

# Version Control Using Subversion ver: 0.2

# *Subversion*

- ◆ This presentation is based on Linux server running svn
- ◆ Comments should be sent to me at the email below:
  - ◆ [anand@vsa-services.com](mailto:anand@vsa-services.com)

# *What is VC?*

- ◆ Ability to manage changes to files.
- ◆ Can we make changes, realize we made a mistake and undo the mistake?
- ◆ Can we find out who did what changes, especially when collaborating?

# *Subversion*

- ◆ Traditionally we use CVS
- ◆ Introducing SVN : Superior “CVS”
- ◆ Get it from:
  - ◆ Your Linux distro packages
  - ◆ <http://subversion.tigris.org>

Read documentation:

<http://svnbook.red-bean.com/nightly/en/index.html>

Browse archives and/or subscribe to the mailing list

[users@subversion.tigris.org](mailto:users@subversion.tigris.org)

and of-course, use Google.

# *CVS Problems*

- ◆ Does not handle directories versioning
- ◆ Differentiated binary and text files
- ◆ Non-atomic operations
- ◆ Not flexible (eg: per dir access control)
- ◆ Many irritants: eg: Replaced files carry the old history.
- ◆ Failed commits problematic – no atomicity

# *What is Subversion*

- ◆ Used for version control of files.
- ◆ Incremental changes, rollback edits
- ◆ Any type of file:
  - ◆ Plain text : source code, config files, xml files
  - ◆ Binary files: Images, audio, video, ROM images etc
- ◆ Svn is Free, Open Source
- ◆ Written by CollabNet in 2000, to be a part of their Enterprise product. Still funded by CollabNet
- ◆ Karl Fogel (Cyclic Software, CollabNet), Ben Collins-Sussman (CollabNet), Jim Blandy (Red-Hat)

# *Subversion – Time machine for files*

- ◆ Directory tree with all your files and revisions /info about changes – **Repository**
- ◆ Access locally, or across the network (eg. Another country)
- ◆ Client-Server architecture
- ◆ SVN is not SCM. It is a generic tool (a hammer). It can manage any fileset, not just C source code.
- ◆ Does not provide build tools etc.
- ◆ Remembers all changes ever done – to the repos.

# *Subversion-Users*

- ◆ KDE Project, Subversion itself
- ◆ Linux IEEE1394 dev project
- ◆ BioInformatics.org (eg:  
<http://bioinformatics.org/websvn/>)
- ◆ GCC project
- ◆ LANL
- ◆ HP, Sun, Samsung, Intel, Motorola, SK Tel
- ◆ BEA, Sybase
- ◆ Allianz, Barclays, First American, Dresdner Kleinwort
- ◆ Aventis, TCS, SAIC – many many more

# *Subversion-Goals*

- ◆ Fix CVS bugs, misfeatures
- ◆ As compatible as possible with cvs (atleast to the user)
- ◆ Efficient Branching, Tagging

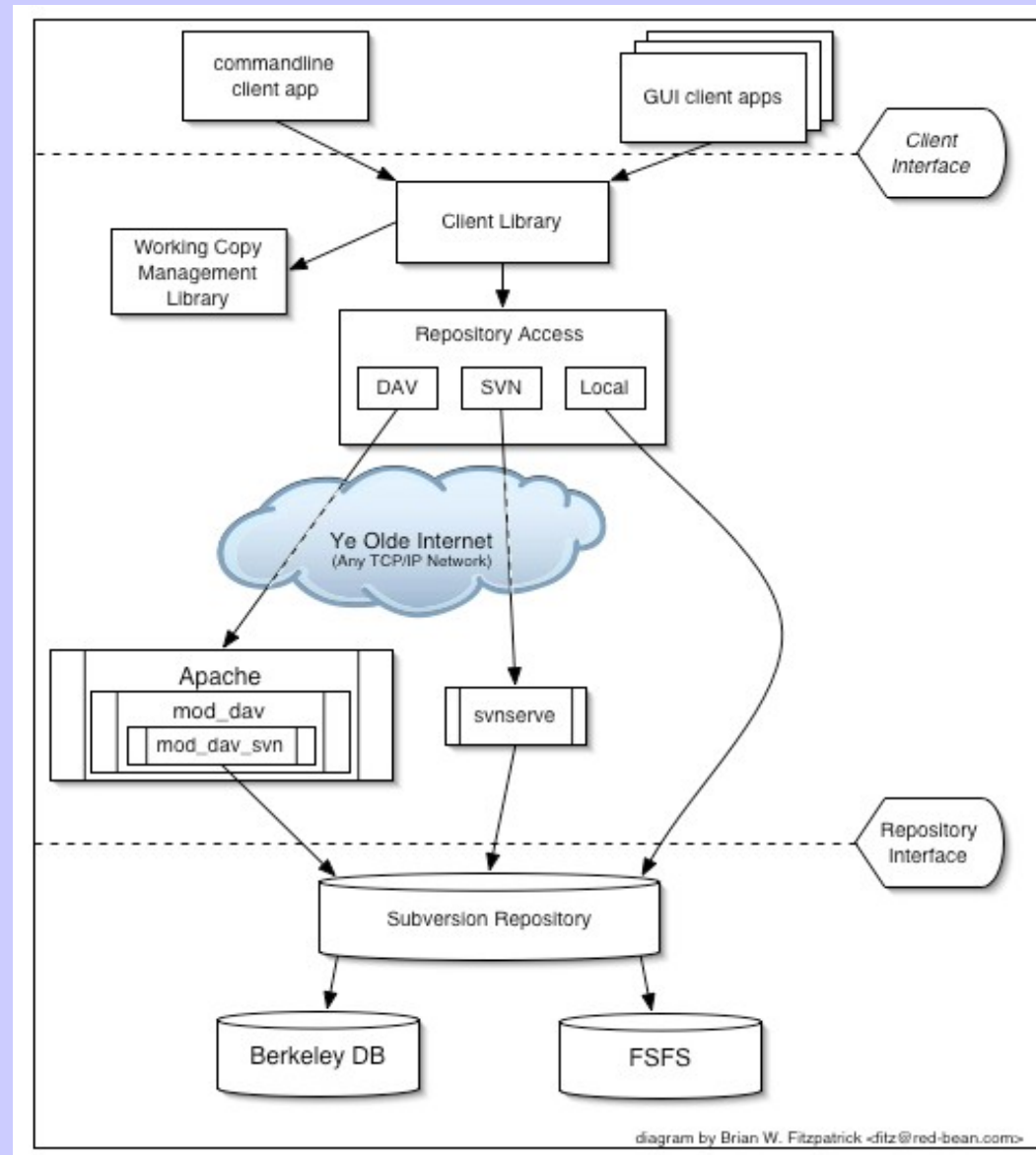
# *Subversion-Commercial Support*

- ◆ Many companies can provide svn support
- ◆ You can always go to CollabNet ([www.collabnet.com](http://www.collabnet.com))

# *Subversion-Key Features*

- ◆ Directory and file versioning (vs CVS' files only)
- ◆ Atomic Commits
- ◆ File & Dir Versioned Metadata : Any set of key/value pairs -- also versioned.
- ◆ Multiple network access layers: Native svn server, Apache, svn-thru-ssh
- ◆ Binary difference algo – works with text and bin data
- ◆ Efficient Branching and Tagging
- ◆ Implemented as c libraries
- ◆ well defined APIs – write software with your fav language (Perl, Python, C)

# Subversion-Architecture



# *Subversion*

- ◆ Runs on Linux, FreeBSD, Unix, OSX, Windows
- ◆ Uses Apache Runtime Lib ( file access, mem mgmt, network access)
- ◆ Not dependent on Apache, though
- ◆ Repos is a virtual directory – do not mess inside!
- ◆ Svn has no “conceptual understanding” of projects. You do.
- ◆ Svn know one thing – Directory tree. That's all.

# *Subversion-sample installation*

```
root@satdevel:~# apt-get install subversion
```

The following extra packages will be installed:

```
db4.2-util libapr0 libexpat1 libneon24 libpcre3 libsvn0 libxml2  
patch
```

The following NEW packages will be installed:

```
db4.2-util libapr0 libexpat1 libneon24 libpcre3 libsvn0 libxml2  
patch subversion
```

0 upgraded, 9 newly installed, 0 to remove and 2 not up-  
graded.

Need to get 3208kB of archives.

# *Subversion*

- ◆ Svn The command-line client program.
- ◆ Svnversion A program for reporting the state (in terms of revisions of the items present) of a working copy.
- ◆ Svnlook A tool for inspecting a Subversion repository.
- ◆ Svnadmin A tool for creating, tweaking or repairing a Subversion repository.
- ◆ Svndumpfilter A program for filtering Subversion repository dump streams.
- ◆ mod\_dav\_svn A plug-in module for the Apache HTTP Server, used to make your repository available to others over a network.
- ◆ Svnserve A custom standalone server program, runnable as a daemon process or invocable by SSH; another way to make your repository available to others over a network.

# *Subversion – client software*

- ◆ svn command line client -linux/unix/osx/cygwin
- ◆ esvn – Linux, Windows (based on Qt)
- ◆ kdesvn – KDE svn client
- ◆ rapidsvn – Linux (GTK)
- ◆ tortoiseSVN – Windows
- ◆ WebSVN – webbased PHP scripts.

# *Subversion -access methods*

- ◆ http://, https:// - use Apache
- ◆ svn://, svn+ssh:// - use svnserve
- ◆ file:/// - use filesystem

# *Subversion – create repos*

- ◆ Prepare the source code/files for upload:

```
mkdir idedriver
```

```
cd idedriver
```

```
mkdir branches tags trunk
```

```
cd trunk
```

```
cp -axv /usr/src/linux/drivers/ide .
```

```
cd ../..
```

# *Subversion – create repos*

```
# svnadmin create –fs-type fsfs /var/svn/idedriver  
# ls /var/svn/idedriver  
conf dav db format hooks locks README.txt  
#
```

for RHEL, CentOS:

```
#chown -R apache /var/svn/idedriver
```

for debian, ubuntu :

```
#chown -R www-data /var/svn/idedriver
```

tip: Create repos of type fsfs only and not BDB type

tip: Add a section to specify the access rights to the svn access file! (eg: I use /etc/svn-access – filename defined in apache config)

# *Subversion – add files – first time*

◆ prep input:

```
mkdir idedriver
```

```
cd idedriver
```

```
mkdir branches tags trunk
```

```
cp -axv /usr/src/linux/drivers/ide .
```

◆ Upload:

```
cd
```

```
svn import idedriver
```

```
file:///home/user01/proj1/idedriver -m
```

```
“first upload”
```

Wait for the upload to finish

# *Subversion – check...*

```
$ svn list file:///home/sysadmin/proj1/  
idedriver/
```

```
$ svn list
```

```
  file:///home/sysadmin/proj1/idedriver  
branches/  
tags/  
trunk/
```

◆ Upload OK. You might delete your input directory, if you wish

```
rm -rf /home/sysadmin/idedriver
```

# *Subversion - Checkout*

◆ Get a personal copy of idedriver to idedrv:  
svn co http://localhost/svn/idedriver/trunk idedrv

Note that the parameter idedrv is optional directory name.  
If not provided, repos is checked out to current directory.

The http:// url assumes you have setup svn access via apache.

# Subversion - Diff

```
cd idedrv/trunk/ide ; svn diff
```

- ◆ returns nothing since original and local copy are identical.

- ◆ edit ide-dma.c and change some lines. Try now:

```
$ svn diff
```

```
[sysadmin@vmcos4n00 ide]$ svn diff
```

```
Index: ide-dma.c
```

```
=====
```

```
--- ide-dma.c (revision 1)
```

```
+++ ide-dma.c (working copy)
```

```
@@ -89,7 +89,7 @@
```

```
#include <asm/io.h>
```

```
#include <asm/irq.h>
```

```
-
```

```
+/* WE ADD A COMMENT */
```

```
static const struct drive_list_entry drive_whitelist [] = {
```

```
    { "Micropolis 2112A" , "ALL" },
```

# *Subversion - revert*

- ◆ Ah, we made a mistake, just now, can we recover:

```
$ svn revert ide-dma.c
```

```
Reverted 'ide-dma.c'
```

```
$ svn diff
```

```
$
```

# *Subversion- commit*

- ◆ Now edit ide-dma.c again and let us commit the changes:

```
$vi ide-dma.c
```

```
$svn commit -m "add comments to ide-dma.c -  
anand"
```

```
Sending      ide/ide-dma.c
```

```
Transmitting file data .
```

```
Committed revision 2.
```

```
$
```

# *Subversion - update*

- ◆ If others have been working on the repos, and committing, you need to pull down those changes:

svn update

Note: svn responds with files listed with the following status:

A=add, R=replace, D=delete, G=merged successfully, C=conflict

# *Subversion – log, status*

◆ Check log entries:

```
$svn log
```

```
-----  
r1 | sysadmin | 2006-05-03 12:32:16 +0800 (Wed, 03 May 2006) | 1 line
```

```
first upload
```

```
-----  
$
```

```
svn log –revision {15:35}
```

```
svn status
```

tip: check status while you are inside your local repos tree.

# *Subversion -versions*

- ◆ HEAD : latest in the repos
- ◆ BASE: What is in your local repos (ignoring any local modifications)
- ◆ COMMITTED: Equal to BASE, or one rev before BASE where a file changed
- ◆ PREV: COMMITTED-1

# *Subversion - example*

```
svn diff -r PREV:COMMITTED ide-dma.c
```

```
svn diff -r HEAD
```

```
svn log -r BASE
```

```
svn log -r {2006-05-03}
```

tip: use tomorrow's date to get the latest else you will recv changes as of yesterday (2006-05-03 00:00)

```
svn co -r {13:45}
```

## *Subversion – add / delete files/dir*

- ◆ You can't just delete a file/dir in your local repo and commit. You have to also inform svn (or mark for add/del):

```
svn delete file.txt
```

- ◆ You need to tell svn of a new file/dir addition to your repos, eg:

```
cd $MYREPOS
```

```
cp /tmp/file.txt .
```

```
svn add file.txt
```

Note: dir adds are recursive

```
svn mkdir http://localhost/svn/idedriver/etc
```

# *Subversion – file copy, move, view, list*

- ◆ You can't just copy a file locally:

```
svn copy ide-lib.c newlib.c
```

```
A      idelib.c
```

```
svn move ide-lib.c idelib1.c
```

```
A      idelib1.c
```

```
D      ide-lib.c
```

```
svn cat http://localhost/svn/idedriver/Makefile
```

```
svn list http://localhost/svn/idedriver
```

# *Subversion -conflicts*

svn status -u

- ◆ will display files that can conflict
- ◆ All commits will fail until you resolve the conflict
- ◆ You can throwaway your changes:

svn revert file.txt

- ◆ Call the other dev, discuss and agree to use one of the versions:

svn resolved file.txt

svn ci -m "string....."

# *Subversion- recovering deleted files*

- ◆ If you know the file name/path, hunt down the rev when it was deleted, then:

```
cd repos
```

```
svn delete ide-pnp.c
```

```
svn ci -m "del file"
```

```
new rev=5
```

```
svn copy -r 4
```

```
http://localhost/svn/idedriver/trunk/ide/ide-pnp.c
```

```
ide-pnp.c
```

# *Subversion - WebSVN*

- ◆ Please watch the demo

# *Subversion-Branches*

- ◆ Create a branch when you need to start from the main repos, but work independently
- ◆ You need to share you mods with everyone without affecting the main repos.

eg: linux kernel has several branches: famous among them: -git -aa (andrea arcangeli), -mm (andrew morton), quilt tree (greg hartman), netdev network drivers (jeff garzik) etc

Note: Linux kernel uses git and not svn but the concept is same.

# *Subversion-Branches*

## ◆ Create a branch:

```
mkdir work
```

```
cd work
```

```
svn co http://localhost/svn/idedriver idedriver
```

```
cd idedriver
```

```
svn copy trunk/ide branches/ide
```

```
svn status
```

```
    A +  branches/ide
```

or:

```
svn copy http://localhost/svn/idedriver/trunk
```

```
    http://localhost/svn/idedriver/branches/branch2
```

# *Subversion-branches*

◆ Merge – actually diff-and-patch repos->your working copy

◆ use

```
svn merge -r rev1:rev2  
http://localhost/svn/trunk  
branches/branch2
```

# *The End*

- ◆ This presentation is based on the svn book, Version Control with Subversion, by Ben Collins Sussman, Brian W. Fitzpatrick, C. Michael Pilato and my experience with svn
- ◆ This presentation can be redistributed as follows:
  - No commercial re-distribution: eg, as part of a for-profit CDROM. Seek my permission first.
  - Must attribute the document creator.
  - Share alike: If you use this document and enhance it or modify, share the modifications or the modified document
  - Which means I apply: Creative Commons License, <http://creativecommons.org/licenses/by-nc-sa/2.5/>

# *The End*

- ◆ Thanks for your time. If you have any feedback, corrections or questions please contact me: Anand Vaidya, [anand@vsa-services.com](mailto:anand@vsa-services.com)
- ◆ This document was created with OpenOffice on Linux

