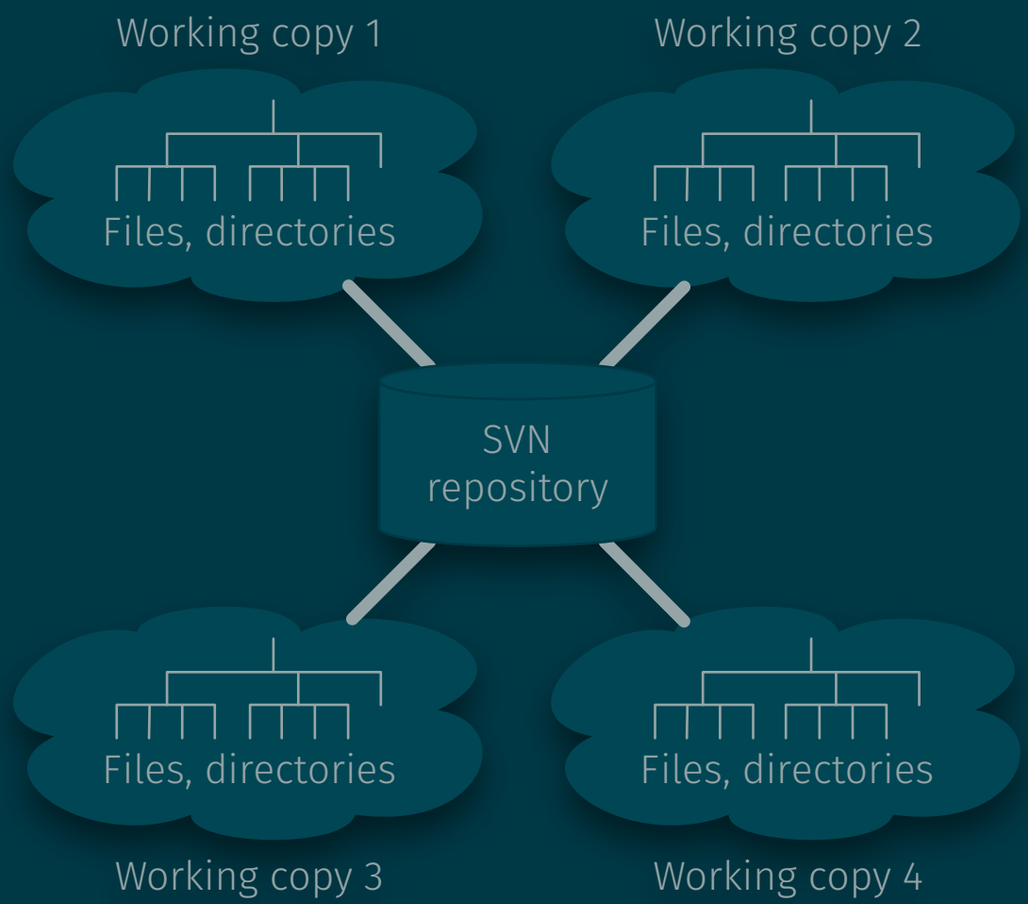
Two main types of VCS:

- **Centralized:** one central repository (RCS, SCCS, VCS, Subversion, ...)
- **Distributed:** multiple distributed repositories (Git, Darcs, Mercurial, ...)

Version Control Using Subversion (SVN)

A simplified view:

- Backups: Create backups in a repository
- History: “Time machine”, labelled versions
- Collaborative, central repository: Different users can contribute and merge changes



SVN Checkout

- Create an initial working copy:

```
$ svn checkout
```

or

```
$ svn co
```

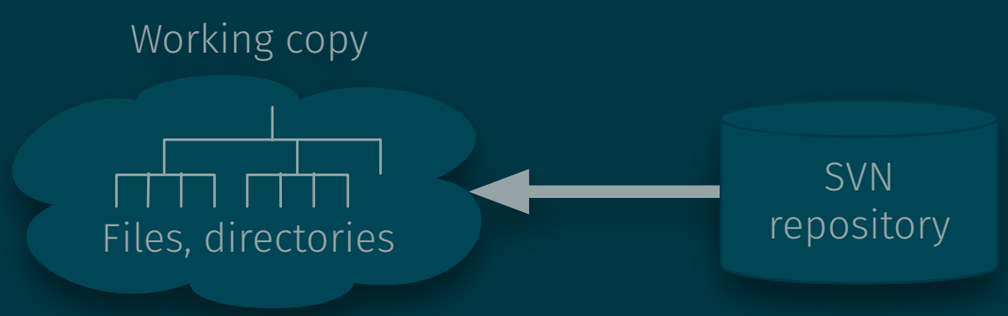
SVN Checkout

- Create an initial working copy:

```
$ svn checkout
```

or

```
$ svn co
```



SVN Add

- Creating files in the working copy does not add them to subversion!
- `svn add` adds them to SVN's list of files to manage
- The SVN repository does not know about the added file yet!

```
$ svn add newfile.txt
```

SVN Commit

- `svn commit` saves local changes to the repository.
- Local working copy is kept.
- Local working copy can then be deleted without affecting the repository.
- `svn commit` requires a log message that documents the changes that were made in this commit.

```
$ svn commit -m”Added newfile.txt”
```

SVN Commit

- `svn commit` saves local changes to the repository.
- Local working copy is kept.
- Local working copy can then be deleted without affecting the repository.
- `svn commit` requires a log message that documents the changes that were made in this commit.

```
$ svn commit -m"Added newfile.txt"
```

```
$ cd .. ; rm -r WorkingCopy
```

SVN Update

- SVN does not allow you to commit changes unless your local copy has an up-to-date view of the repository, including changes others may have committed from their working copies.
- `svn update` updates your local copy according to the current state of the repository.
- This may create conflicts that you may have to resolve before committing your changes.
- The chance for conflicts increases the longer you work without running `svn update`. $\Longrightarrow$ Run `svn update` periodically.

SVN Update

- SVN does not allow you to commit changes unless your local copy has an up-to-date view of the repository, including changes others may have committed from their working copies.
- `svn update` updates your local copy according to the current state of the repository.
- This may create conflicts that you may have to resolve before committing your changes.
- The chance for conflicts increases the longer you work without running `svn update`. $\Longrightarrow$ Run `svn update` periodically.

(Re)moving files

- Moving a file will not move it in the repository.
- Removing a file will not remove it from the repository.
- SVN complains about the missing file the next time we try to run `svn commit`.
- `svn rm` removes the file from the repository.
- `svn mv` renames or moves the file within SVN's file tree.
- Changes will take effect when running `svn commit` next.

SVN Troubleshooting

- Do not interrupt an SVN operation (unless it's hung, takes very long). (This may leave SVN in a corrupt state.)
- Helpful commands: `svn info`, `svn status -v`, `svn log -v`
- A working copy contains a hidden `.svn` directory, which stores administrative information about the working copy.
- Resolve problems by moving or removing a working copy and checking out a new copy.
- If you allow SVN to save your password, you can remove it with `rm ~/.subversion/auth/svn.simple/*`.

SVN and Git

- Many version control systems
  (SCCS → RCS → CVS → Subversion, Git, Mercurial, Darcs, ...).

- SVN and Git are the most popular representatives of two competing philosophies:
  - SVN (RCS, CVS, ...): One centralized repository
  - Git (Mercurial, Darcs, ...): Fully distributed, no centralized repository
  - More on Git later.

- Git can be used in a subversion-like manner:
  - `svn co` ≈ `git clone`
  - `svn add` ≈ `git add`
  - `svn commit` ≈ `git commit` + `git push`
  - `svn update` ≈ `git pull`