VVDN
TECHNOLOGIES

SVN

Subversion Control System.

Author:
Aashish Patel
Rakesh Pillai

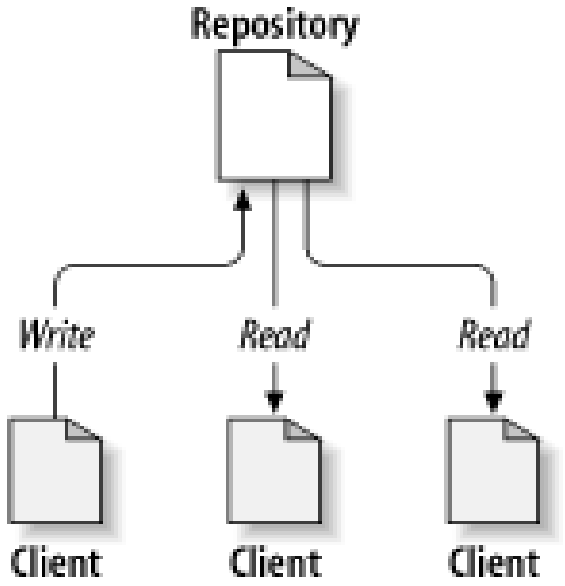
Basics

- What Version Control system does?
  - Tracks incremental versions (or revisions) of files/directories over time.

- What makes a version control system useful ?
  - It allows you to explore the changes which resulted in each of those versions .
  - facilitates the arbitrary recall of the same.

The Repository

- The core of the version control system.
- The central store of that system's data.
- Stores information in the form of a *filesystem tree.*
- *Any number of clients connect to the repository,* and read or write to these files.

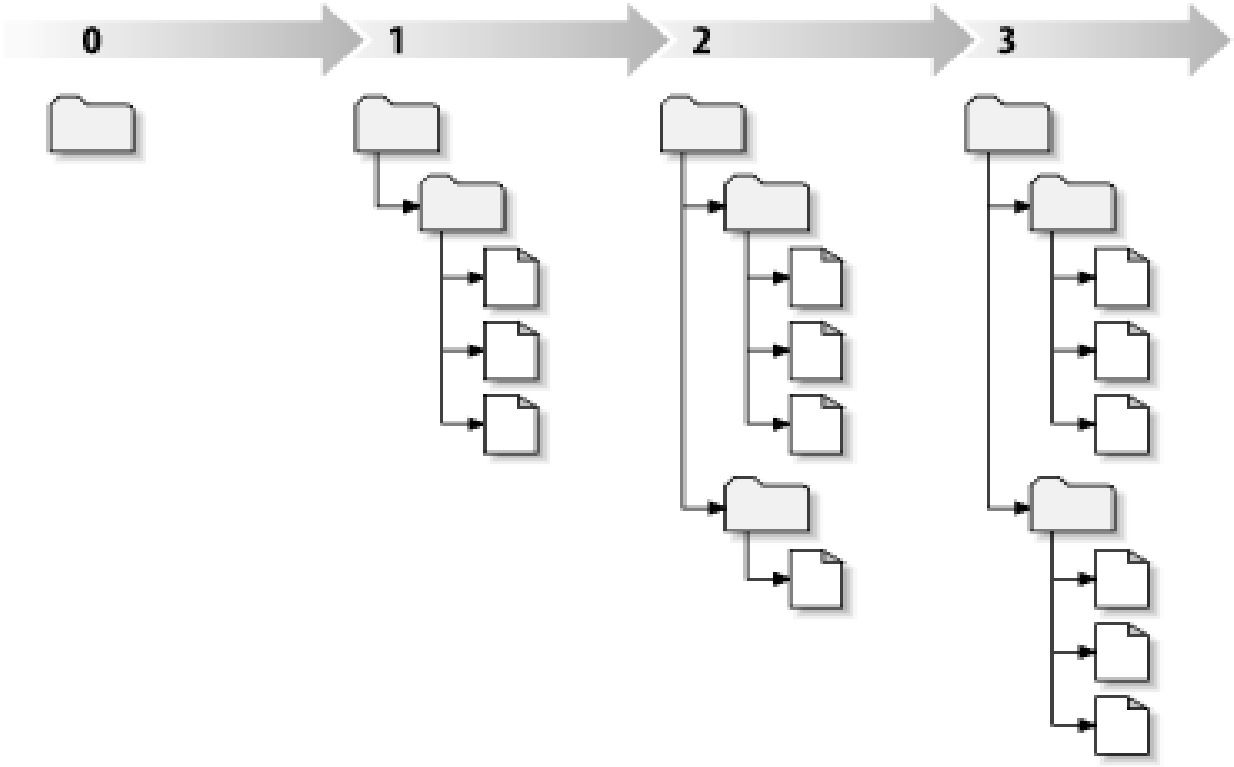


Revisions

- Each commit to repository creates a new state of the filesystem tree, called a ***revision***.
- A client commits any no. of files and directories as a single **atomic transaction.**
- That is either all changes are accepted into the repository, or none of them is.

Revisions

- Each revision is assigned a unique natural number, one greater than the number assigned to previous revision.

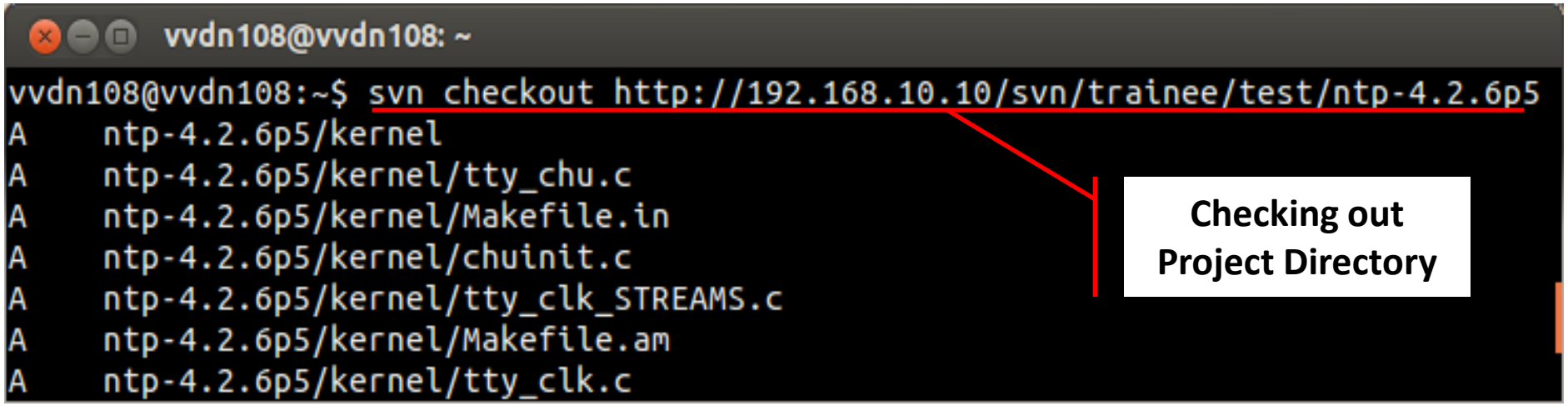


Subversion Working Copies

- It (WC) is an ordinary directory tree on your local system, containing a collection of files.
- You can edit , make changes, compile(if source file) in usual way.
- You can even have multiple working copies of the same project.

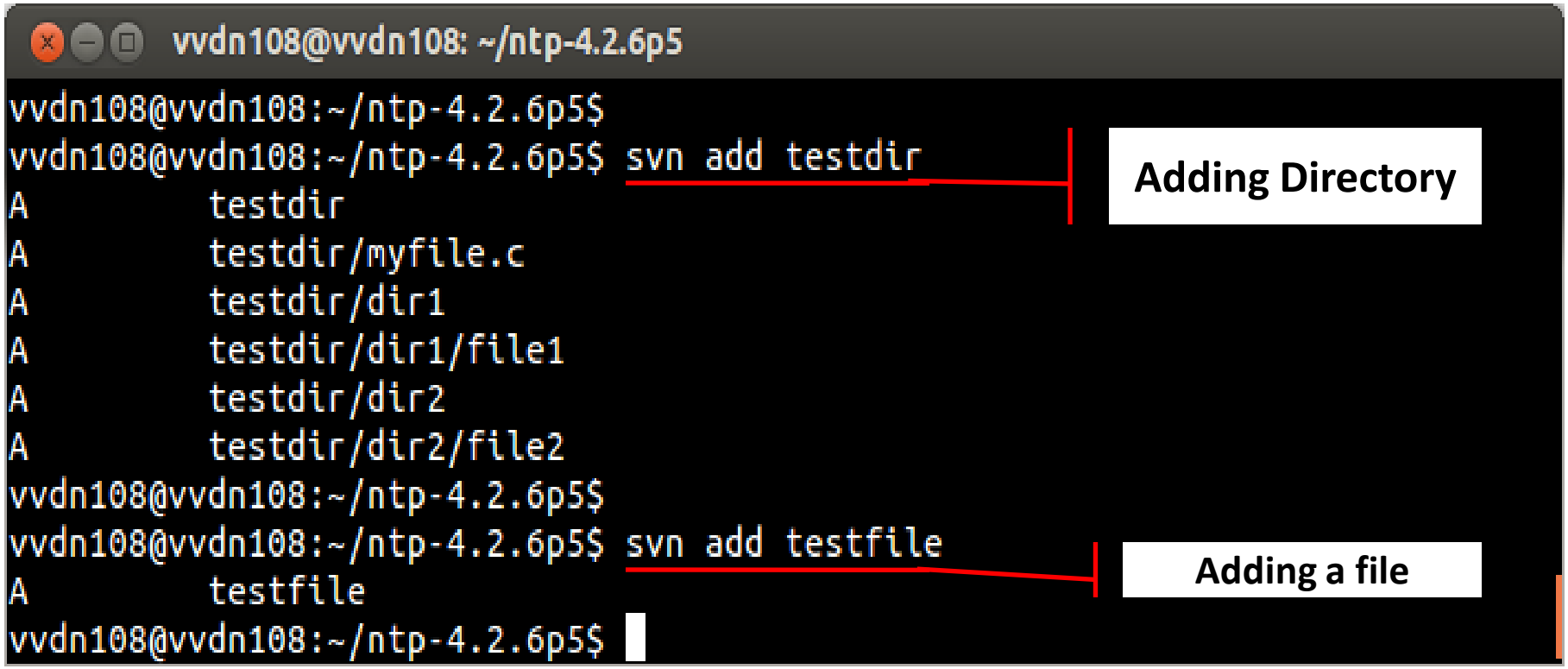
Fundamental Working Copy Interactions

- How to get a Working Copy ?
  - To get a working copy, you must ***check out*** *some subtree of the repository.*



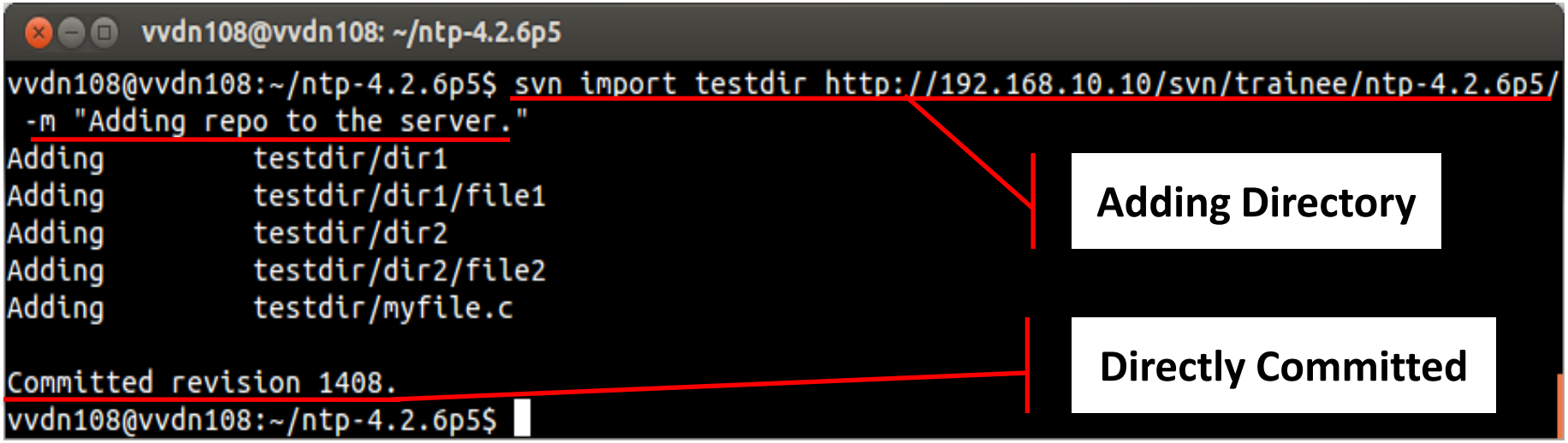
Importing Files and Directories

- You can get new files into your Subversion repository in two ways: **svn import and svn add.**



Importing Files and Directories

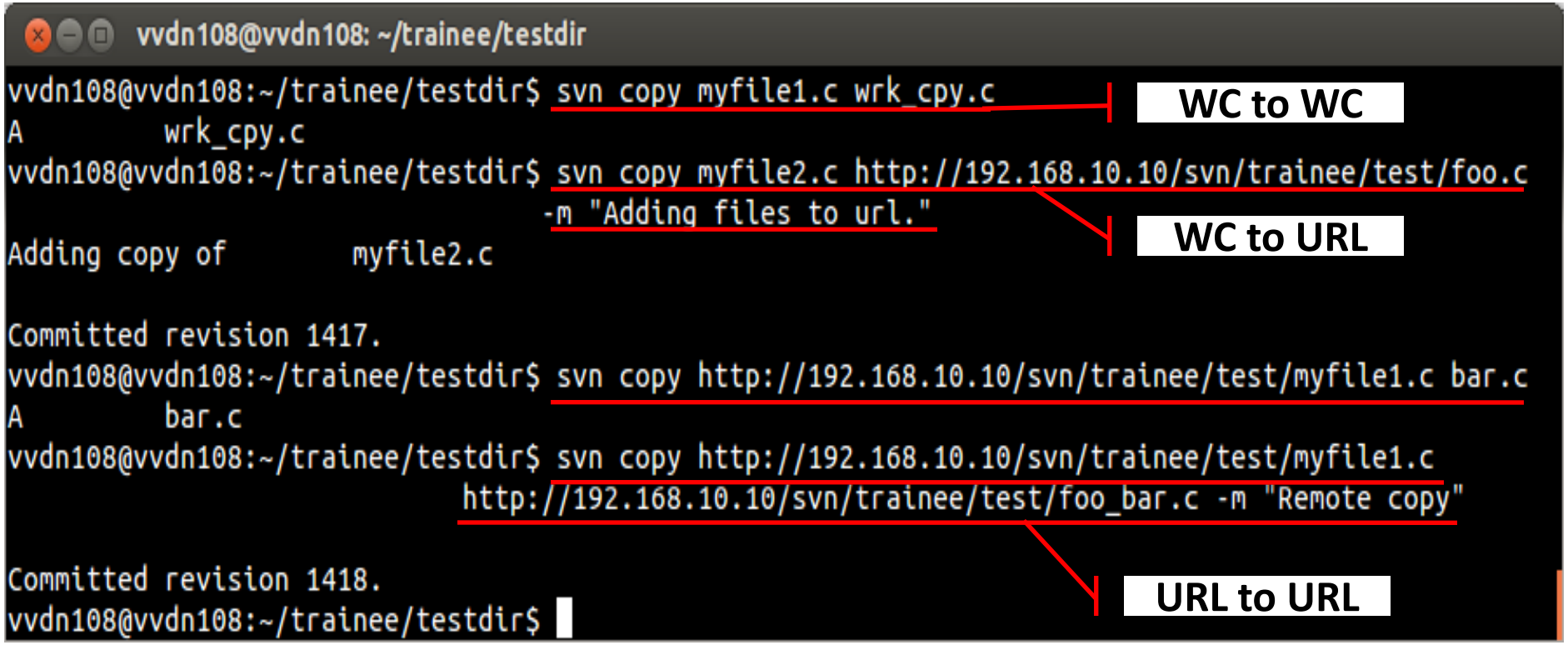
- **svn import** command used to copy an tree of files into a repository.
- It doesn't require a working copy
- Your files are ***immediately committed*** to the repository.



Other Useful SVN commands

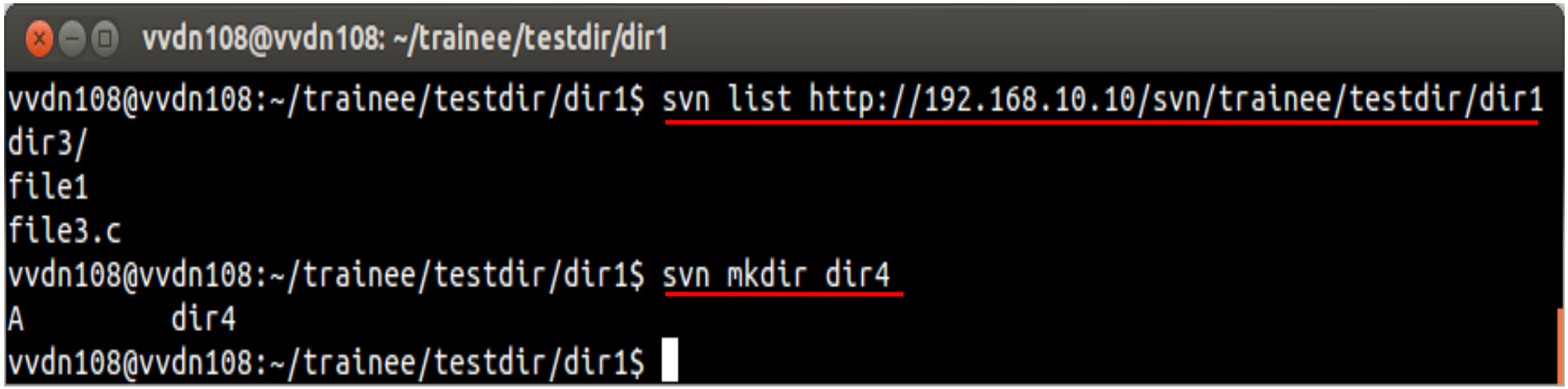
- **svn copy (cp)** — Copy a file or directory in a WC or in the repository. SRC and DST can be WC/URL .

  svn copy SRC[@REV]... DST



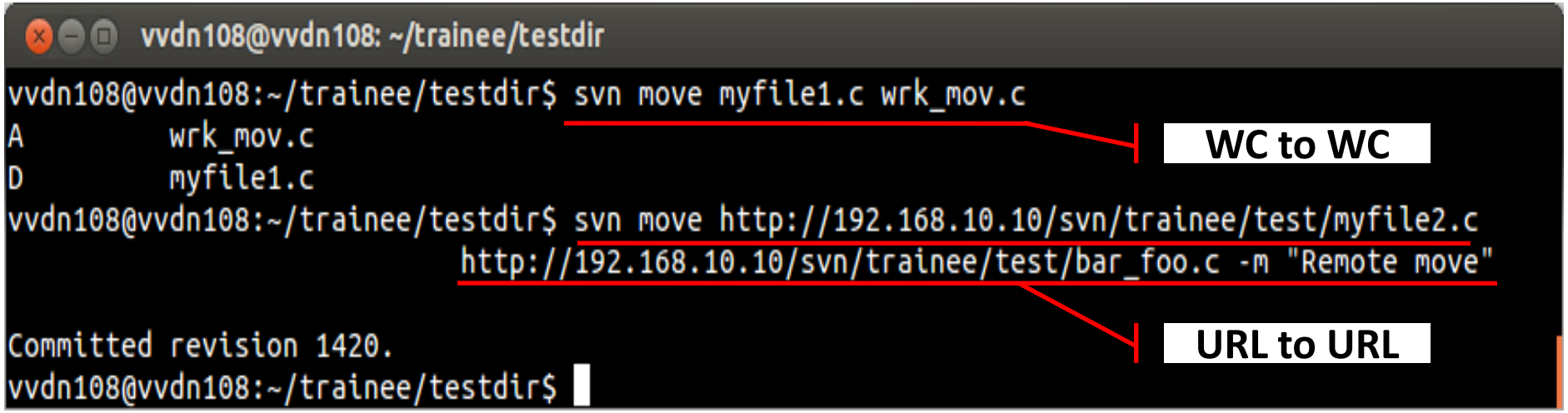
Other Useful SVN commands

- **svn list (ls)** — List directory entries in the repository.
- **svn mkdir** — Create a new directory under version control.



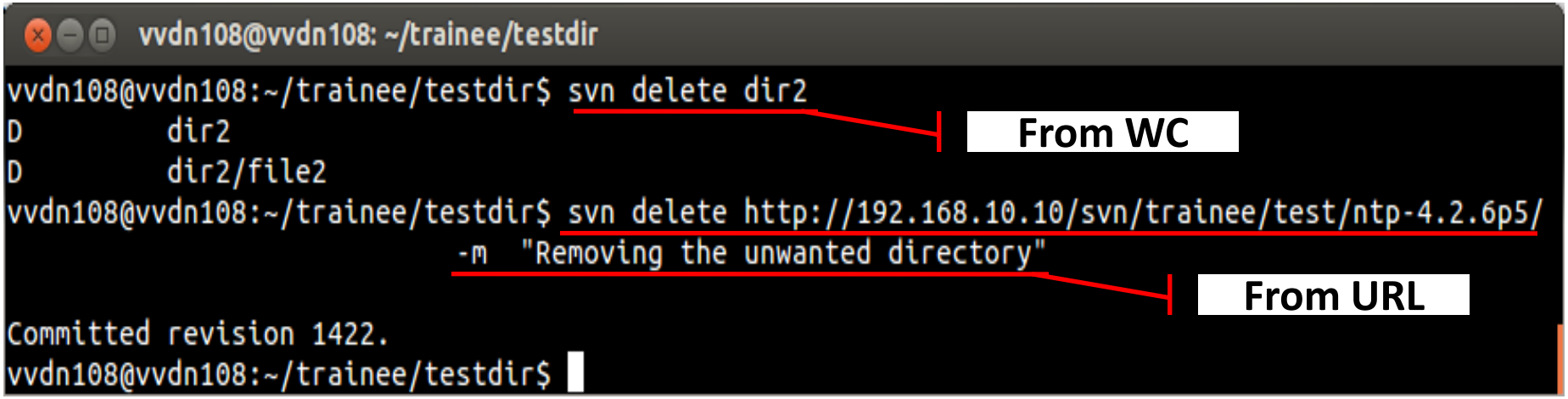
Other Useful SVN commands

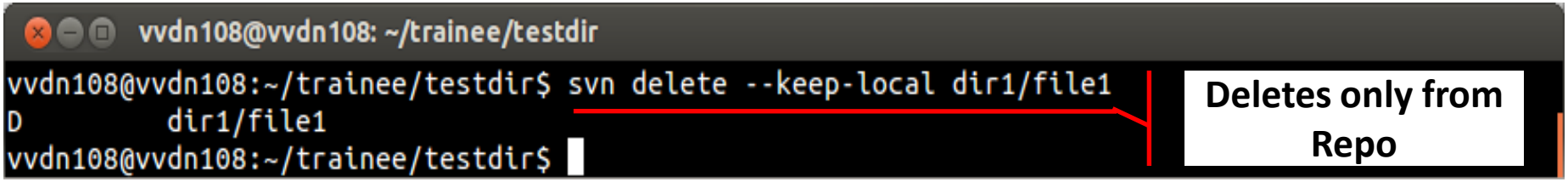
- **svn move (mv)** — Move a file or directory.
- Equivalent to an ***svn copy*** followed by ***svn delete*****.**



Other Useful SVN Commands

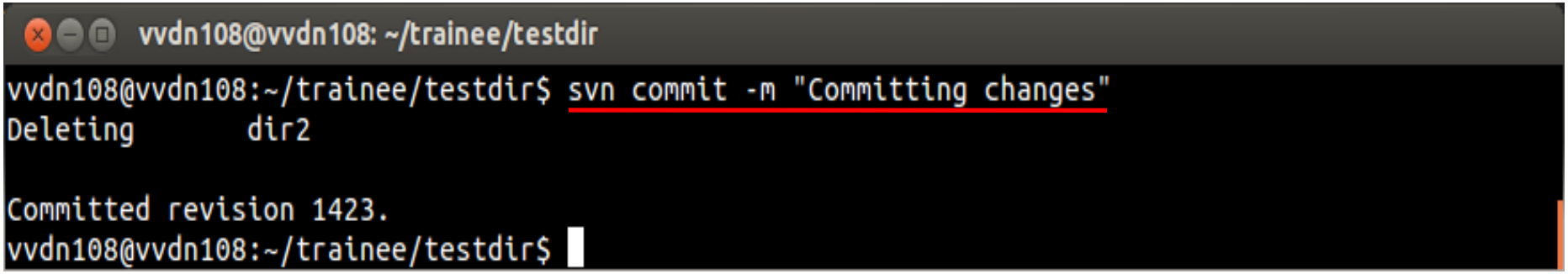
- **svn delete** (del, remove, rm) — Delete an item from a working copy or the repository.





Other Useful SVN Commands

- **svn commit (ci)** — Send changes from your working copy to the repository.
- Until we commit changes i.e. adding a file , deleting etc they won't be reflected on Version control system.



Other Useful SVN Commands

- **svn update (up)** — Update your working copy.

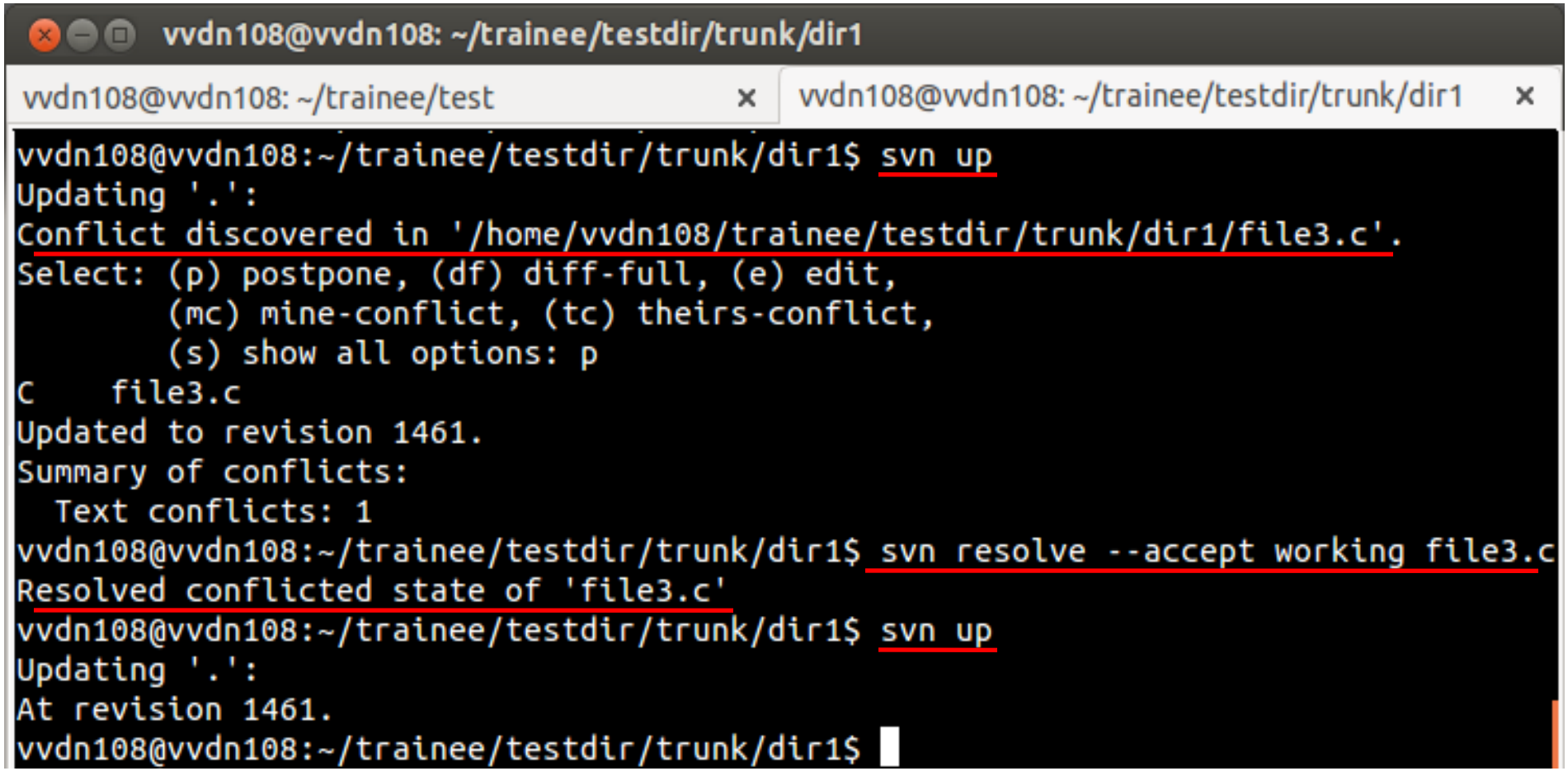
```
vvdn108@vvdn108: ~/trainee/test
vvdn108@vvdn108:~/trainee/test$ svn update
Updating '.':
D    ntp-4.2.6p5
Updated to revision 1423.
vvdn108@vvdn108:~/trainee/test$
```

- Otherwise, it synchronizes the working copy to the revision given by the --revision (-r) option.

```
vvdn108@vvdn108: ~/trainee/test
vvdn108@vvdn108:~/trainee/test$ svn update -r 1400
Updating '.':
D    .
Updated to revision 1400.
vvdn108@vvdn108:~/trainee/test$
```

Other Useful SVN Commands

- **svn resolve** — Resolve conflicts on working copy files or directories.



Other Useful SVN Commands

- **svn log** — Display commit log messages.

```
vvdn108@vvdn108: ~/trainee/test
vvdn108@vvdn108:~/trainee/test$ svn log -r 1422:1423
------------------------------------------------------------------------
r1422 | rakeshp | 2013-08-08 14:38:32 +0530 (Thu, 08 Aug 2013) | 1 line

Removing the unwanted directory
------------------------------------------------------------------------
vvdn108@vvdn108:~/trainee/test$
```

```
vvdn108@vvdn108: ~/trainee/test
vvdn108@vvdn108:~/trainee/test$ svn log http://192.168.10.10/svn/trainee/testdir
------------------------------------------------------------------------
r1423 | rakeshp | 2013-08-08 14:39:44 +0530 (Thu, 08 Aug 2013) | 1 line

Committing changes
------------------------------------------------------------------------
r1421 | rakeshp | 2013-08-08 14:36:06 +0530 (Thu, 08 Aug 2013) | 1 line

Added dir
------------------------------------------------------------------------
r1419 | rakeshp | 2013-08-08 14:09:55 +0530 (Thu, 08 Aug 2013) | 1 line

Added files
------------------------------------------------------------------------
```

Managing Project with SVN

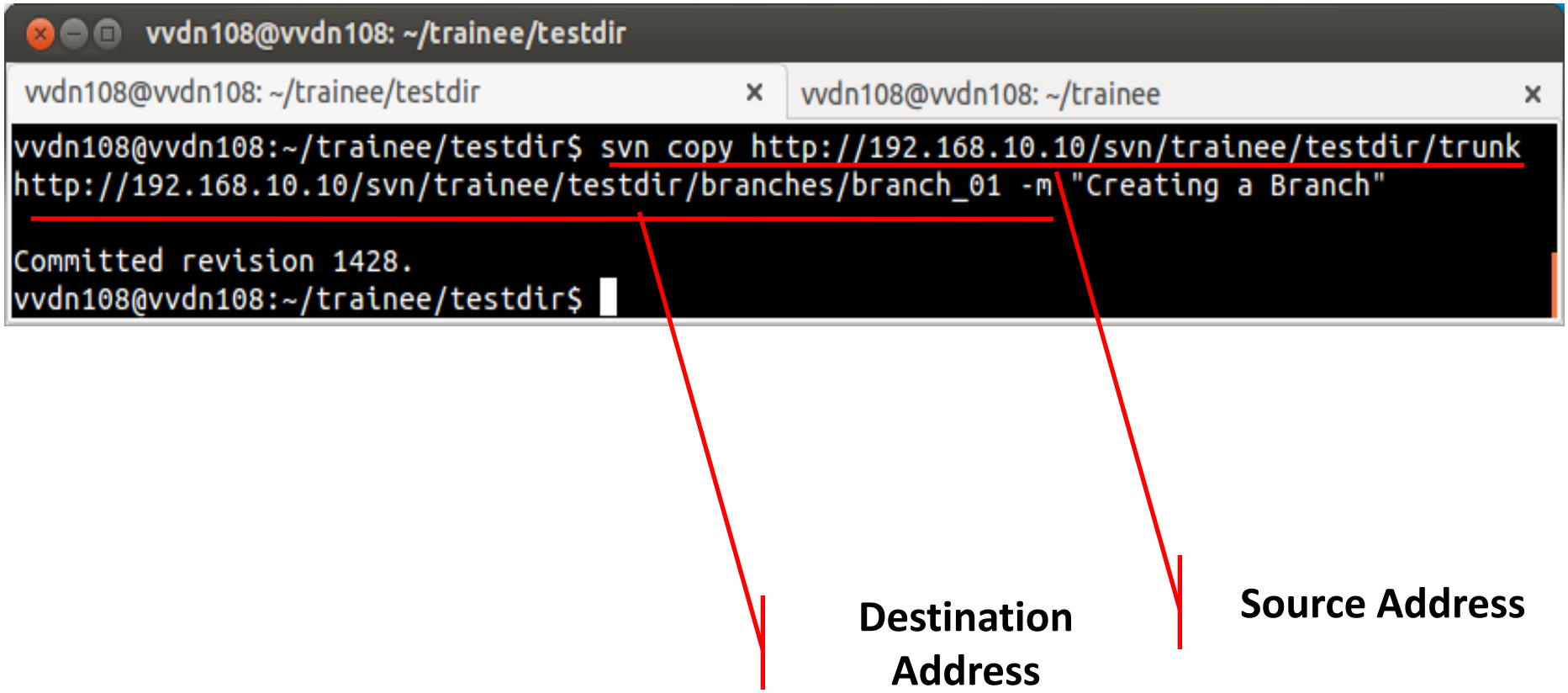
Trunks :

- Main place were stable code can be found.
- **NEVER** work directly on the trunk.
- **Unless** you have to deal with a bug which is quick and easy to fix.
- **Do not** make too many exceptions to the previous statement.
- ***Every other situation must imply the creation of a branch*.**
- Do not commit changes (from a branch merge) to the trunk which may break it .

Managing Project with SVN

How to create your own Branch ?

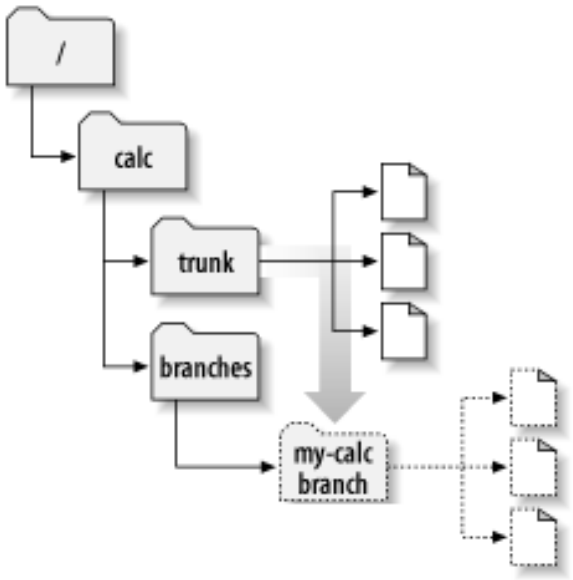
- Your own branch or a line of development, in the project repository.



Managing Project with SVN

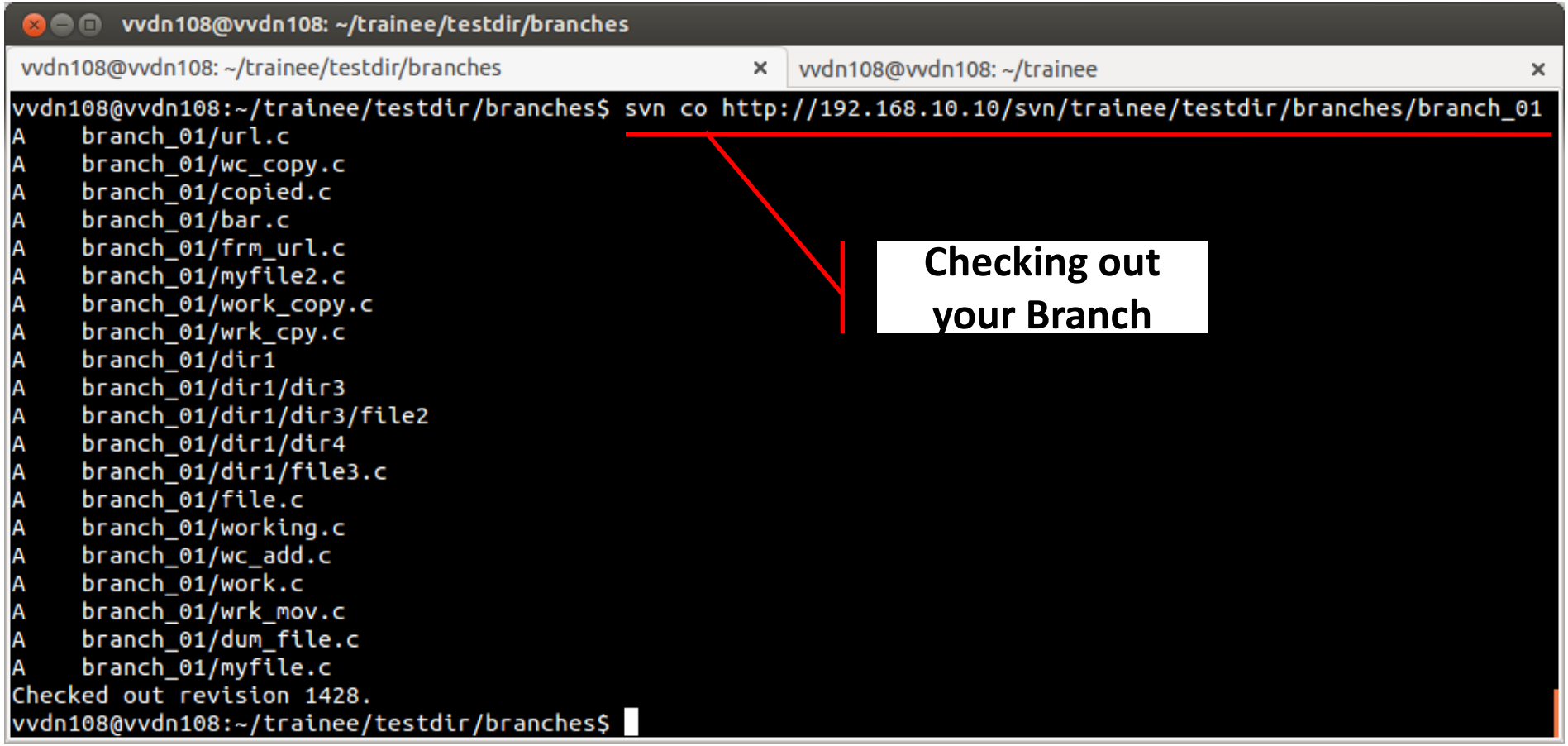
Cheap Copies:

- Subversion doesn't actually duplicate any data.
- It creates a new directory entry that points to an *existing* tree (kind of hardlink).
- It duplicates data only when it is necessary to disambiguate different versions of objects.



Managing Project with SVN

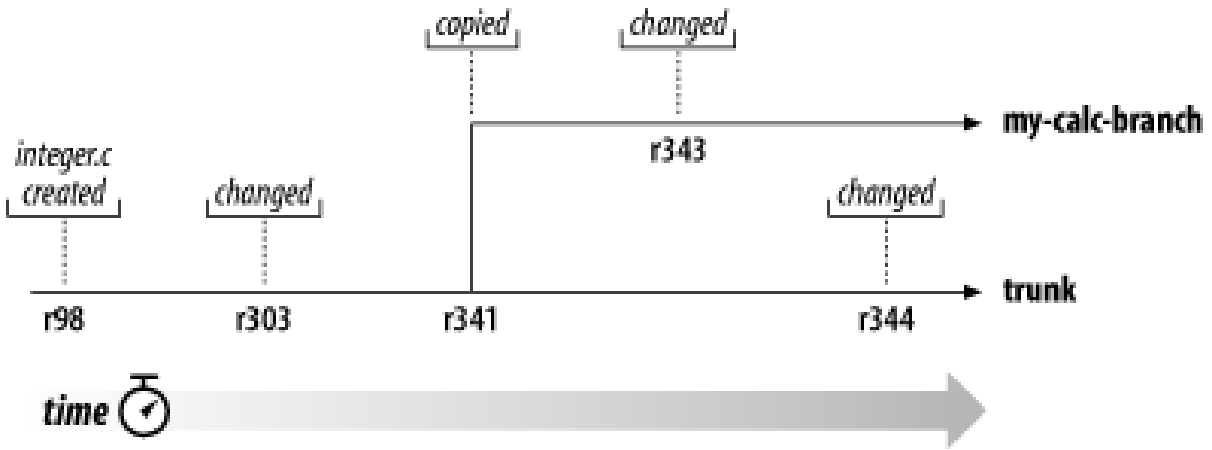
- **Working with Your Branch**
  - Check out a new working copy your branch
  - Start using it .



Managing Project using SVN

When things get interesting!!

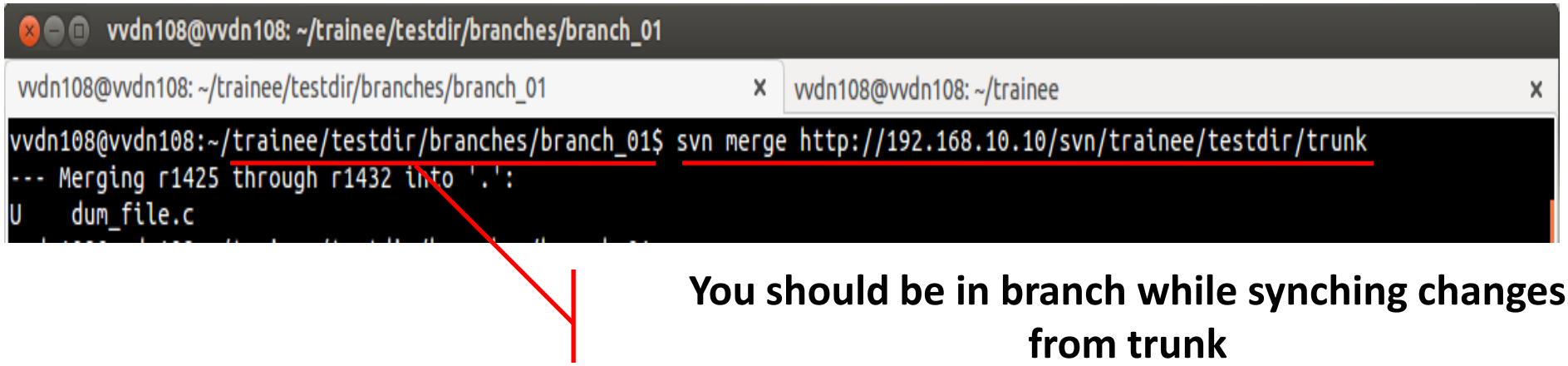
- Changes made by you in your branch will not be visible to someone working on trunk.
- Similarly changes made on trunk won't be visible to you in branch.



Managing Project Using SVN

Basic Merging:

- Keeping your Branch in synch:



You should be in branch while synching changes from trunk

- ***Committing is must after merge for changes to reflect on repo .***

```
vvdn108@vvdn108:~/trainee/testdir/branches/branch_01$ svn ci -m "Reflecting changes done in trunk to the branch"
Sending        dum_file.c
Transmitting file data .
Committed revision 1436.
```

Managing Project Using SVN

- **Reintegrating a Branch:**
  - Bring your branch into sync with the trunk again.

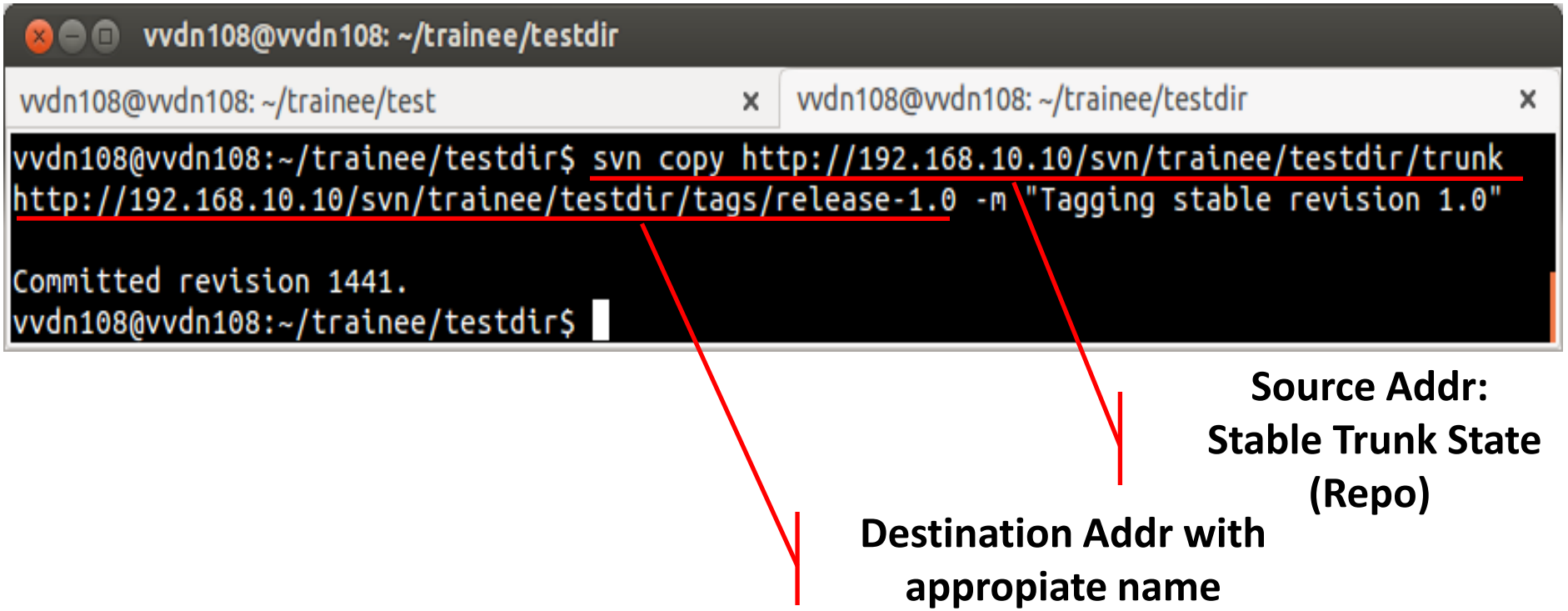
Once you have reintegrated your branch you may delete it.

Managing Project using SVN

Tag:

A tag is just a “snapshot” of a project in time.

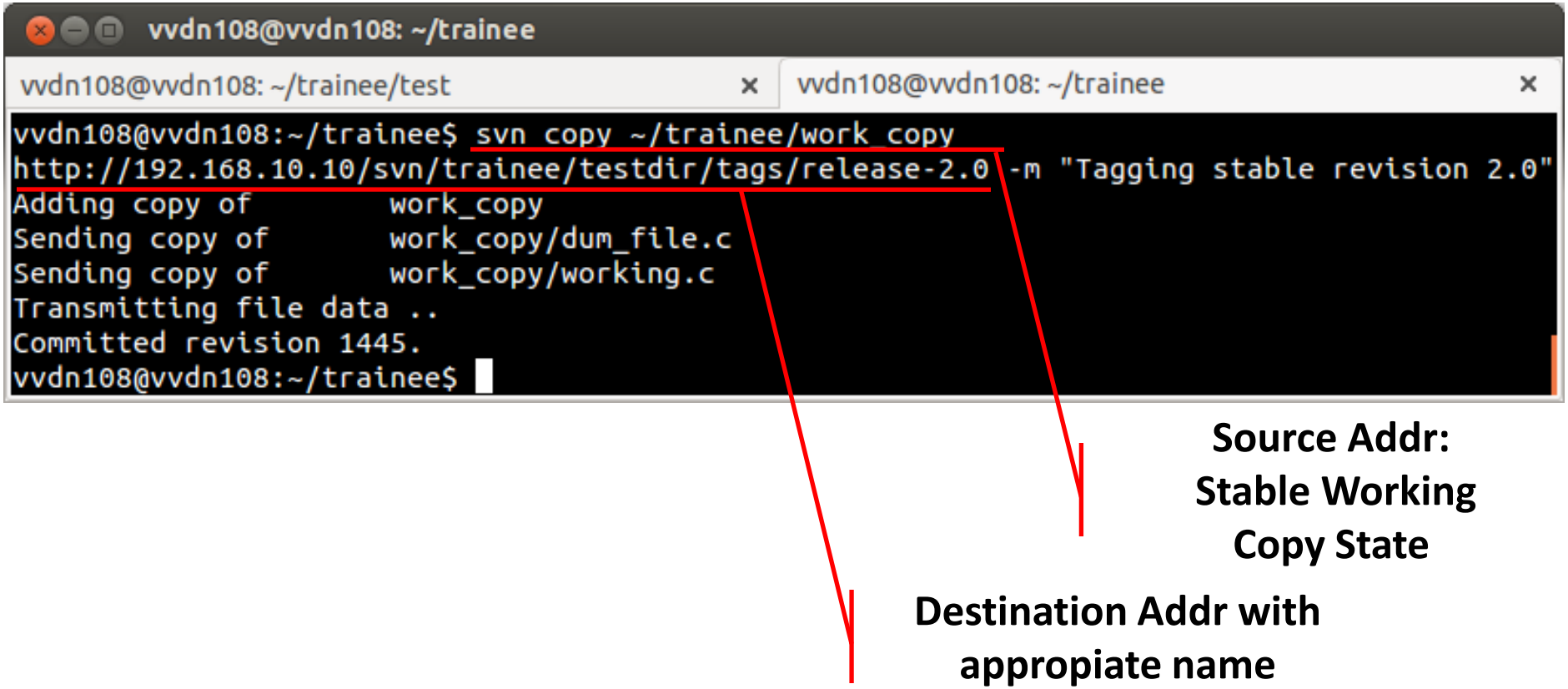
- **Creating a Simple Tag**



Managing Project using SVN

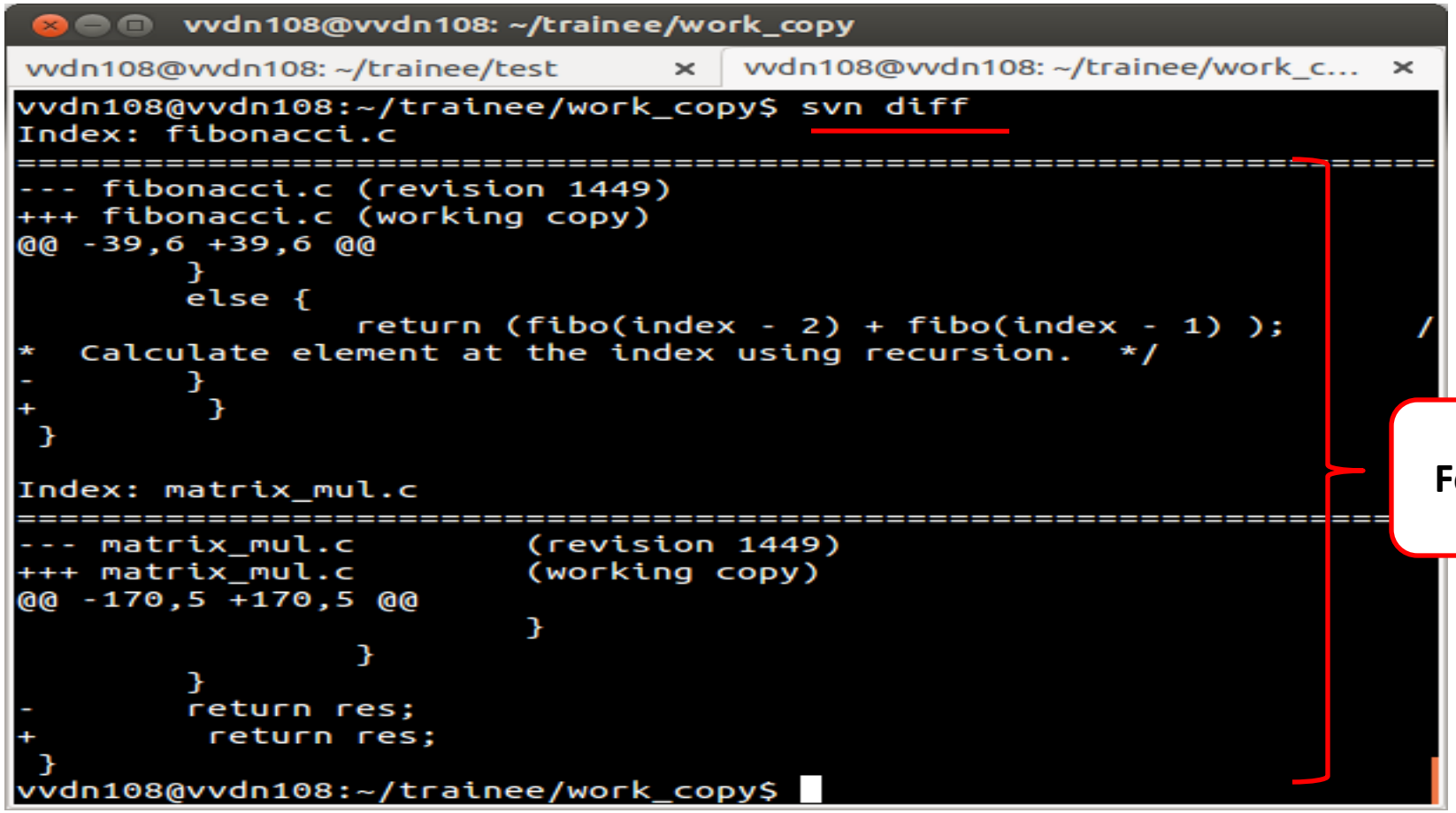
Tag:

- **Creating a Complex Tag**



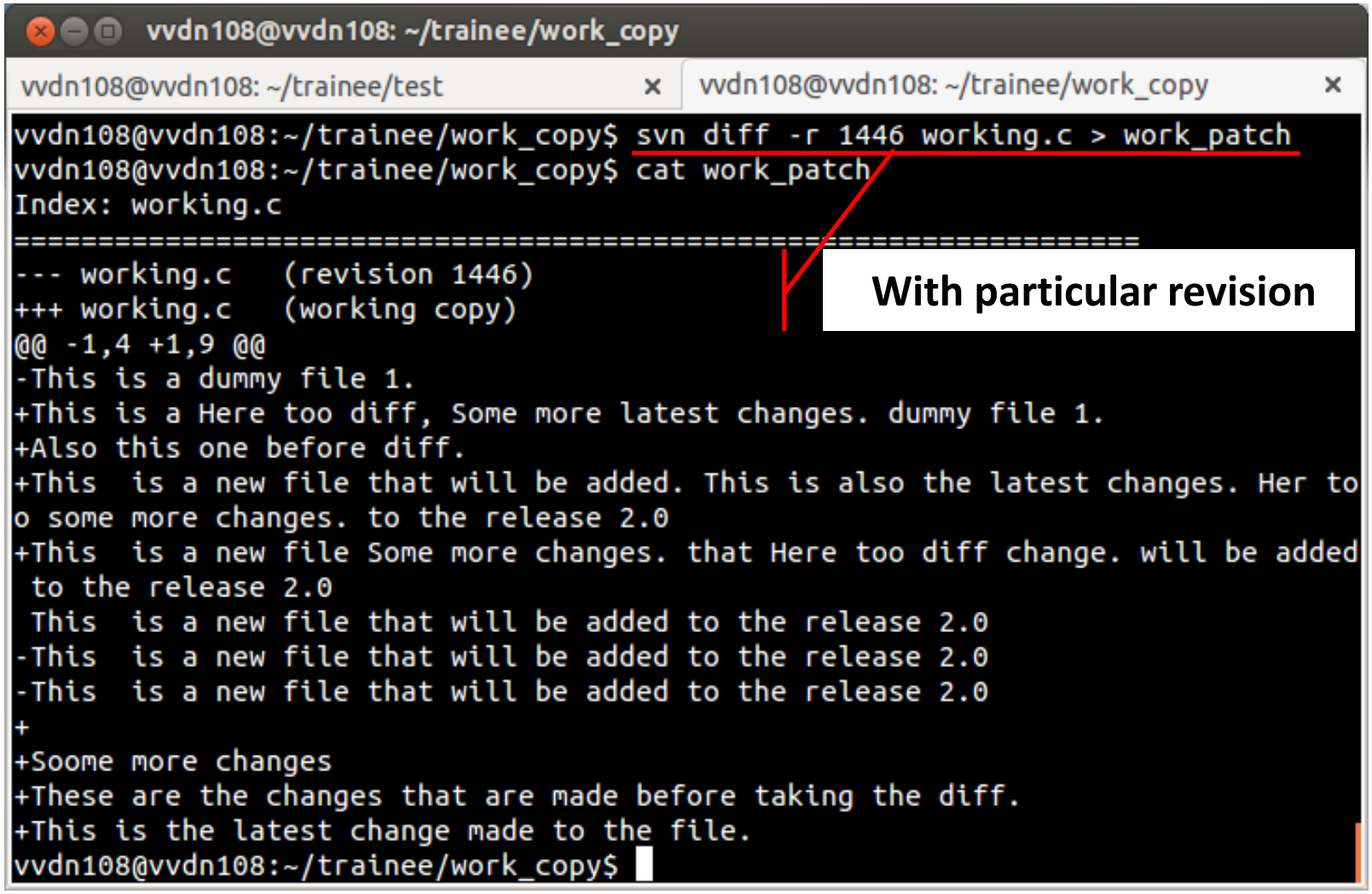
Useful commands for Project management

- **svn diff (di)** — This displays the differences between two revisions or paths.
  - Examining local changes



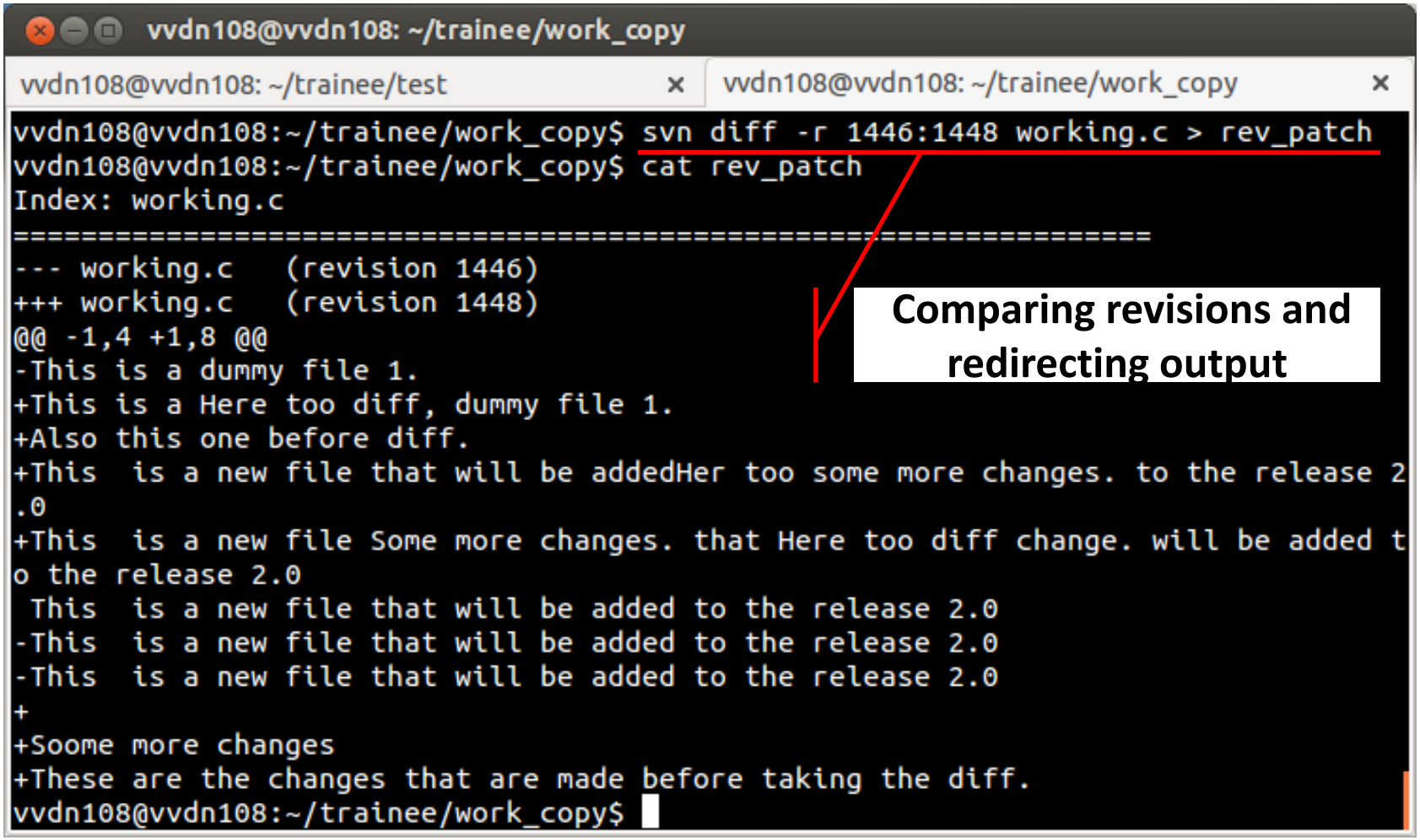
Useful commands for Project management

- **Comparing your working copy to the repository**



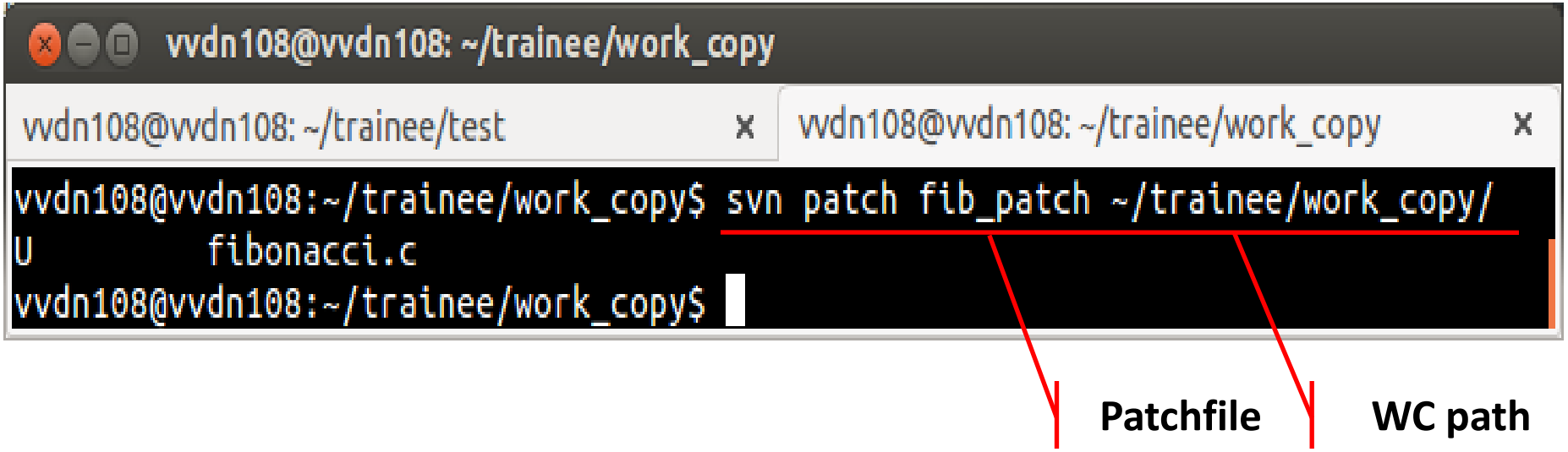
Useful commands for Project management

- **Comparing repository revisions.**

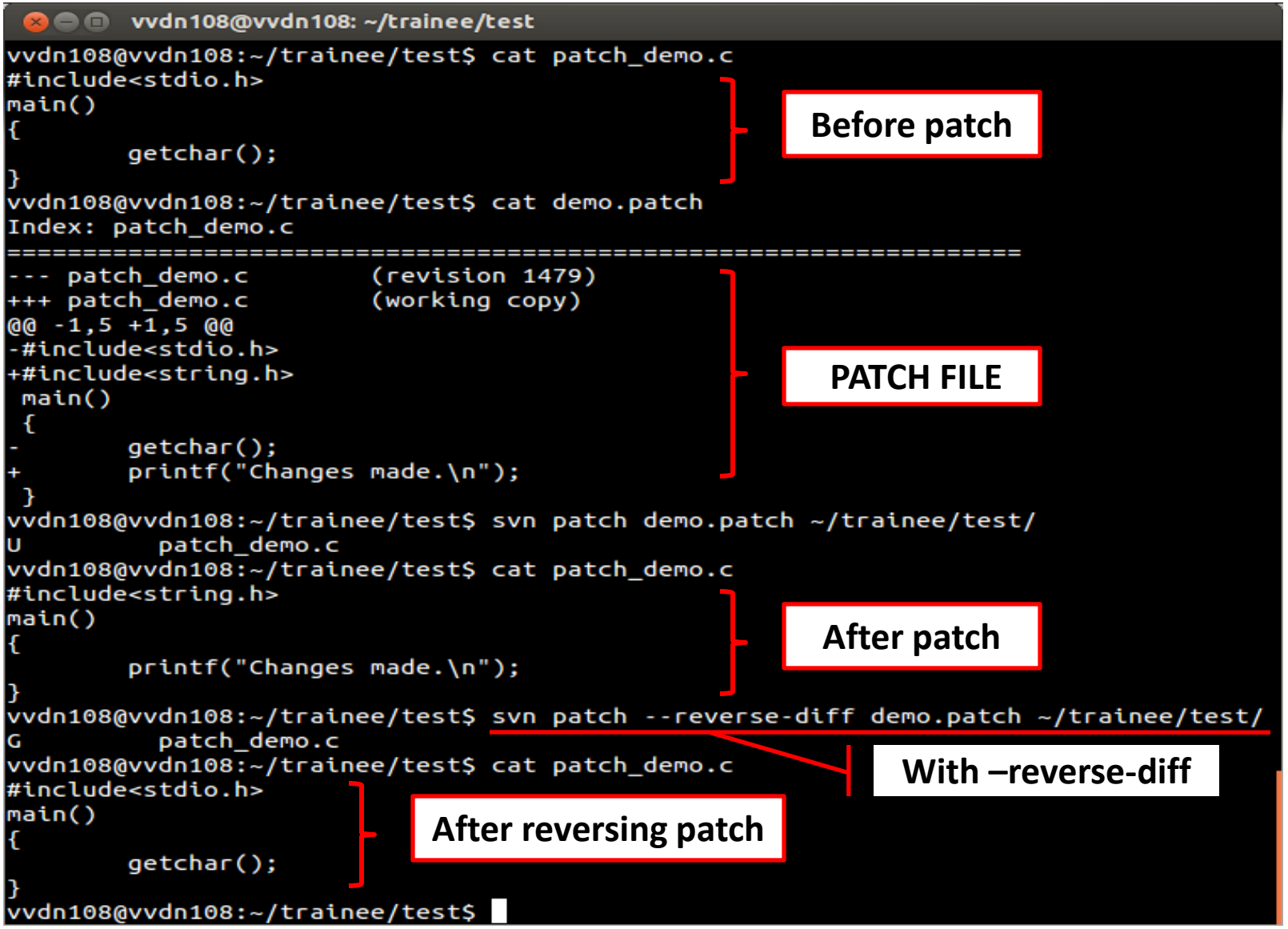


Useful commands for Project management

- **svn patch** — Apply changes represented in a unidiff patch to the working copy.

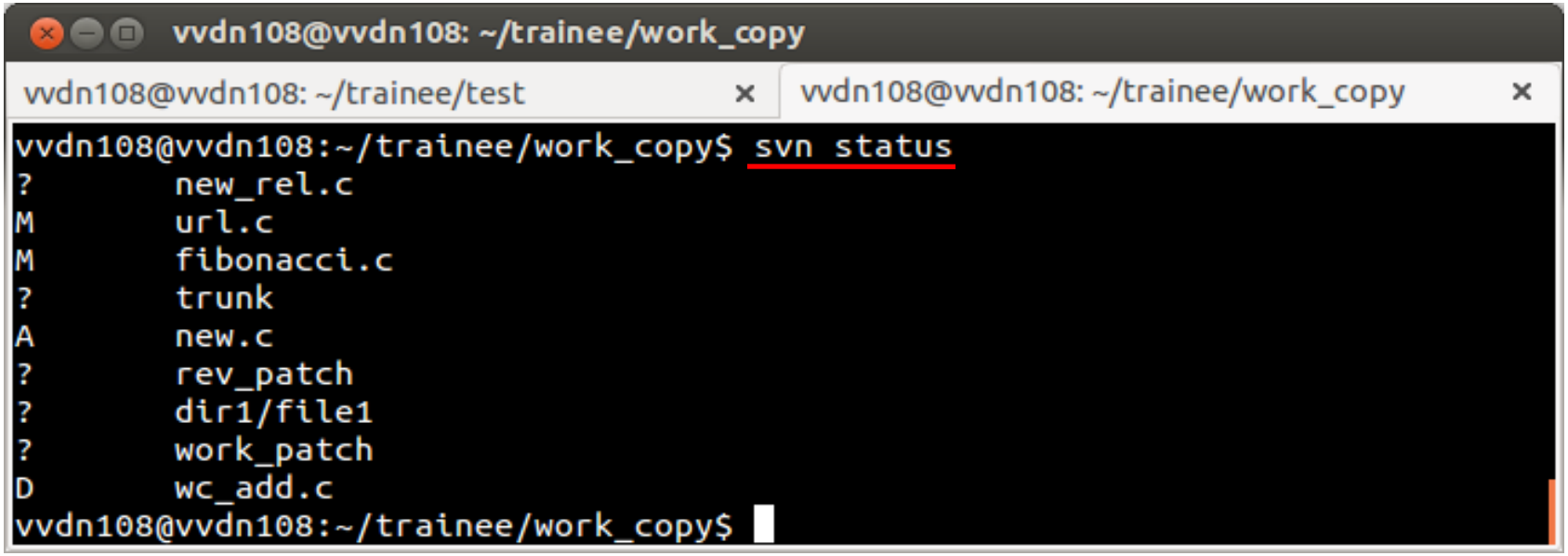


Reversing the changes made by Patch



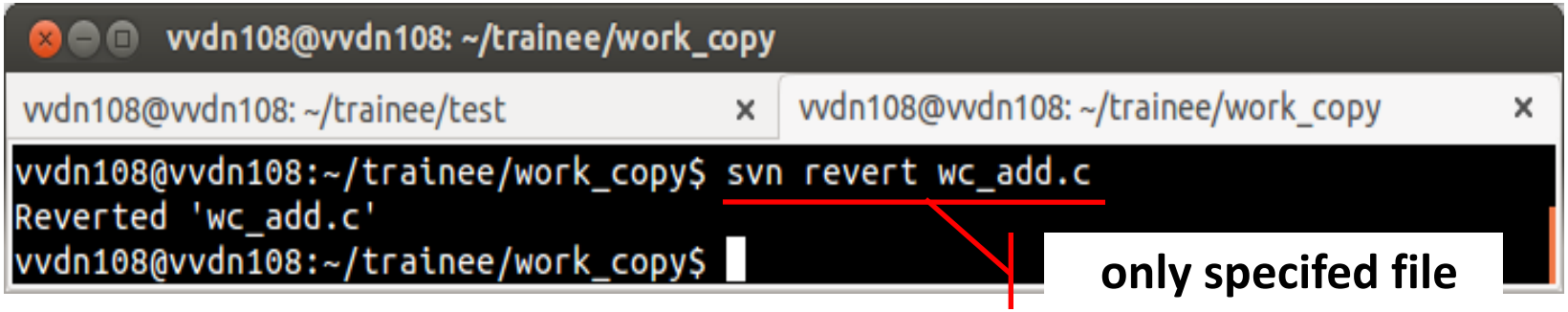
Useful commands for Project management

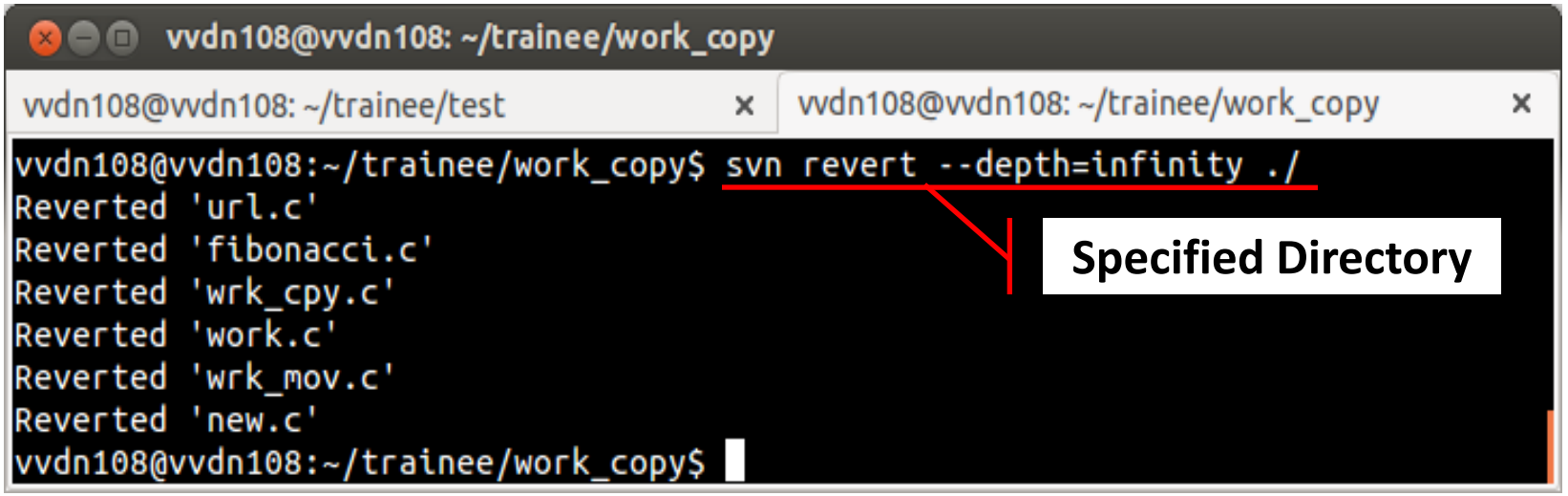
- **svn status (stat, st)** — Print the status of working copy files and directories.



Useful commands for Project management

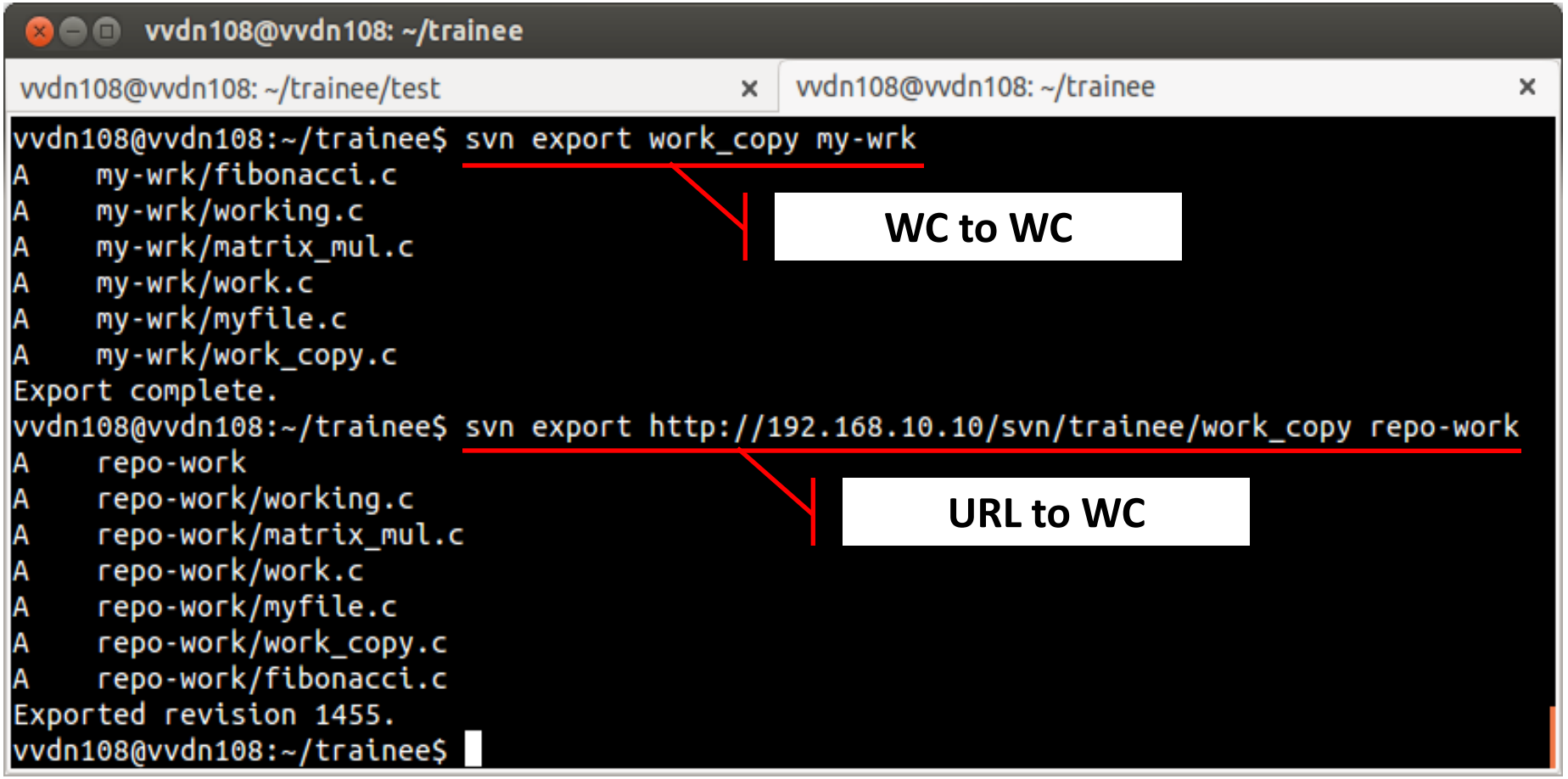
- **svn revert** — Undo all local edits.





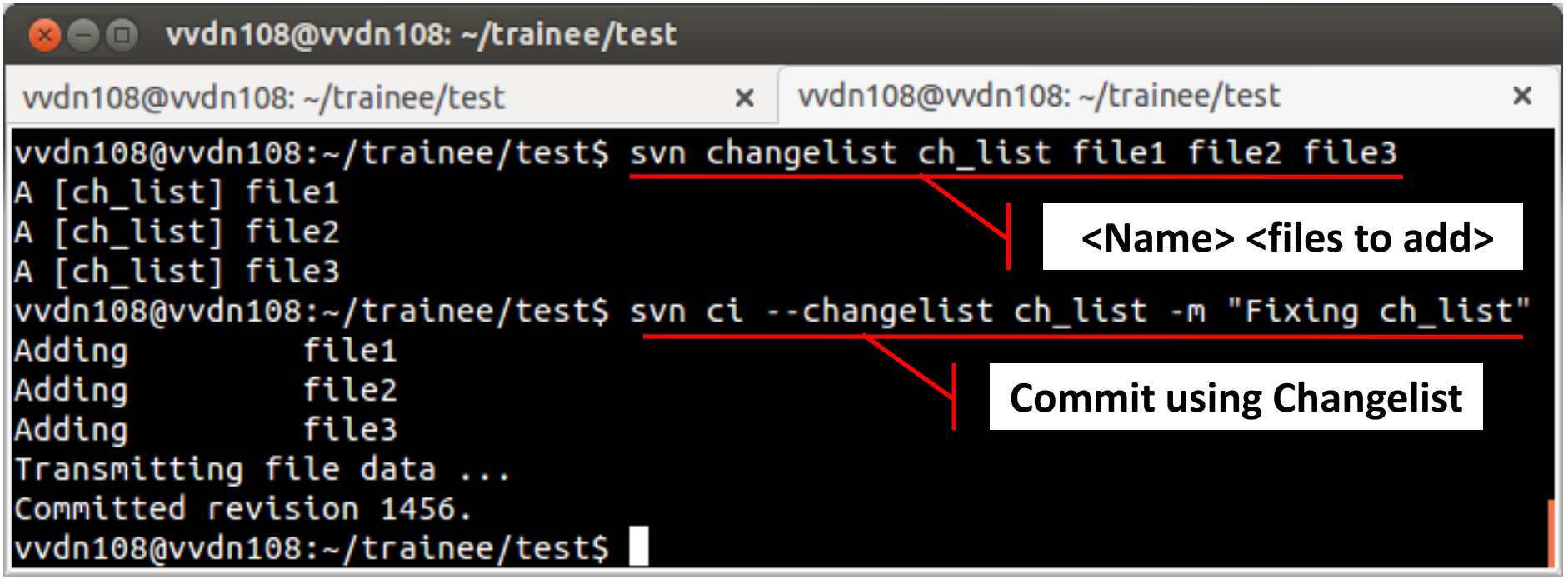
Useful commands for Project management

- **svn export** — Export a clean directory tree.



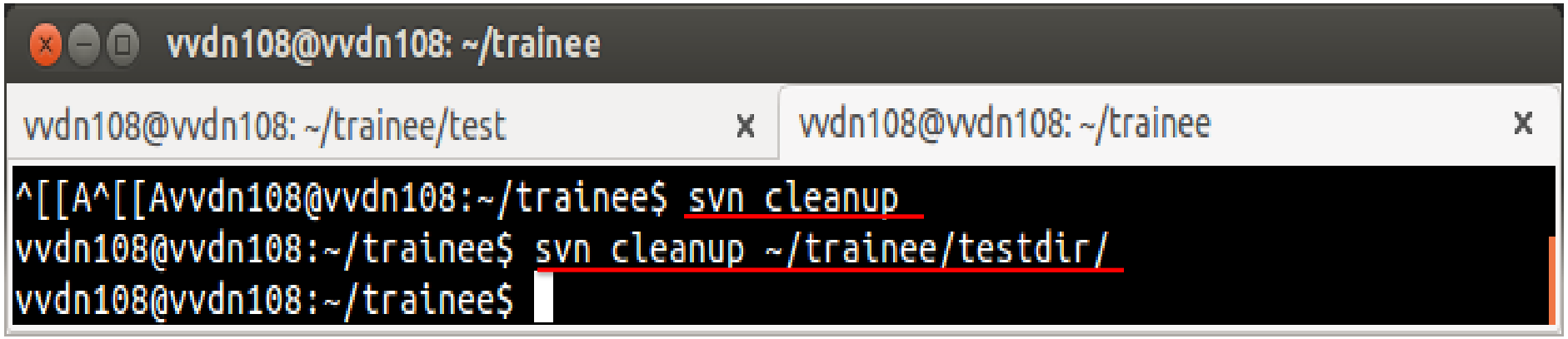
Useful commands for Project management

- **svn changelist (cl)** — Associate (or deassociate) local paths with a changelist.



Useful commands for Project management

- **svn cleanup** — Recursively clean up the working copy, removing working copy locks and resuming unfinished operations.



Some Useful Practices while using SVN

- **Follow a consistent directory structure.**

- The Subversion project officially recommends :
  - a project root representing an anchoring point for project.
    - three subdirectories: /trunk, /branches, and /tags.

- A repository may contain only one project root, or no. of them.

```
Project1/
    branches/
    tags/
    trunk/
Project2/
    branches/
    tags/
    trunk/
```

Some Useful Practices while using SVN

General practices:

- **Single commit should only reflect a single purpose**
  - Fixing of a specific bug
  - Addition of a new feature
  - or some particular task.

- **Try not to resolve multiple issues with single commit.**

- **Detailed Commit messages are helpful to others.**

Some Useful Practices while using SVN

General practices:

- **Certain commits are disallowed**
  - Cannot commit deletion of a file or directory which doesn't have working revision HEAD.
  - Cannot commit property change to directory which doesn't have working revision HEAD.

- **Should not commit binaries (executables) to SVN repo.**

- **svn update before commit changes to avoid conflicts .**

- **Commit often & Update very often.**

Some Useful Practices while using SVN

General practices:

- Whenever you need to **copy, delete, move or rename** files or folders in repository.
- Do so using **corresponding file operations** in the version control system.
- If done only on local file system **the history of changes will be lost forever.**

Some Useful Practices while using SVN

Branching Best Practices

- **Should be created for new features & major bugs.**

- **Commit often**
  - Commit small changes frequently
  - Rather than committing many small changes at once in chunk.
  - Reduces merge conflicts

- **Delete unwanted branches** –
  - A successful merge has been performed.
  - A successful reintegration merge has been performed

Some Useful Practices while using SVN

Merging Best Practices:

- **Start with a checkout or an update.**

- **Merge on logical checkpoints**
  - On reaching a certain level of stability and maturity.
  - Never merge when an experimental change has been made.

- **Use log messages**

- **Merge soon**
  - So that the team remains updated with changes you made.

- **Run Frequent ‘SVN Updates'**
  - So that you stay updated with changes made in the trunk.

Some Useful Practices while using SVN

Tagging Best Practices:

- Before merges
- Tag major releases
- Before making sweeping changes
- Use proper tag names

References

- http://svnbook.red-bean.com/nightly/en/index.html
- http://en.wikipedia.org/wiki/Apache_Subversion
- http://codefez.com/creating-projects-in-subversion-trunk-tags-branches/
- http://blog.jmfeurprier.com/2010/02/08/svn-trunk-branches-and-tags/
- http://jaxenter.com/tutorial-apache-subversion-best-practices-46037.html