An Introduction to Subversion

Flavio Stanchi

August 16, 2014

Features of Subversion

- It's a version control system

Features of Subversion

- It's a version control system
- Uses a centralized model:
  - Server-client approach

Features of Subversion

- It's a version control system
- Uses a centralized model:
  - Server-client approach
  - Version merging

Features of Subversion

- It's a version control system
- Uses a centralized model:
  - Server-client approach
  - Version merging
  - With wireless connections everywhere, it's rarely a limitation

Features of Subversion

- It's a version control system
- Uses a centralized model:
  - Server-client approach
  - Version merging
  - With wireless connections everywhere, it's rarely a limitation
- Easy to learn (but slower than Git)

Features of Subversion

- It's a version control system
- Uses a centralized model:
  - Server-client approach
  - Version merging
  - With wireless connections everywhere, it's rarely a limitation
- Easy to learn (but slower than Git)
- It's free

A bit of history

- The SVN open-source project was founded in 2000 by CollabNet

A bit of history

- The SVN open-source project was founded in 2000 by CollabNet
- Version 1.0 was released in February 2004

A bit of history

- The SVN open-source project was founded in 2000 by CollabNet
- Version 1.0 was released in February 2004
- In 2009 it was accepted into Apache Incubator

A bit of history

- The SVN open-source project was founded in 2000 by CollabNet
- Version 1.0 was released in February 2004
- In 2009 it was accepted into Apache Incubator
- Git development started in 2005 to substitute BitKeeper (Linus Torvalds is one of the founders)

Getting Subversion

- Subversion can be found at `https://subversion.apache.org`

Getting Subversion

- Subversion can be found at `https://subversion.apache.org`
- Version 1.8 is the last release at this time

Getting Subversion

- Subversion can be found at `https://subversion.apache.org`
- Version 1.8 is the last release at this time
- Client for Windows:
  - TortoiseSVN (free): `http://tortoisesvn.net/`

Getting Subversion

- Subversion can be found at `https://subversion.apache.org`
- Version 1.8 is the last release at this time
- Client for Windows:
  - TortoiseSVN (free): `http://tortoisesvn.net/`
- Clients for Mac:
  - Xcode (free): `https://developer.apple.com/xcode/downloads/`
  - Versions ($): `http://versionsapp.com/`

Getting Subversion

- Subversion can be found at `https://subversion.apache.org`
- Version 1.8 is the last release at this time
- Client for Windows:
  - TortoiseSVN (free): `http://tortoisesvn.net/`
- Clients for Mac:
  - Xcode (free): `https://developer.apple.com/xcode/downloads/`
  - Versions ($): `http://versionsapp.com/`
- If you are using Linux ...

Getting Subversion

- Subversion can be found at `https://subversion.apache.org`
- Version 1.8 is the last release at this time
- Client for Windows:
  - TortoiseSVN (free): `http://tortoisesvn.net/`
- Clients for Mac:
  - Xcode (free): `https://developer.apple.com/xcode/downloads/`
  - Versions ($): `http://versionsapp.com/`
- If you are using Linux . . . use the terminal!

Subversion on Cornell servers

- CISER on RSCH106

Subversion on Cornell servers

- CISER on RSCH106
- ECCO

Subversion on Cornell servers

- CISER on RSCH106
- ECCO
- Quick reference guide at `http://www2.vrdc.cornell.edu/news/documentation/subversion/`

Where to create a repository?

- TeamForge:
  - Create an account at `https://forge.cornell.edu`
  - Follow the instructions at `http://www.it.cornell.edu/services/subversion/howto/index.cfm`

Where to create a repository?

- TeamForge:
  - Create an account at `https://forge.cornell.edu`
  - Follow the instructions at `http://www.it.cornell.edu/services/subversion/howto/index.cfm`
- GitHub at `https://github.com/`

Where to create a repository?

- TeamForge:
  - Create an account at `https://forge.cornell.edu`
  - Follow the instructions at `http://www.it.cornell.edu/services/subversion/howto/index.cfm`
- GitHub at `https://github.com/`
- Use *svnserve* as a lightweight custom server

Where to create a repository?

- TeamForge:
  - Create an account at `https://forge.cornell.edu`
  - Follow the instructions at `http://www.it.cornell.edu/services/subversion/howto/index.cfm`
- GitHub at `https://github.com/`
- Use *svnserve* as a lightweight custom server
- Test repository at `http://repository.vrdc.cornell.edu/public/test` (When prompted for a login, use 'testuser'/'testuser')

Centralized version control

- Server-client approach
  - The repository is located in the server

Centralized version control

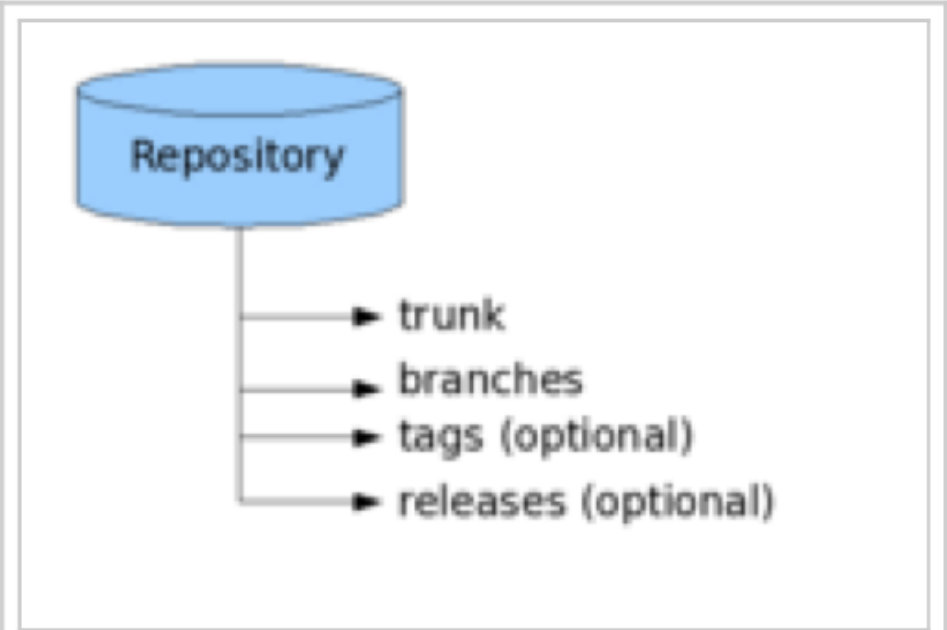
- Server-client approach
  - The repository is located in the server
  - No version control over local copies: *commit* is also used for sharing (no need to *push*)

Centralized version control

- Server-client approach
  - The repository is located in the server
  - No version control over local copies: *commit* is also used for sharing (no need to *push*)
- Version merging:
  - Multiple editors can check out any given file
  - Discrepancies are handled upon checkin

Generic setup

- Trunk: contains all the clean code
- Branches: where all initial work occurs
- Tags and releases (optional)



Generic repository setup

Local copy

- The repository may be remote or local . . .

Local copy

- The repository may be remote or local . . . but you don't usually work directly with it

Local copy

- The repository may be remote or local . . . but you don't usually work directly with it
- Instead, check out a local copy of the repository (or of its subelements)

Local copy

- The repository may be remote or local . . . but you don't usually work directly with it
- Instead, check out a local copy of the repository (or of its subelements)
- Make changes to the local copy
  - Important: use Subversion commands to do this, so that every change is registered

Local copy

- The repository may be remote or local . . . but you don't usually work directly with it
- Instead, check out a local copy of the repository (or of its subelements)
- Make changes to the local copy
  - Important: use Subversion commands to do this, so that every change is registered
- Commit the changes back into the repository
  - Add a commit (log) message
  - Every commit is registered with a revision number

Local copy

- Note: direct changes to the repository are immediately applied ...

Local copy

- Note: direct changes to the repository are immediately applied . . . while changes to the local copy are applied to the repository upon commit

Local copy

- Note: direct changes to the repository are immediately applied . . . while changes to the local copy are applied to the repository upon commit
- Hence, commit frequently!

The terminal

- On Linux and on OSX, use the terminal

The terminal

- On Linux and on OSX, use the terminal
- Advantages:
  - Flexibility
  - It's not so complicated

The terminal

- On Linux and on OSX, use the terminal
- Advantages:
  - Flexibility
  - It's not so complicated
- Every command must be preceded by *svn*

```
server> svn co repository:trunk /programs/production/prod/current
```

The terminal

- On Linux and on OSX, use the terminal
- Advantages:
  - Flexibility
  - It's not so complicated
- Every command must be preceded by *svn*

```
server> svn co repository:trunk /programs/production/prod/current
```

- svn <subcommand> [options] [args]

Basic workflow

1. Update your working copy or check out a new one
   - Commands: *co* (check out), *update*

Basic workflow

1. Update your working copy or check out a new one
   - Commands: *co* (check out), *update*
2. Make changes
   - Use your favorite editors
   - Commands: *add*, *delete*, *copy*, *move*

Basic workflow

1. Update your working copy or check out a new one
   - Commands: *co* (check out), *update*
2. Make changes
   - Use your favorite editors
   - Commands: *add*, *delete*, *copy*, *move*
3. Review the changes
   - Commands: *status*, *diff*

Basic workflow

1. Update your working copy or check out a new one
   - Commands: *co* (check out), *update*
2. Make changes
   - Use your favorite editors
   - Commands: *add*, *delete*, *copy*, *move*
3. Review the changes
   - Commands: *status*, *diff*
4. Fix any mistake
   - Command: *revert*

Basic workflow

1. Update your working copy or check out a new one
   - Commands: *co* (check out), *update*
2. Make changes
   - Use your favorite editors
   - Commands: *add*, *delete*, *copy*, *move*
3. Review the changes
   - Commands: *status*, *diff*
4. Fix any mistake
   - Command: *revert*
5. Resolve conflicts
   - Command: *resolve*

Basic workflow

1. Update your working copy or check out a new one
   - Commands: *co* (check out), *update*
2. Make changes
   - Use your favorite editors
   - Commands: *add*, *delete*, *copy*, *move*
3. Review the changes
   - Commands: *status*, *diff*
4. Fix any mistake
   - Command: *revert*
5. Resolve conflicts
   - Command: *resolve*
6. Publish changes
   - Command: *ci* (commit)

Shallow checkouts

- When using the *checkout* command, the option *--depth* allows us to selectively checkout parts of the content of a directory

Shallow checkouts

- When using the *checkout* command, the option *--depth* allows us to selectively checkout parts of the content of a directory
  - *--depth empty* to include only the immediate target (no children)

Shallow checkouts

- When using the *checkout* command, the option *--depth* allows us to selectively checkout parts of the content of a directory
  - *--depth empty* to include only the immediate target (no children)
  - *--depth files* to also include its files (but no directory children)

Shallow checkouts

- When using the *checkout* command, the option *--depth* allows us to selectively checkout parts of the content of a directory
  - *--depth empty* to include only the immediate target (no children)
  - *--depth files* to also include its files (but no directory children)
  - *--depth immediates* to include all of its immediate file or directory children

Shallow checkouts

- When using the *checkout* command, the option *--depth* allows us to selectively checkout parts of the content of a directory
  - *--depth empty* to include only the immediate target (no children)
  - *--depth files* to also include its files (but no directory children)
  - *--depth immediates* to include all of its immediate file or directory children
  - *--depth infinity* to have full recursion (default)

Common tasks

- Accessing a previous version of a file
  - Commands: *copy*, *export*, with option *-r*

Common tasks

- Accessing a previous version of a file
  - Commands: *copy*, *export*, with option *-r*
- Each revision has an associated number
  - HEAD revision is the latest
  - Using the option *-r 23* will refer to revision 23

Common tasks

- Accessing a previous version of a file
  - Commands: *copy*, *export*, with option *-r*
- Each revision has an associated number
  - HEAD revision is the latest
  - Using the option *-r 23* will refer to revision 23
- Identifying changes

Common tasks

- Accessing a previous version of a file
  - Commands: *copy*, *export*, with option *-r*
- Each revision has an associated number
  - HEAD revision is the latest
  - Using the option *-r 23* will refer to revision 23
- Identifying changes
- Merging a branch back into the trunk

Your best friend

- The most important command is . . .

Your best friend

- The most important command is . . . *help*

Your best friend

- The most important command is . . . *help*
- Calling *help* alone will print a summary of the commands and their usage

Your best friend

- The most important command is . . . *help*
- Calling *help* alone will print a summary of the commands and their usage
- Calling *help* followed by the name of a command will print a short description of the command and its options

Your best friend

- The most important command is . . . *help*
- Calling *help* alone will print a summary of the commands and their usage
- Calling *help* followed by the name of a command will print a short description of the command and its options
- Options are often useful (and sometimes necessary), but it's hard to remember them all: use *help*!