

# Using Subversion for Source Code Control



Michael McLennan

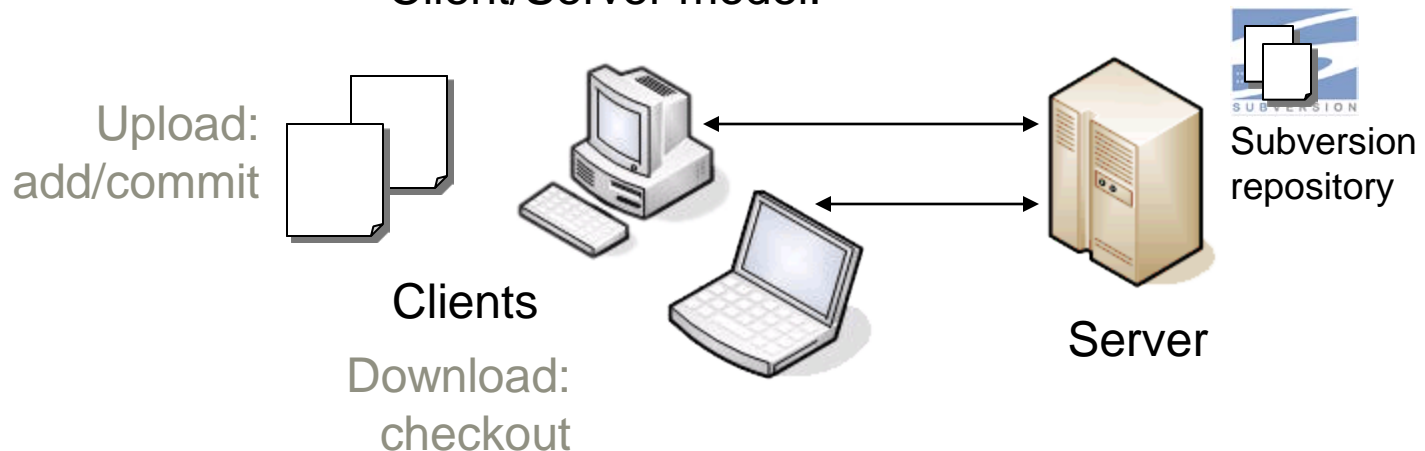
*HUBzero® Platform for Scientific Collaboration*

*Purdue University*

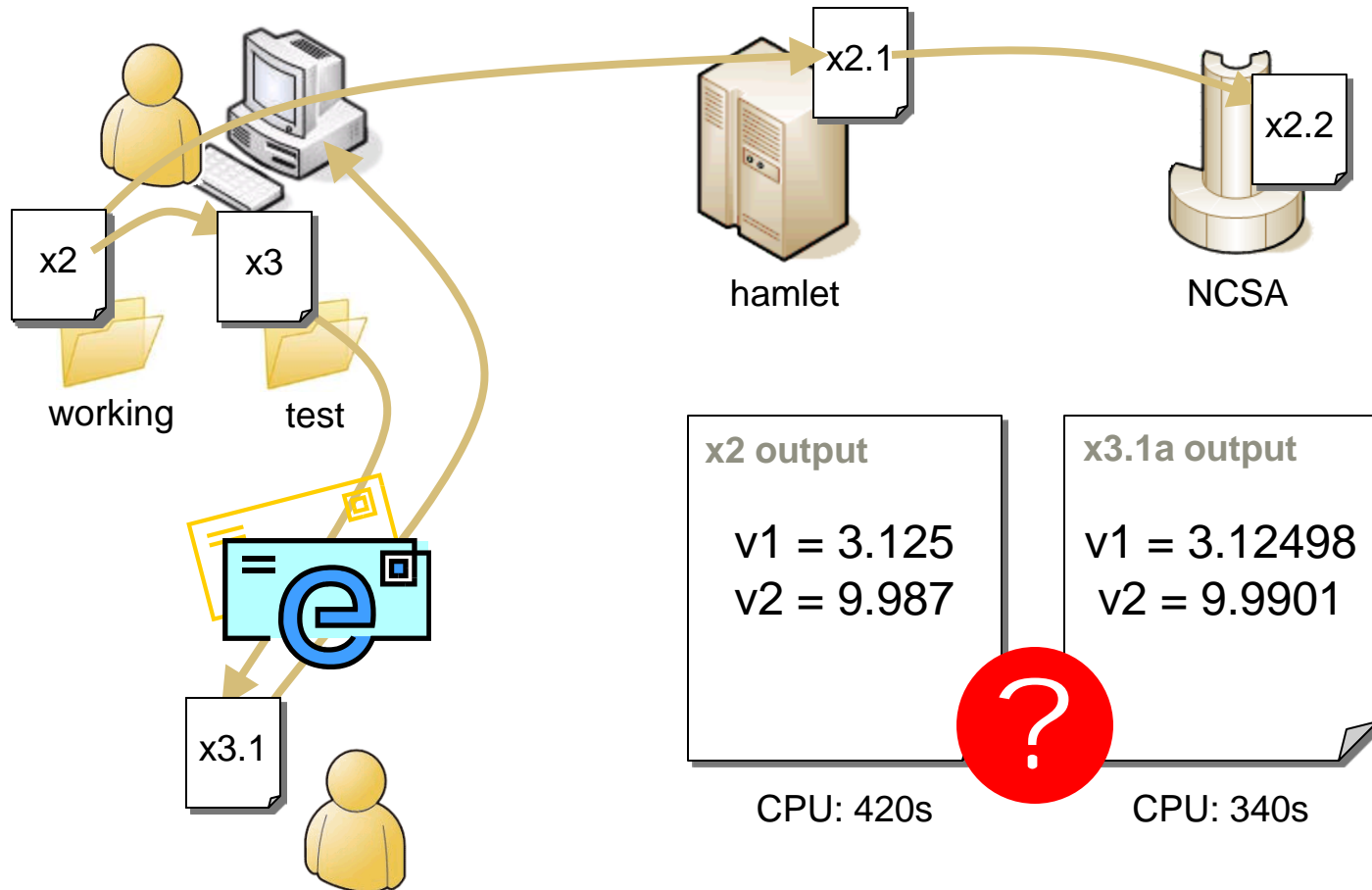


- CVS on steroids
- Created by developers at CollabNet
- In development since 2000
- In production (version 1.0) since Feb 2004
- Open source (Apache/BSD-style license)
- Unix/Linux, Win32, BeOS, OS/2, MacOS X
- Home page: <http://subversion.tigris.org/>

Client/Server model:

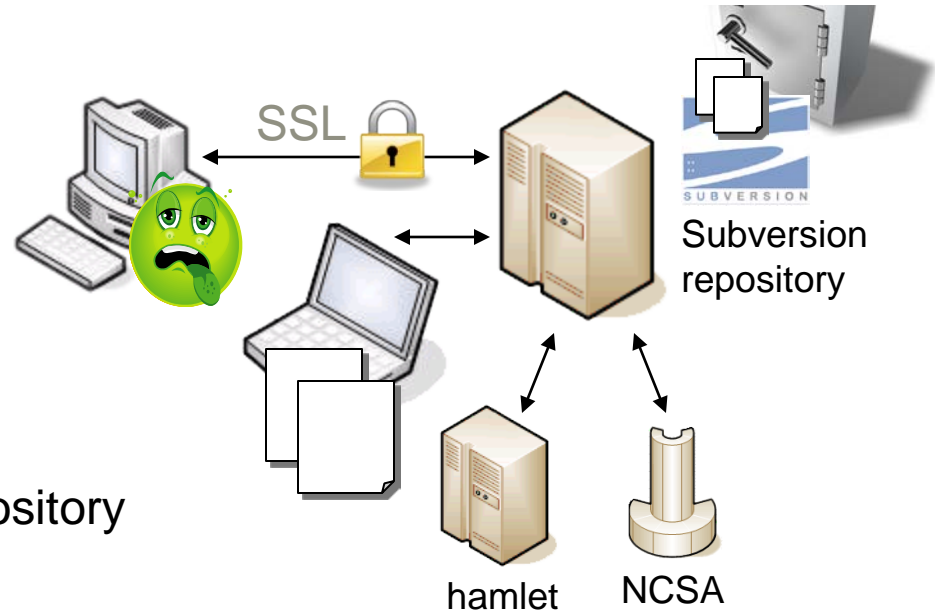


Does this sound familiar?



- **It's better than CVS**  
svn commands are nearly identical to cvs ( )
- **You'll feel more secure**  
SSL transport between client/server; repository
- **Where did I put that...**  
It's in the repository
- **Who broke the build?**  
Look at the revision history
- **Your hard drive just died**  
No problem, your code is in the repository

Date	Rev	Chgset	Author	Log Message
03/14/06 09:56:01	8	8	clarksm	Modified to facilitate d
02/20/06 13:03:51	7	7	clarksm	Fixed make install, cle
09/26/05 21:56:42	6	6	mmc	Fixed the tool explan:
09/09/05 14:19:59	5	5	clarksnano	Added missing vtk to
09/09/05 12:06:43	4	4	mmc	- Fixed the labeling o
09/07/05 14:05:59	3	3	clarksnano	Moved direct VNC inv
08/31/05 18:45:48	2	2	clarks	Added Rappture GUI .
08/31/05 17:39:13	1	1	clarks	Initial PUNCH version



## If you're using Subversion on your own machine...

- Get your files together

```
mkdir initial
```

```
mkdir initial/trunk
```

```
mkdir initial/branches
```

```
mkdir initial/tags
```

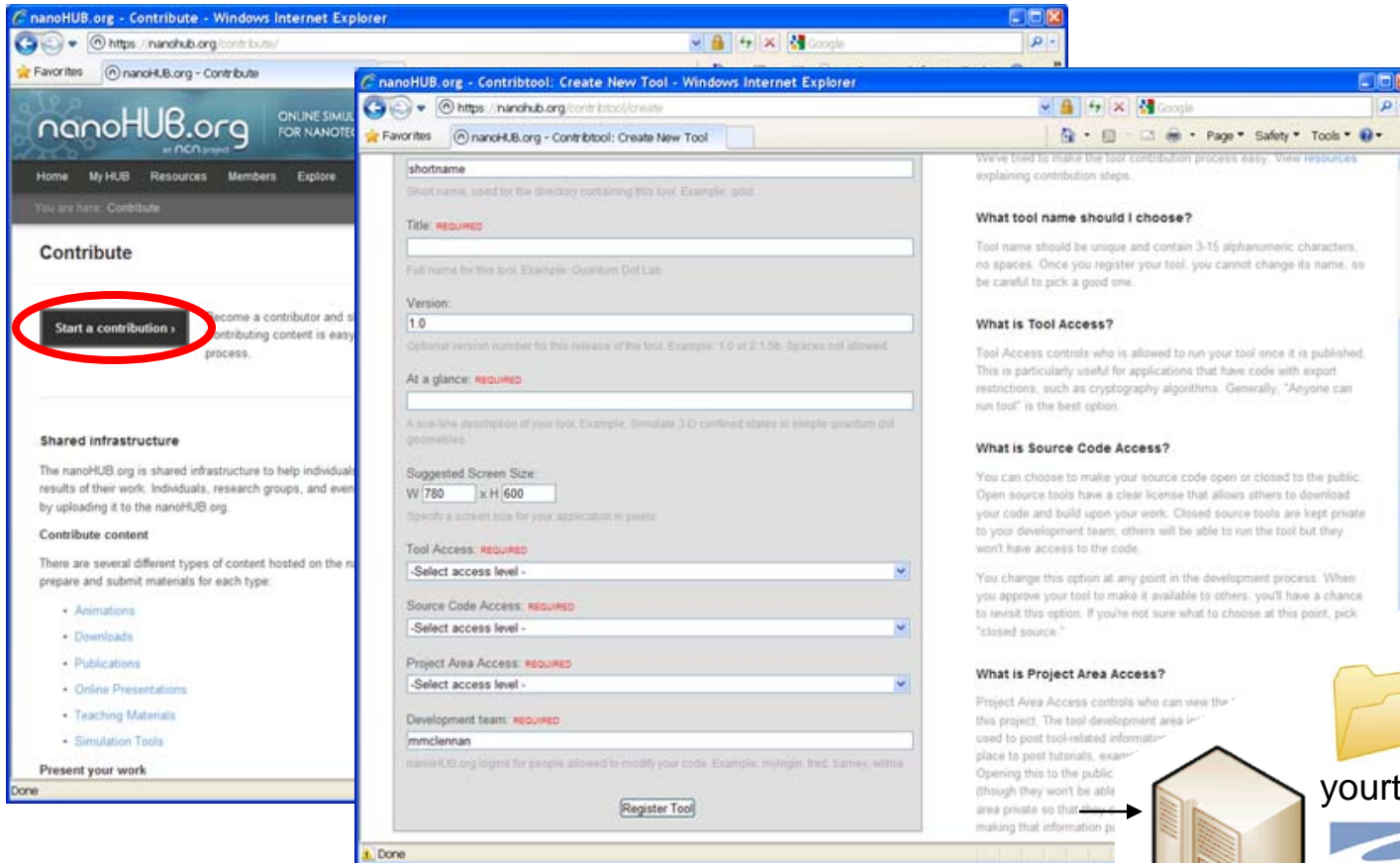
```
mv /home/src/*.c initial/trunk ← put all of your files in the trunk
```

- Create a repository and import your files

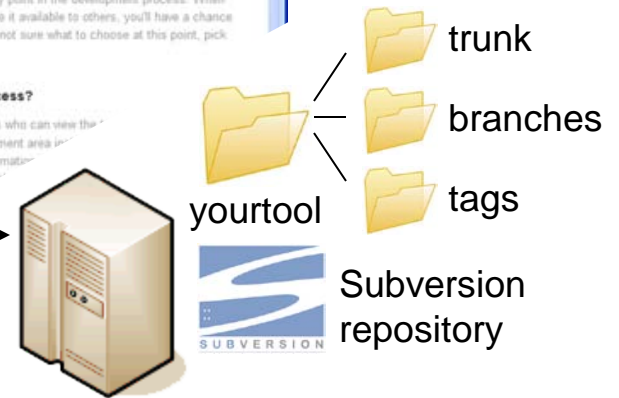
```
svnadmin create --fs-type fsfs /usr/local/svn/repo
```

```
svn import initial file:///usr/local/svn/repo -m "initial content"
```





*Once you register your tool,  
your repository is created automatically*



```
svn checkout https://nanohub.org/tools/yourtool/svn/trunk yourtool
```

```
A    yourtool/rappture
```

```
A    yourtool/doc
```

```
A    yourtool/src
```

```
A    yourtool/bin
```

```
A    yourtool/data
```

```
A    yourtool/middleware
```

```
A    yourtool/examples
```

```
Checked out revision 1.
```

```
mkdir examples/ex1
```

```
vi examples/ex1/README
```

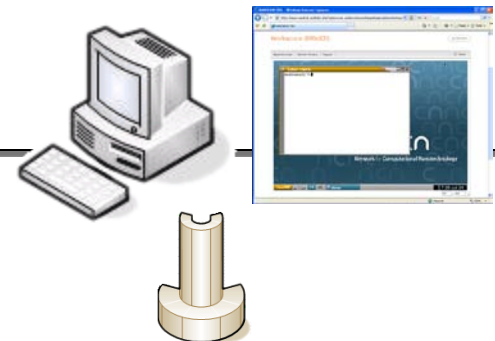
```
svn add examples/ex1
```

```
A    examples/ex1
```

```
A    examples/ex1/README
```

*Instructions in your project area  
at [wiki/GettingStarted](#)*

From any machine...



`cd yourtool` ← commit at the top level, so you don't miss anything

`svn status`

```
A    examples/ex1
A    examples/ex1/README
?    a.out
```

brings up your favorite editor:

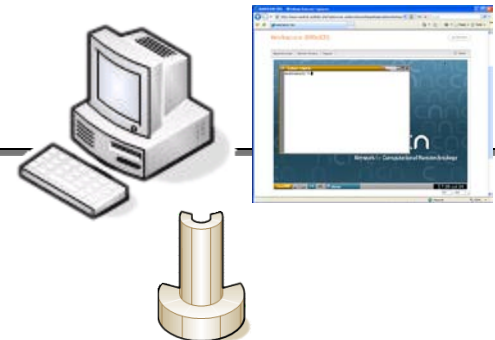
Created ex1 directory and added the README file.

Date	Rev	Chgset	Author	Log Message
07/25/07 21:31:14	2	2	mmc	Created ex1 directory and added the README file.
07/25/07 21:23:09	1	1	root	initial directory structure

`svn commit`

```
Adding      examples/ex1
Adding      examples/ex1/README
Transmitting file data .
Committed revision 2.
```

Instructions in your project area  
at [wiki/GettingStarted](#)



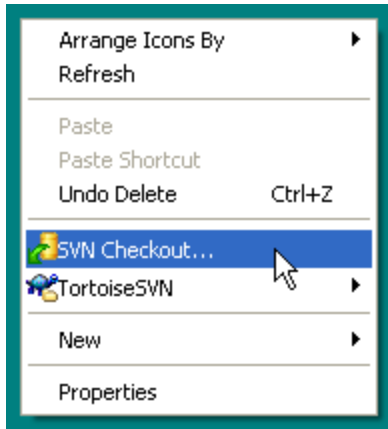




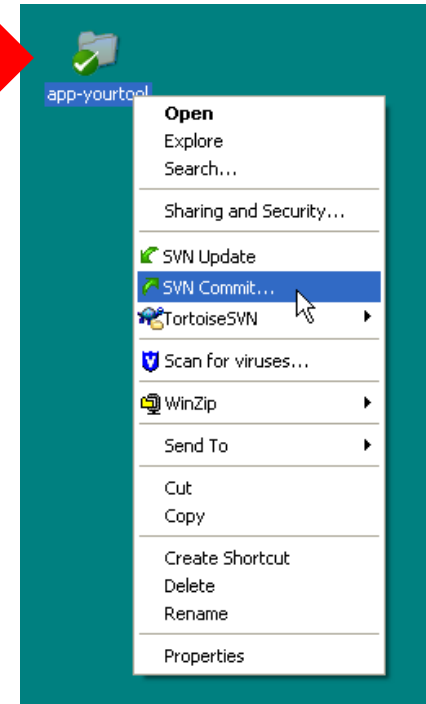
More info:

<http://tortoisesvn.tigris.org/>

Puts svn commands onto the right-mouse-button menu:



Like any other file on your Desktop, but icon shows Subversion control

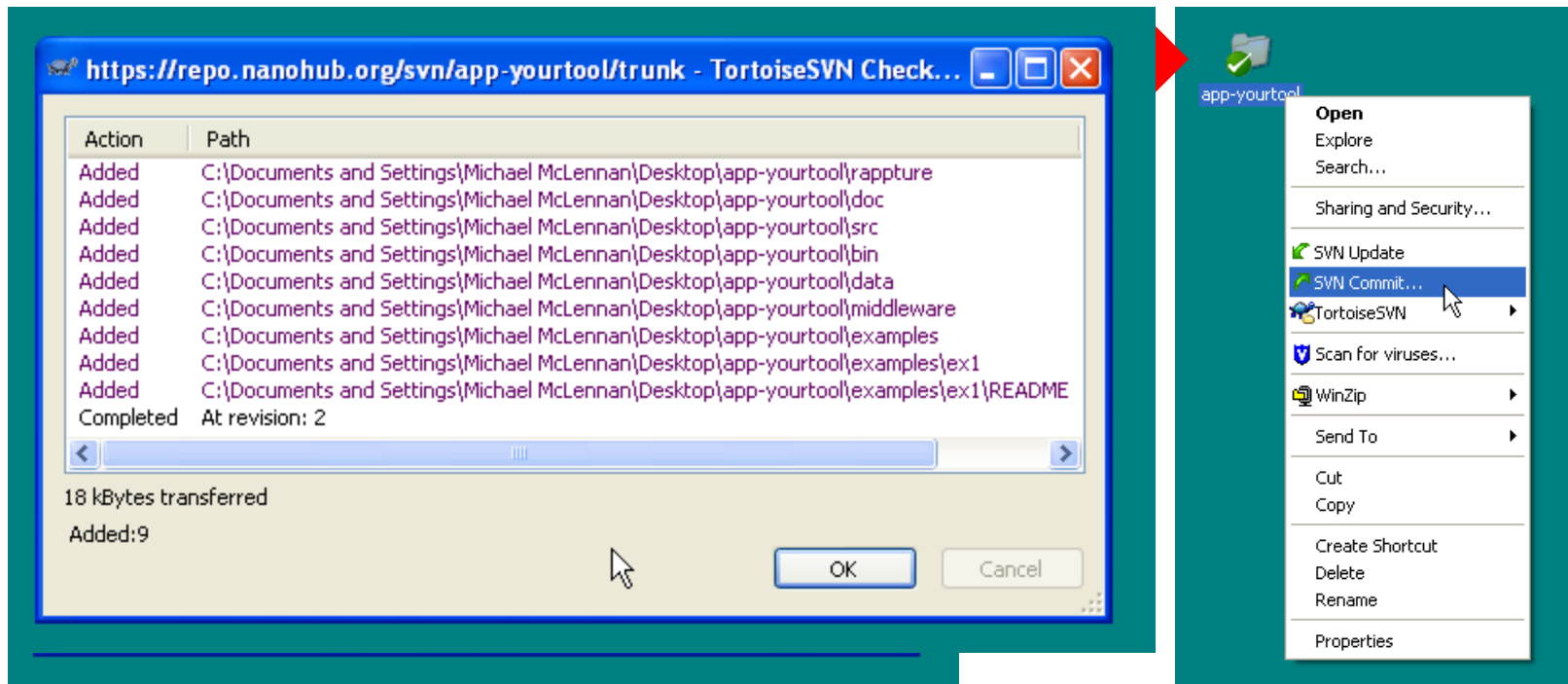




More info:

<http://tortoisesvn.tigris.org/>

Puts svn commands onto the right-mouse-button menu:



```
cd examples/ex1
svn mv README README.txt
A  README.txt
D  README
```

brings up your favorite editor:

Moved some files around.

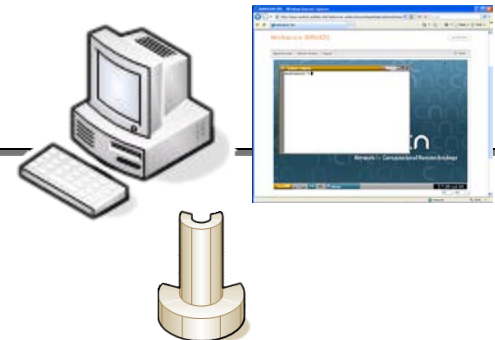
```
cd ../../
svn delete doc
D  doc
```

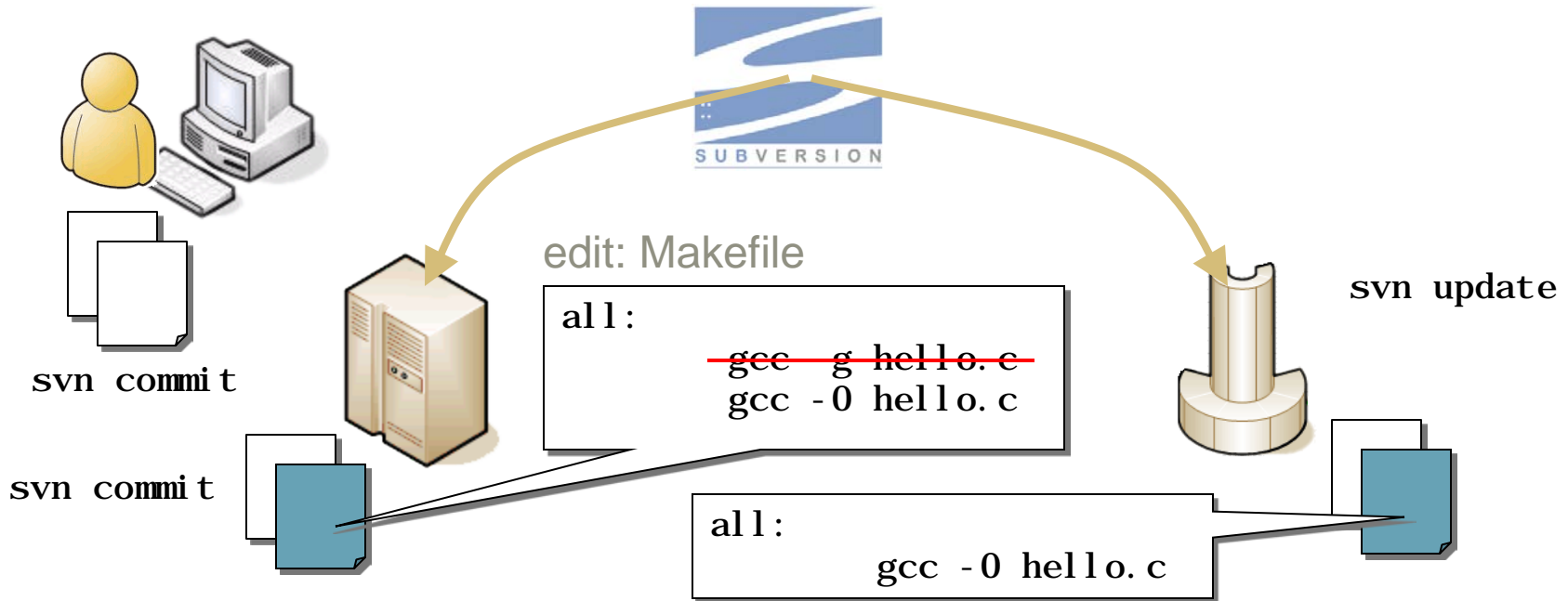
Date	Rev	Chgset	Author	Log Message
07/26/07 09:46:35	3	3	mmc	Moved some files around.
07/25/07 21:31:14	2	2	mmc	Created ex1 directory and added the README file.
07/25/07 21:23:09	1	1	root	initial directory structure

```
svn status
D  doc
A  + examples/ex1/README.txt
D  examples/ex1/README
```

```
svn commit
```

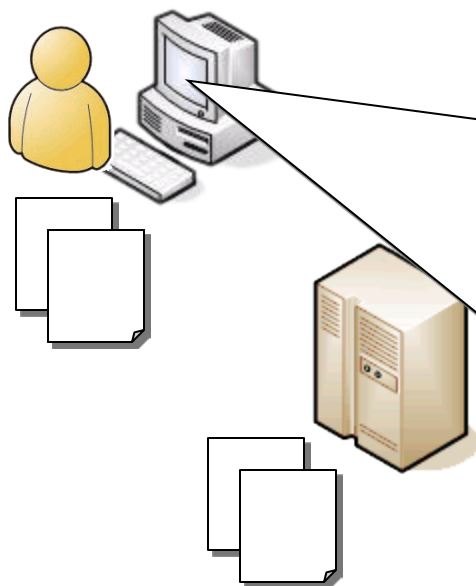
What are these secret codes? See [this page](#).





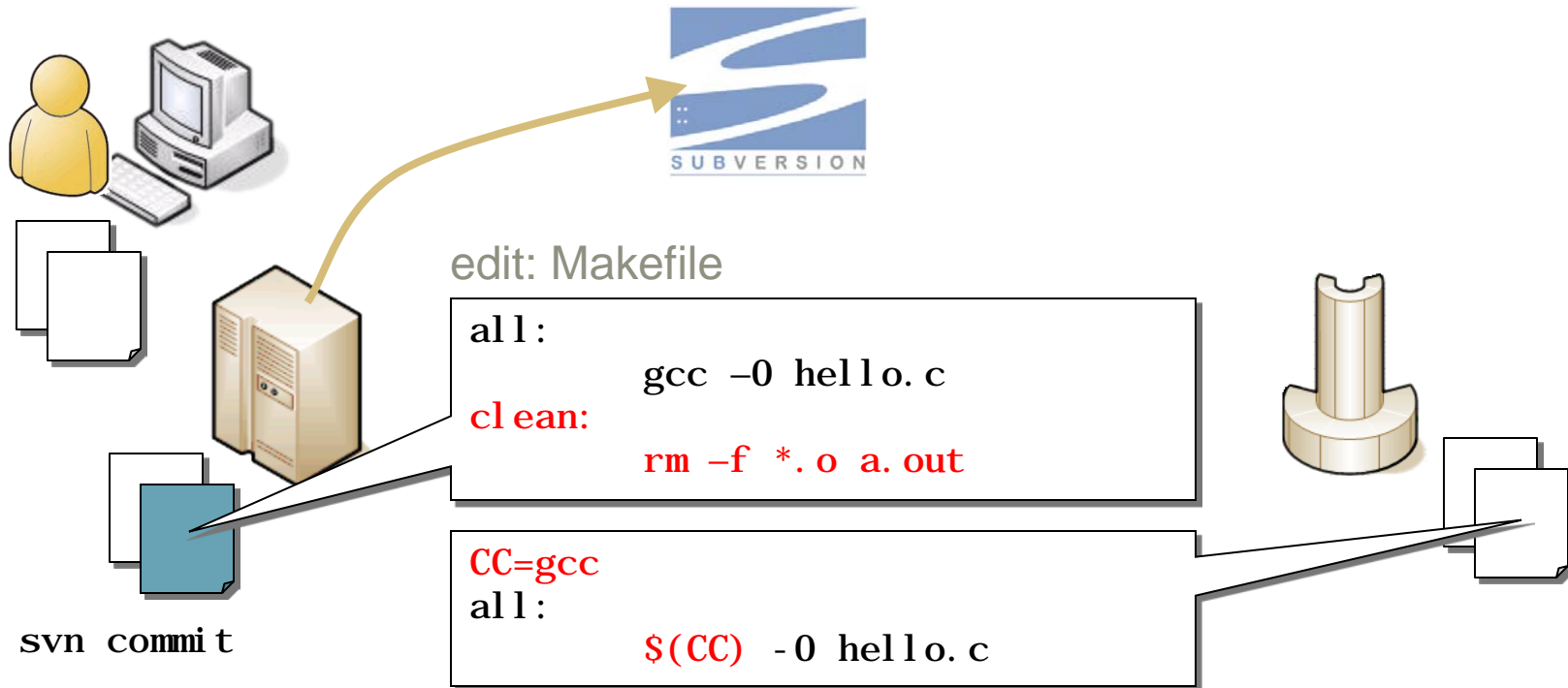
`svn checkout https://nanohub.org/tools/yourtool/svn/trunk yourtool`

- ~~• Copy code around~~
- Move code to new machines with "svn checkout"
- Move changes around with "svn commit" and "svn update"

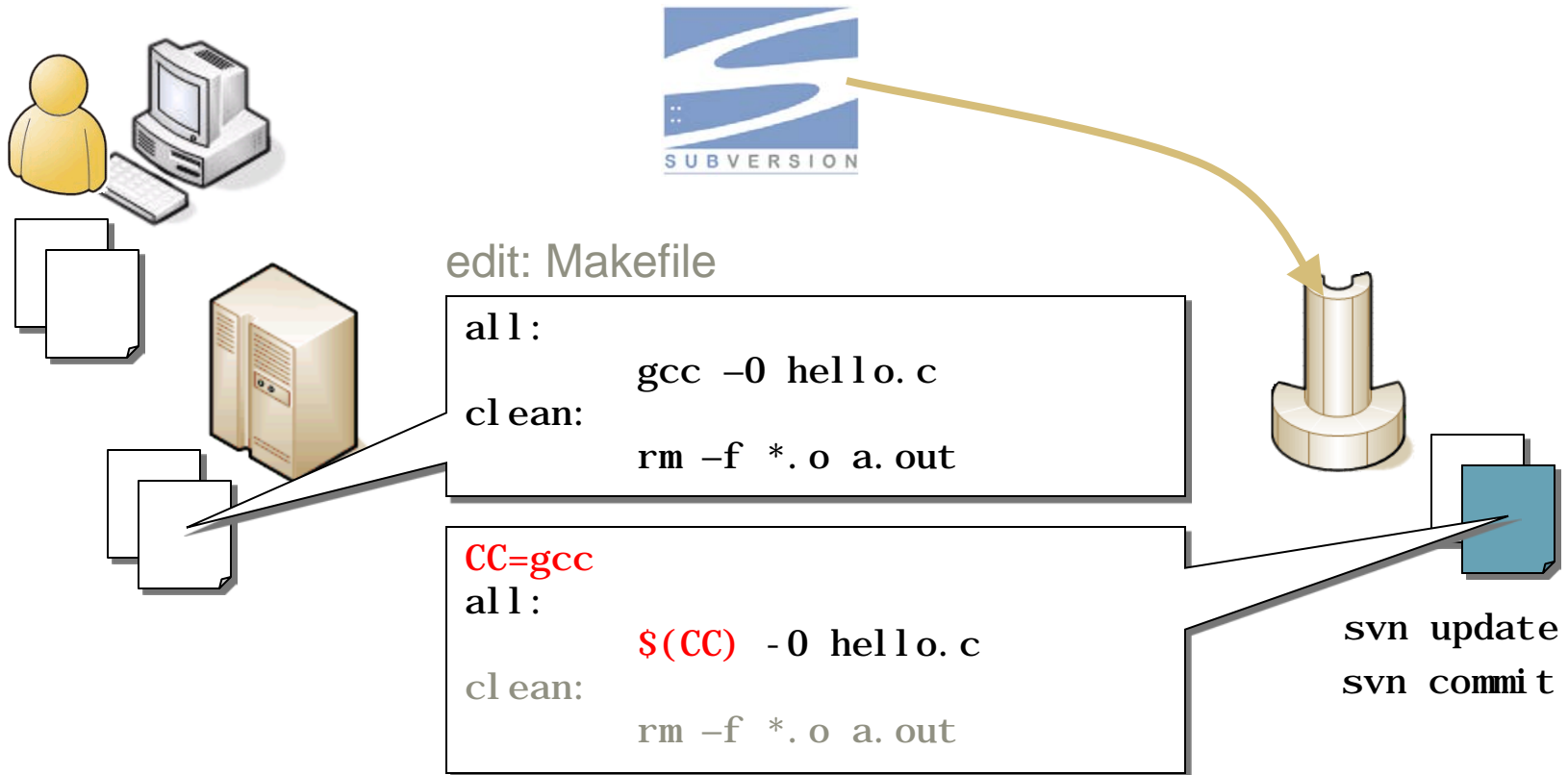


```
svn status
M      src/hello.c
svn diff src/hello.c
Index: src/hello.c
=====
-----
--- src/hello.c      (revision 4)
+++ src/hello.c      (working copy)
@@ -4,6 +4,7 @@
 int
 main(int argc, char **argv)
 {
-   printf("Hello, World!\n");
+   /* say hello to everyone */
+   printf("Hello, Universe!\n");
   exit(0);
}
svn revert hello.c
Reverted 'hello.c'
```

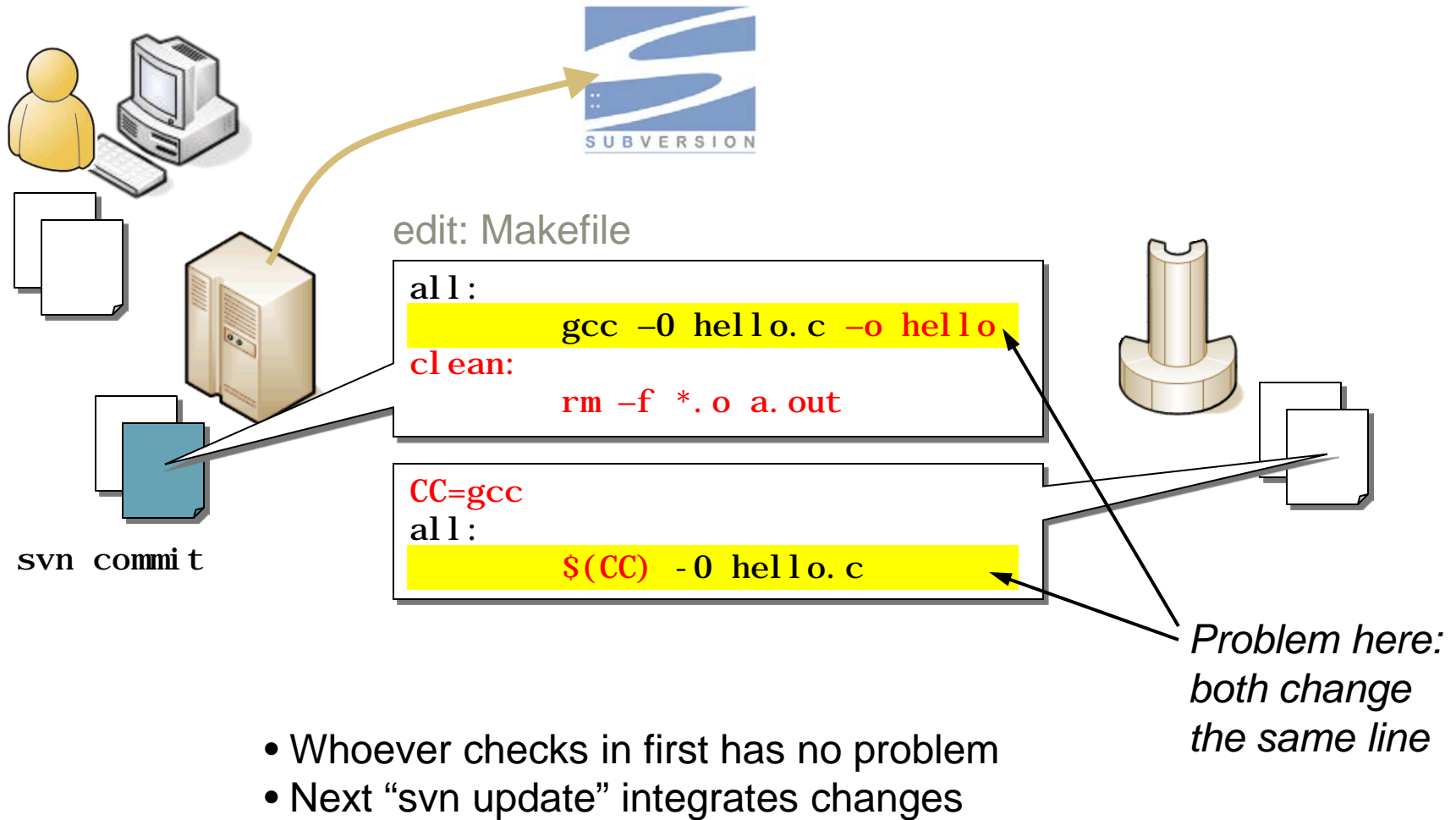
Can also revert  
directory changes  
(adding/deleting files)



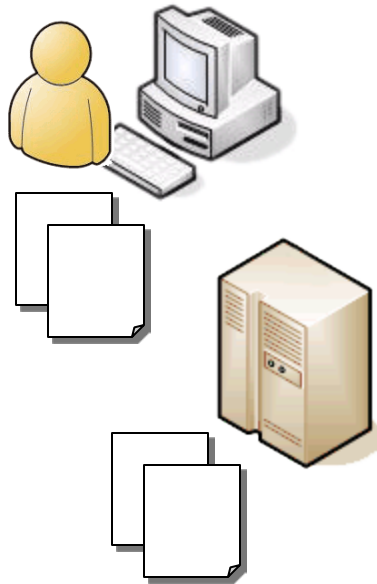
- Whoever checks in first has no problem
- Next “svn update” integrates compatible changes



- Whoever checks in first has no problem
- Next “svn update” integrates compatible changes
- Use “svn commit” to commit the merged changes







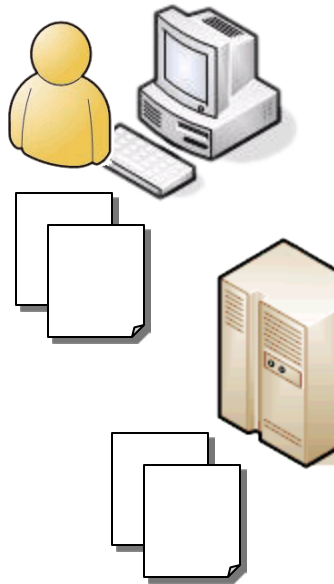
```
svn update
C  Makefile
Updated to revision 6.

svn commit
svn: Commit failed (details follow):
svn: Aborting commit: 'Makefile'
remains in conflict

svn status
?   Makefile.r5
?   Makefile.r6
?   Makefile.mine
C   Makefile
```

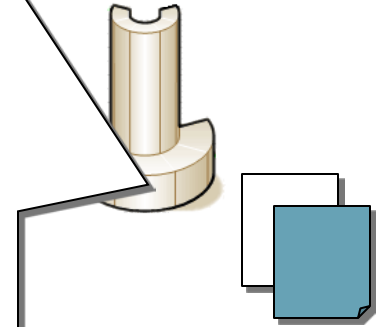
conflict!

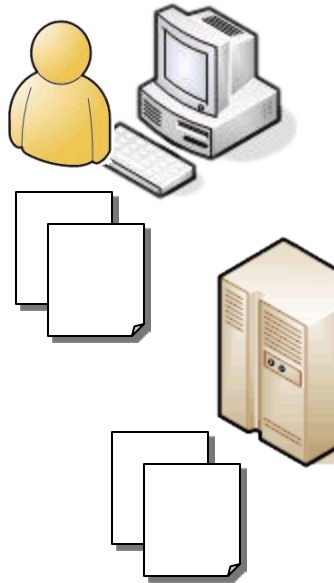
best guess at integrated changes



vi Makefile

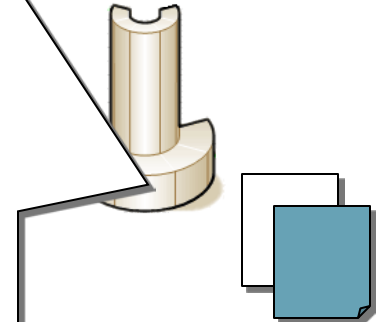
```
CC=gcc
all:
<<<<<< .mi ne
          $(CC) -O hello.c
=====
          gcc -O hello.c -o hello
clean:
          rm -f *.o a.out
>>>>>> .r6
```

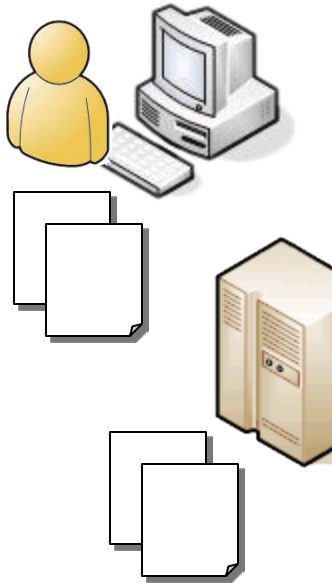




vi Makefile

```
CC=gcc
all:
<<<<<<< . mi ne
          $(CC) -O hello.c
=====
          gcc -O hello.c -o hello
clean:
          rm -f *.o a.out
>>>>>>> . r6
```





vi Makefile

```
CC=gcc
all:
    $(CC) -o hello.c -o hello
clean:
    rm -f *.o a.out
```

svn resolved Makefile

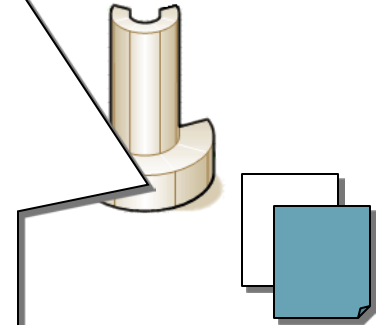
*Resolved conflicted state of 'Makefile'*

svn commit

*Sending src/Makefile*

*Transmitting file data .*

*Committed revision 7.*



- Get the whole distribution

```
svn checkout -r 3 https://nanohub.org/tools/yourtool/svn/trunk/yourtool
```

*get revision 3*

- Get a particular file

```
svn cat -r 5 Makefile _____ show me revision 5
```

```
svn cat -r 5 Makefile > Makefile _____ replace current file with revision 5
```

- Which revision?

```
svn log Makefile _____ show me revisions for this file
```

```
svn log _____ show me revisions for current directory
```

## Good defaults. Subversion usually does the right thing:

```
cp di agram. jpg examples
```

```
svn add examples/di agram. jpg
```

```
A (bin) examples/di agram. jpg
```

```
svn commit
```

```
Adding (bin) examples/di agram. jpg
```

```
Transmitting file data .
```

```
Committed revision 8.
```

Recognized as a binary file:

- no CR/LF translation
- no merges, only replacements

## When it fails, set properties yourself:

```
cp demo. dat examples
```

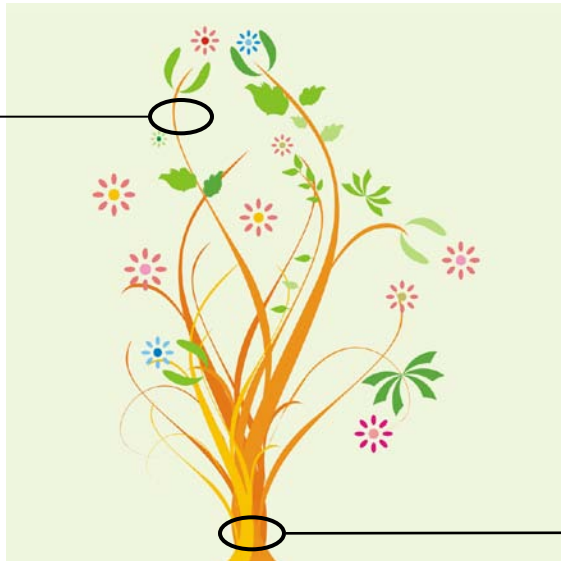
```
svn add examples/demo. dat
```

```
A examples/demo. dat
```

```
svn propset svn: mime- type application/ octet- stream examples/ demo. dat
```

```
svn propdel svn: eol- style examples/ demo. dat
```

more about properties on [this page](#)



Think of your source code repository as a tree...

Tag important versions:

```
svn copy trunk tags/release1.0
```

*Merging between branches is a pain!  
If you need that, use [svk](#).*

Create a branch:

```
svn checkout https://nanohub.org/.../svn no /trunk yourtool -all
cd yourtool -all
svn copy trunk branches/mclennan
svn commit -m "created private branch for mclennan"
```

```
svn checkout https://nanohub.org/.../svn/branches/mclennan yourtool -mcl
                                     instead of /trunk
```

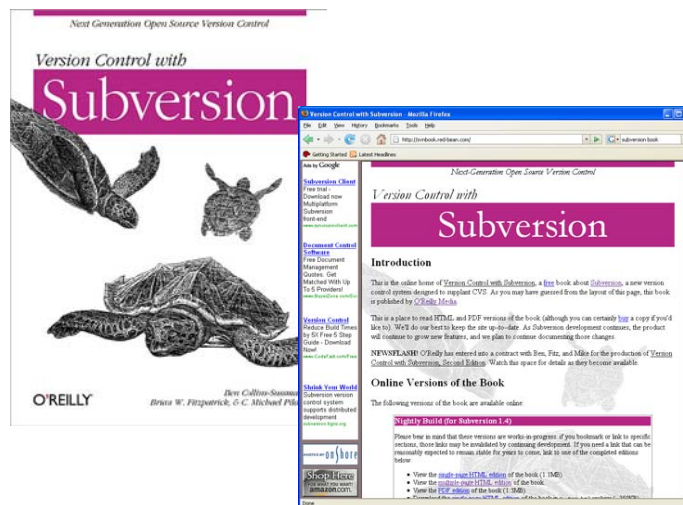
Web site: <http://subversion.tigris.org/>

## Subversion Book

From [O'Reilly & Associates](#), and also [online](#)

Quick-start guide for your project:

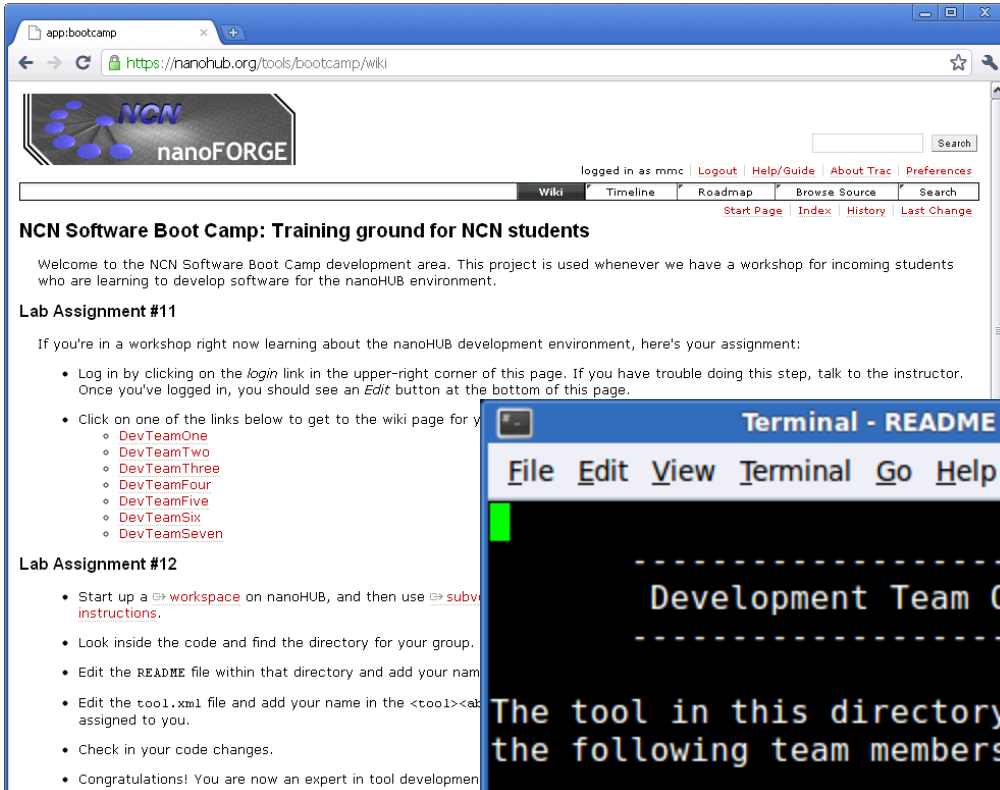
<https://nanohub.org/tools/yourtool/wiki/GettingStarted>



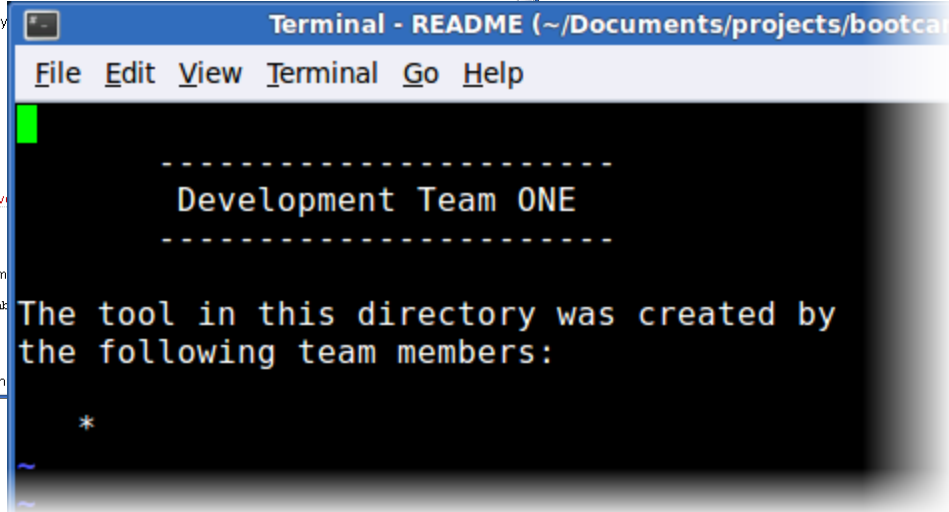


<https://nanohub.org/tools/bootcamp/wiki>

- Work in teams
- Use subversion to check out the source code
- Add your name to the README
- Check in your changes



The screenshot shows a web browser window with the URL <https://nanohub.org/tools/bootcamp/wiki>. The page features the NCN nanoFORGE logo and a navigation menu with options like 'Wiki', 'Timeline', 'Roadmap', 'Browse Source', and 'Search'. The main content area is titled 'NCN Software Boot Camp: Training ground for NCN students' and includes a welcome message and two lab assignments. Lab Assignment #11 provides instructions on logging in and navigating to team-specific wiki pages. Lab Assignment #12 details the steps for setting up a workspace, finding the team directory, and editing the README and tool1.xml files.



The terminal window shows the contents of a README file. The text is as follows:

```

-----
Development Team ONE
-----

The tool in this directory was created by
the following team members:

*
    
```