

# Using Subversion for Source Code Control



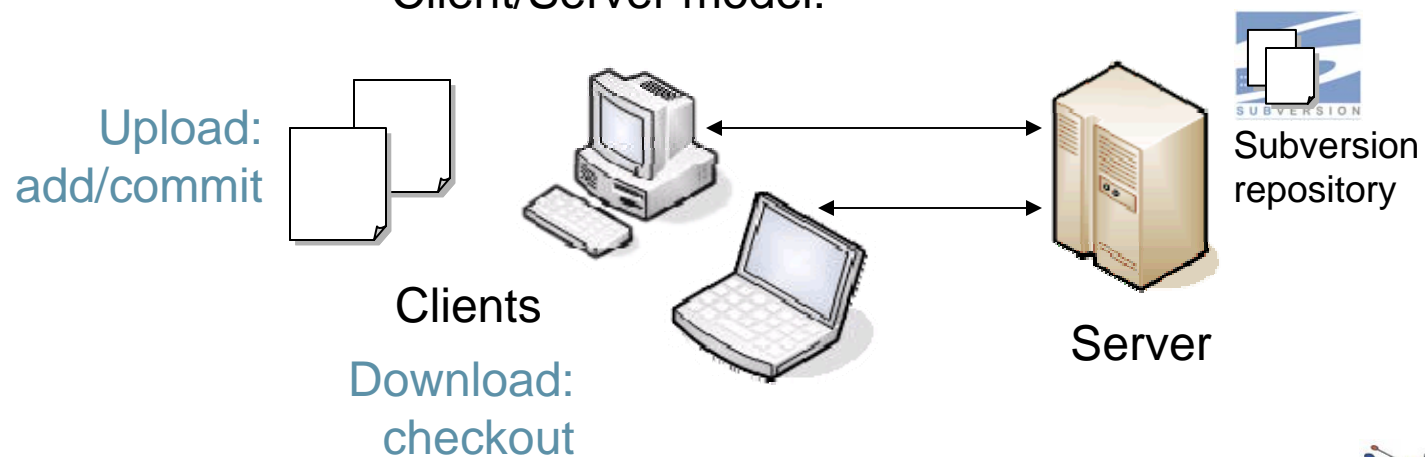
Michael McLennan  
Software Architect  
Network for Computational Nanotechnology

# What is Subversion?



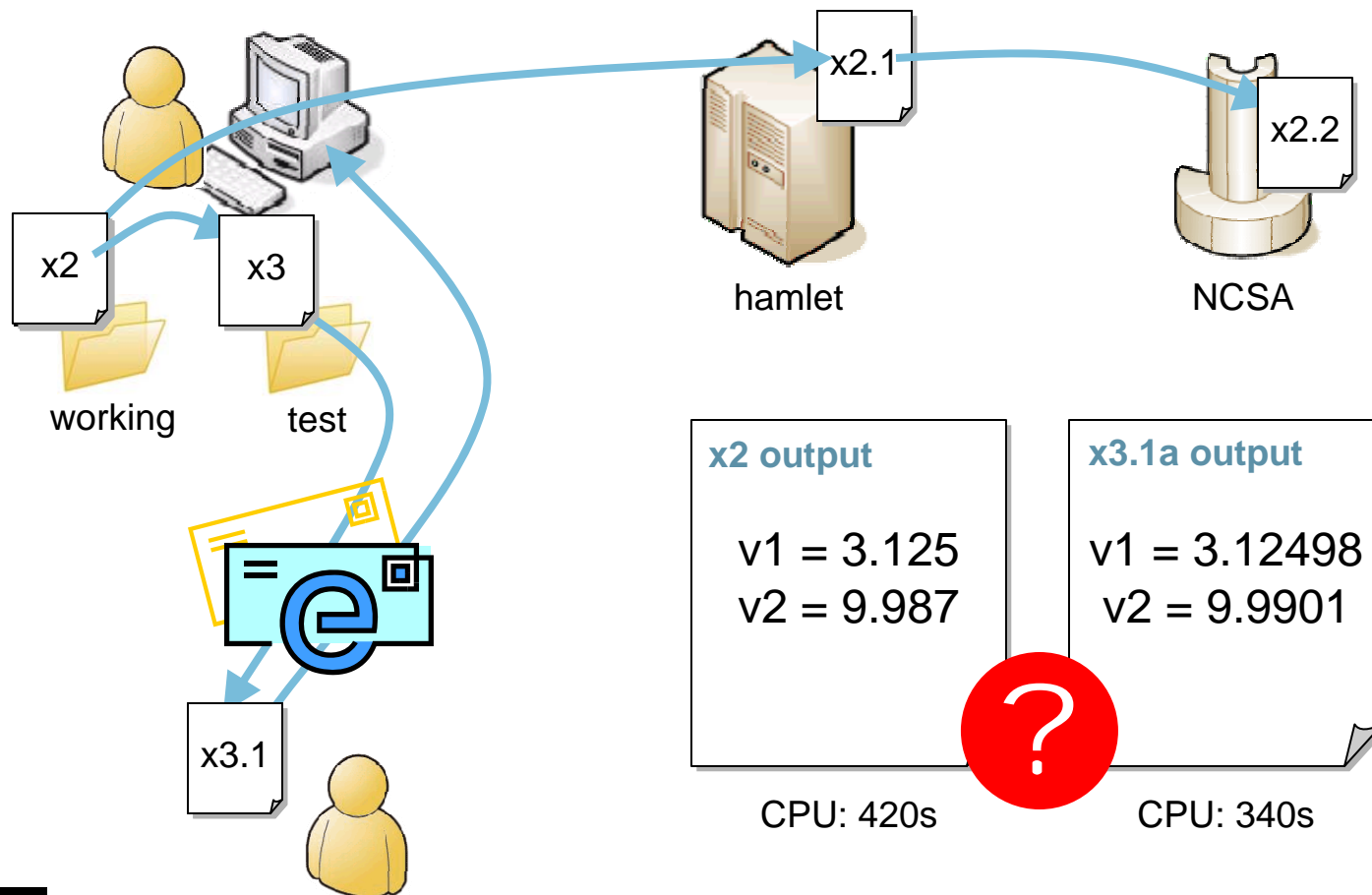
- CVS on steroids
- Created by developers at CollabNet
- In development since 2000
- In production (version 1.0) since Feb 2004
- Open source (Apache/BSD-style license)
- Unix/Linux, Win32, BeOS, OS/2, MacOS X
- Home page: <http://subversion.tigris.org/>

Client/Server model:



# Why bother with Subversion?

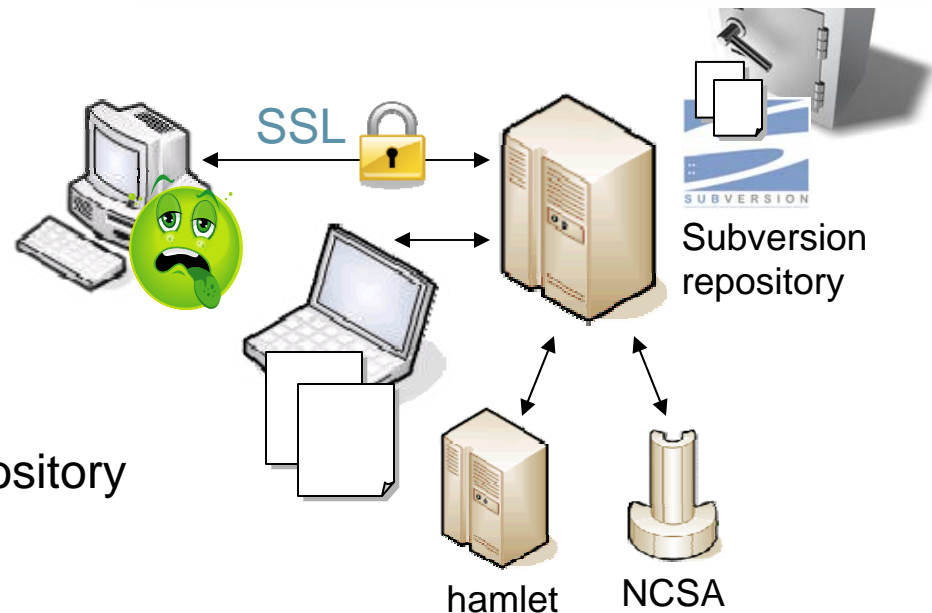
Does this sound familiar?



# Top 5 reasons why you should use Subversion

- 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 explaina
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"
```



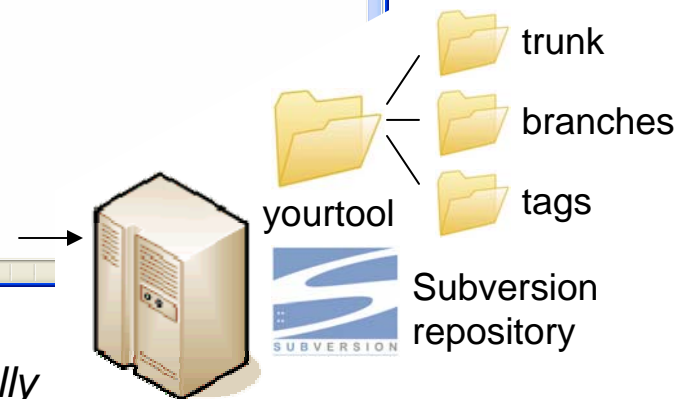
# Getting Started... the nanoHUB way

The image shows two browser windows from nanoHUB. The left window is at <https://www.nanohub.org/contribute/content/> and features a 'Contribute' section with a 'Start a contribution' button circled in red. The right window is at <https://www.nanohub.org/contribute/tools/register/> and displays a registration form with the following fields:

- Tool Name:**  (Short name, used for the directory containing this tool. Example: qdot)
- Title:**  (Full name for this tool. Example: Quantum Dot Lab)
- Version:**  (Optional version number for this release of the tool. Example: 1.0 or 2.1.5b)
- At-a-glance Description:**  (A one-line description of your tool. Example: Simulate 3-D confined states in simple quantum dot geometries.)
- Tool Access:**
- Source Code Access:**
- Development team:**  (nanoHUB logins for people allowed to modify your code. Example: mylogin, fred, barney, wilma)

Below the form is a 'Register Tool' button. To the right of the form, there is explanatory text: 'How do I contribute a simulation tool?' and 'What tool name should I choose?'.

Once you register your tool,  
your repository is created automatically



## Check out your code

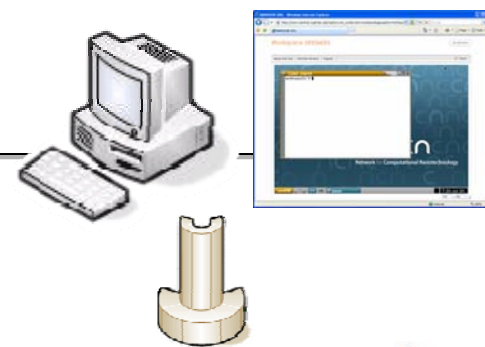
```
svn checkout https://repo.nanohub.org/svn/app-yourtool/trunk app-yourtool
A   app-yourtool /rappture
A   app-yourtool /doc
A   app-yourtool /src
A   app-yourtool /bin
A   app-yourtool /data
A   app-yourtool /middleware
A   app-yourtool /examples
Checked out revision 1.
```

```
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...



# Commit your changes

cd app-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:

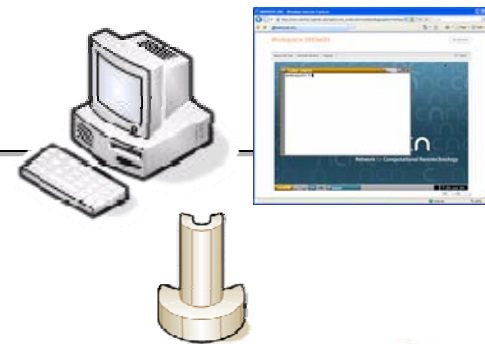
Created ex1 directory and added the README file.

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](#)



# What about Windows?

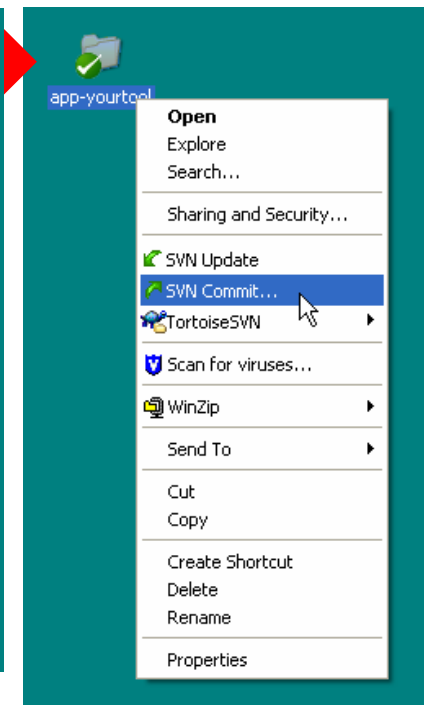
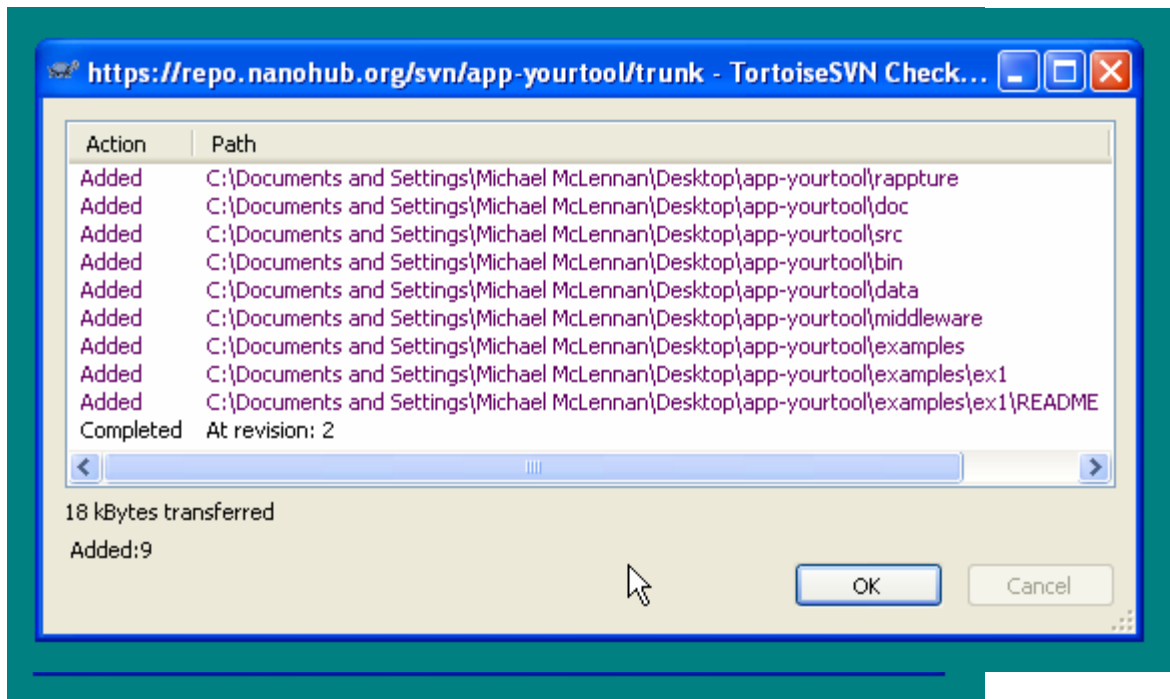


## TortoiseSVN

More info:

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

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



# Moving and removing files

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

brings up your favorite editor:

Moved some files around.

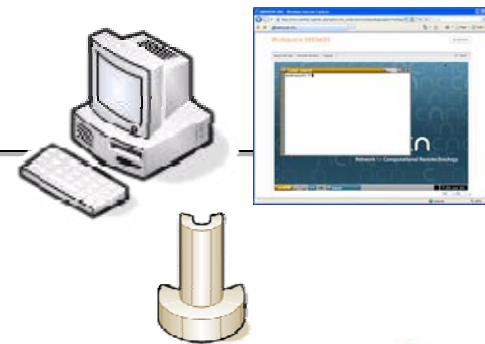
```
cd ../../
svn delete doc
D doc
```

Date	Rev	Chgset	Author	Log Message
07/26/07 09:46:35	3	3	mm	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

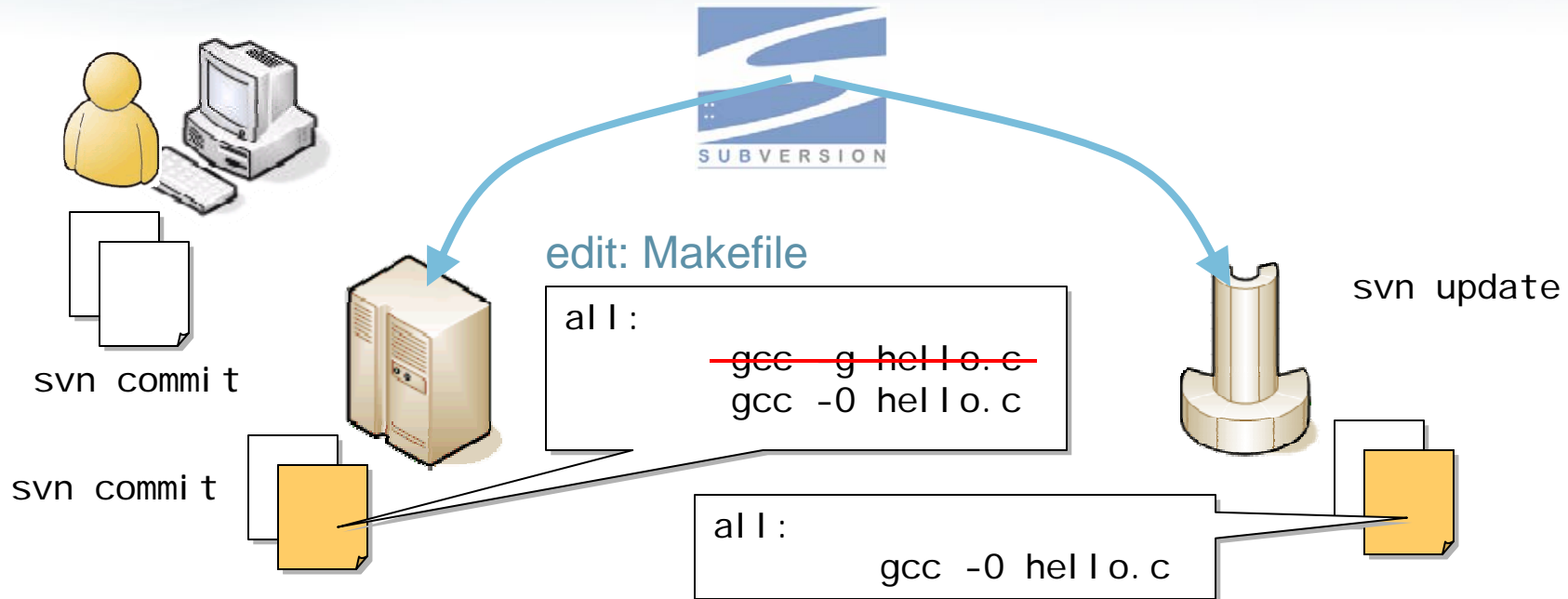
```
svn status
D doc
A + examples/ex1/README.txt
D examples/ex1/README
```

```
svn commit
```

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



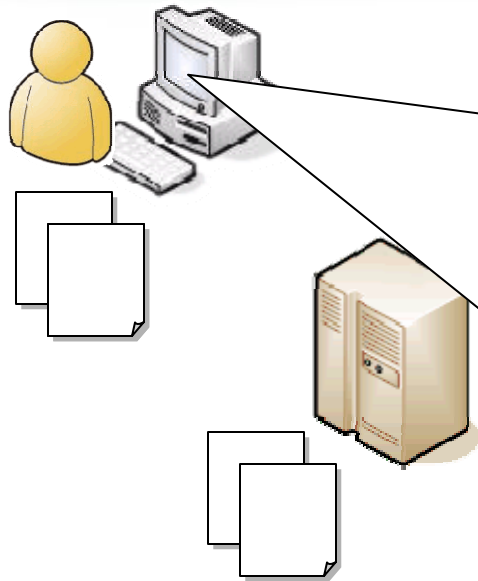
# Editing and updating



svn checkout <https://repo.nanohub.org/svn/app-yourtool/trunk> app-yourtool

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

# Looking for differences and reverting

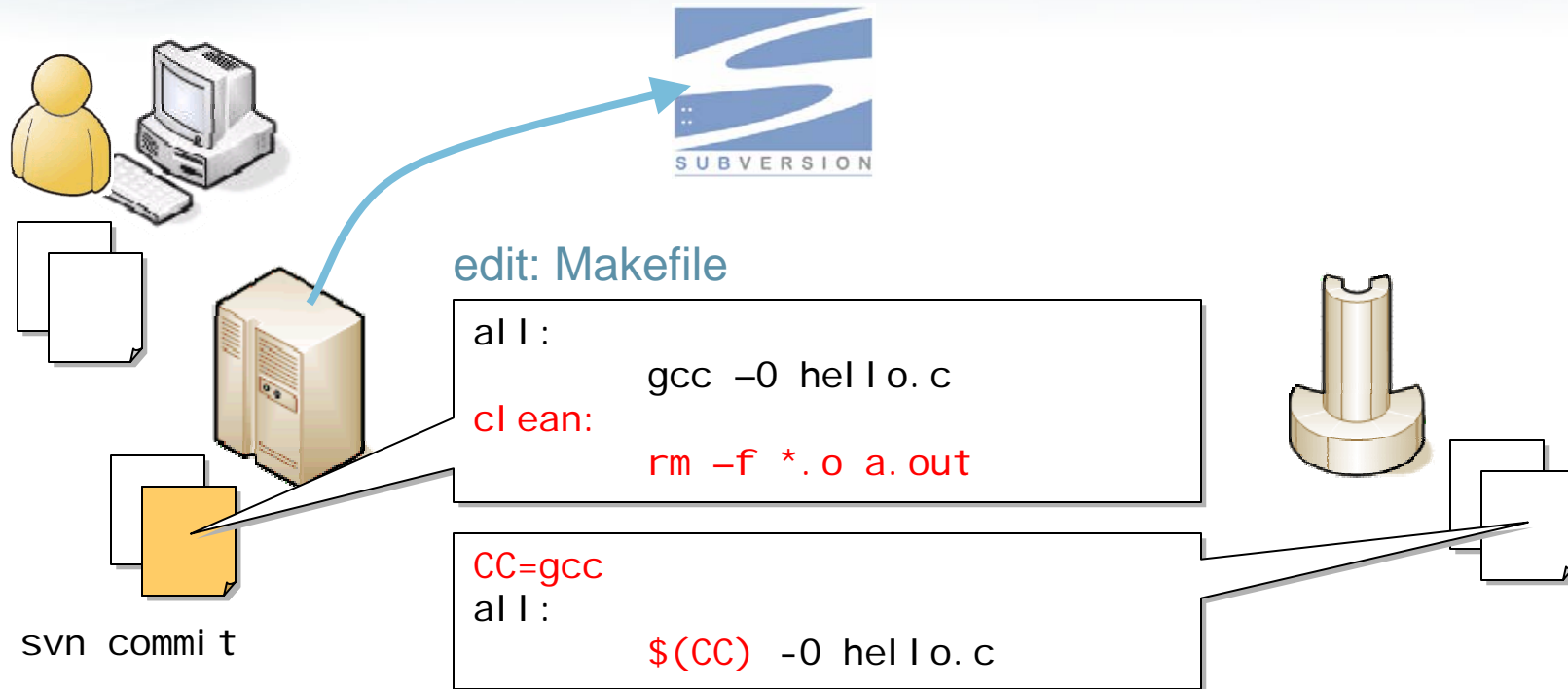


```

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'
    
```

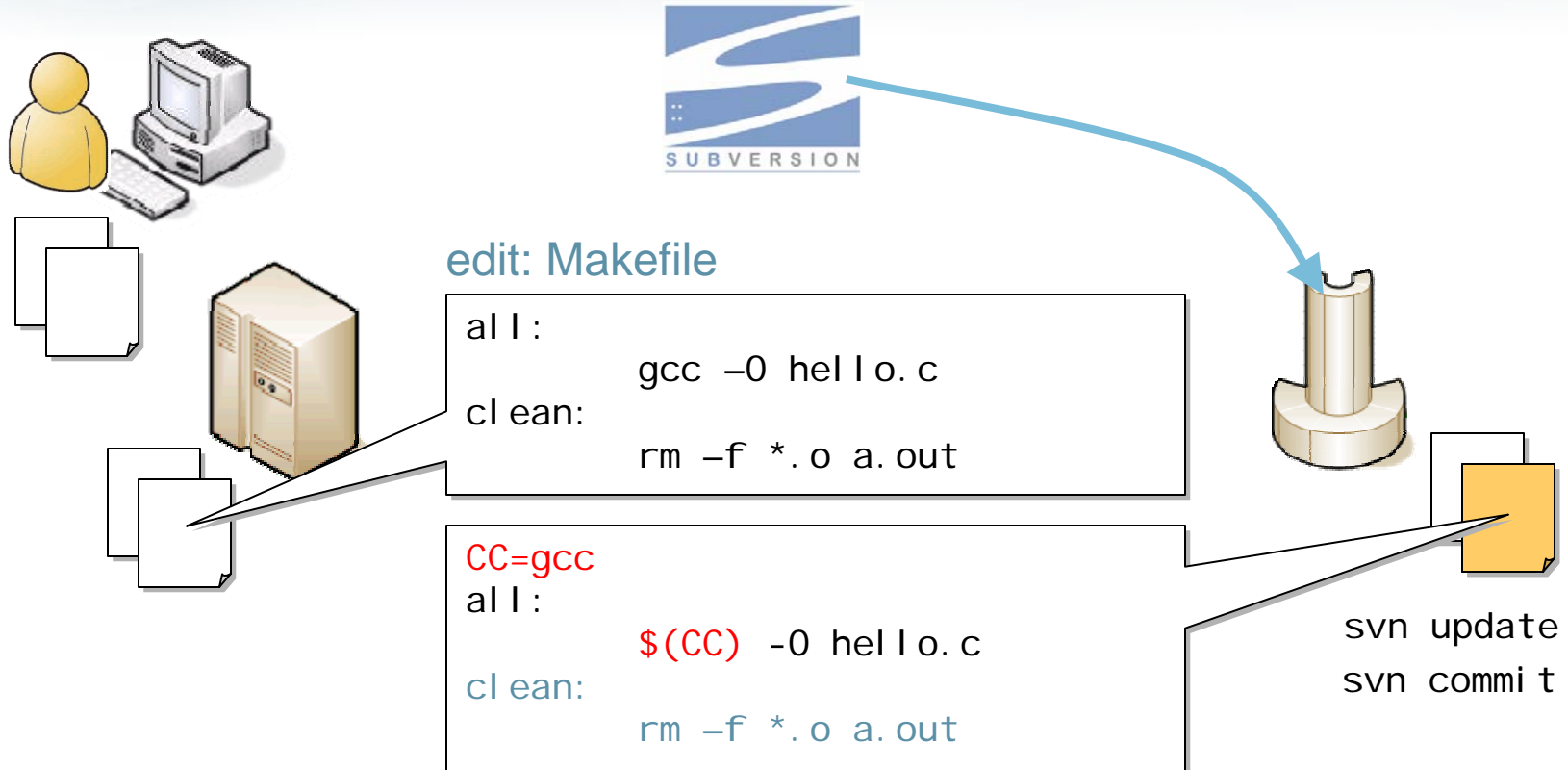
Can also revert  
directory changes  
(adding/deleting files)

# Merging changes



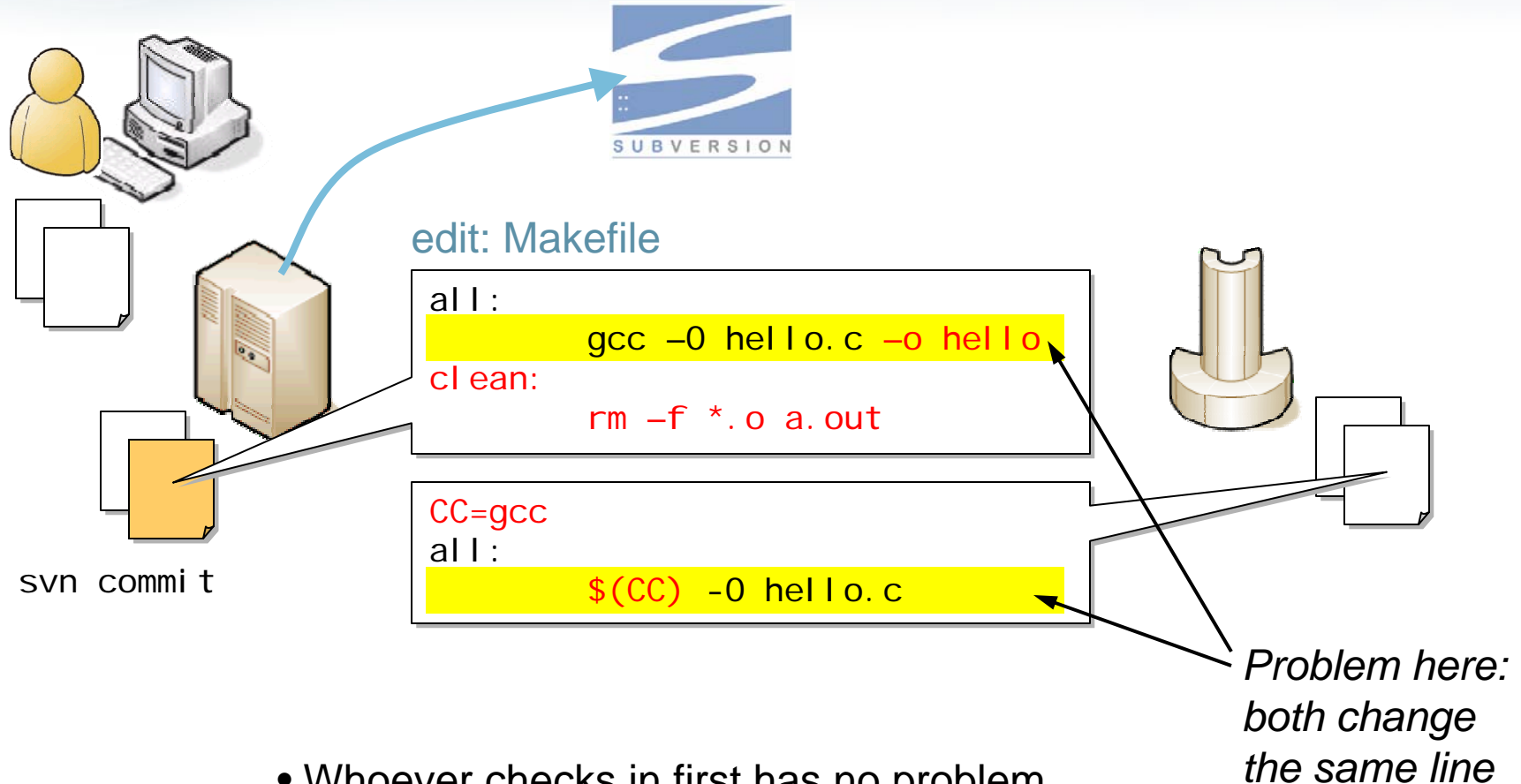
- Whoever checks in first has no problem
- Next “svn update” integrates compatible changes

# Merging changes



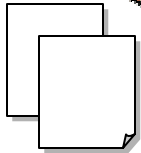
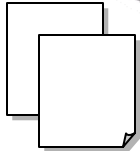
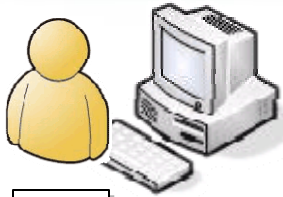
- Whoever checks in first has no problem
- Next “svn update” integrates compatible changes
- Use “svn commit” to commit the merged changes

# Resolving merge conflicts



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

# Resolving merge conflicts



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

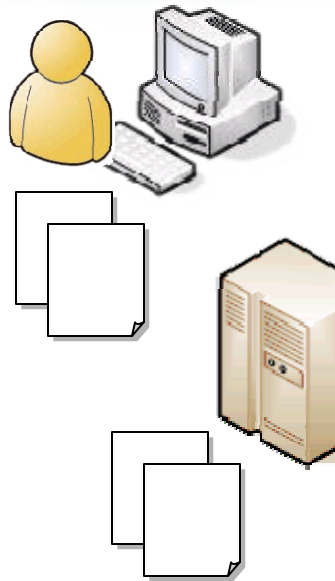
svn commit
svn: Commit failed (details follow):
svn: Aborting commit: 'Makefile'
remains in conflict

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

**conflict!**

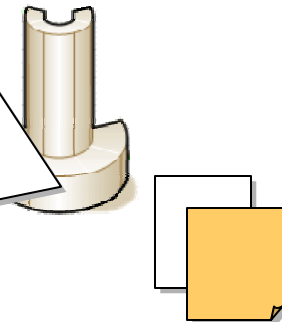
*best guess at integrated changes*

# Resolving merge conflicts

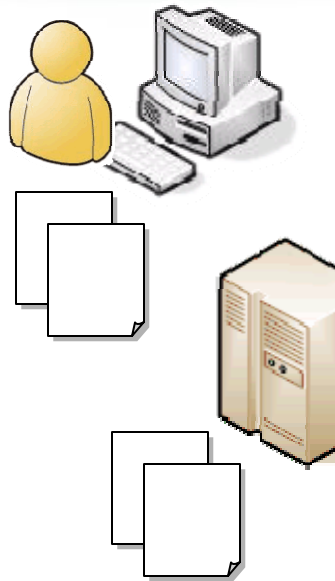


vi Makefile

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

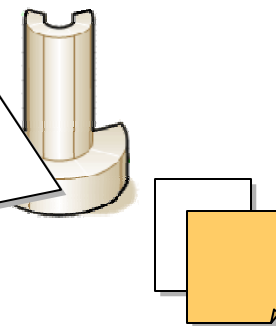


# Resolving merge conflicts

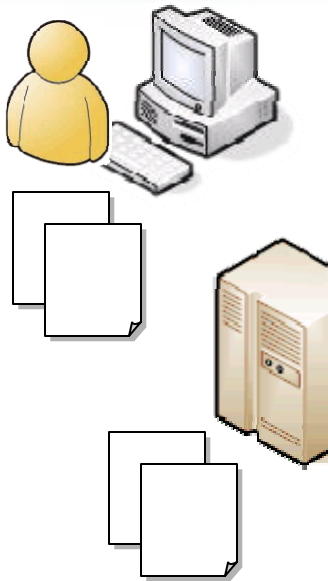


vi Makefile

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



# Resolving merge conflicts



vi Makefile

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

svn resolved Makefile

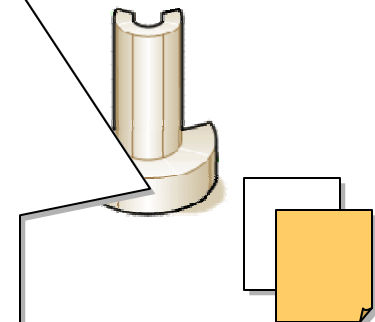
*Resolved conflicted state of 'Makefile'*

svn commit

*Sending src/Makefile*

*Transmitting file data .*

*Committed revision 7.*



## Retrieving an old version

- Get the whole distribution

svn checkout **-r 3** https://repo.nanohub.org/svn/app-yourtool/trunk app-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
```

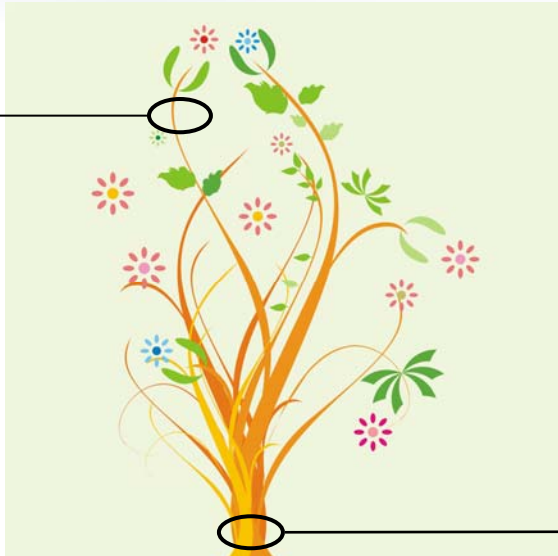
```
A examples/demo. dat
```

```
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](#)

# Branching and Tagging



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](#).*

trunk

Create a branch:

no /trunk  
(

```
svn checkout https://repo.../app-yourtool app-yourtool -all
```

```
cd app-yourtool -all
```

```
svn copy trunk branches/mcl ennan
```

```
svn commit -m "created private branch for mcl ennan"
```

```
svn checkout https://repo.../app-yourtool branches/mcl ennan app-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://developer.nanohub.org/projects/app-yourtool/wiki/GettingStarted>

