

# SUBVERSION

---

# Subversion

- What is source code version control?
  - <http://svnbook.red-bean.com/>
  - allow multiple people to modify the same source code
- ☑ • allow one person to manage multiple versions of their source code
  - move from computer to computer to develop
  - track all changes

# Repository



zeus.cs.pacificu.edu  
/home/punetid/SVNROOT/

Store your source code on zeus  
check it out and edit it on any  
other machine and upload your  
changes back to zeus.

# Client



moe.cs.pacificu.edu  
/home/punetid/workspace/HelloWorld

# Client

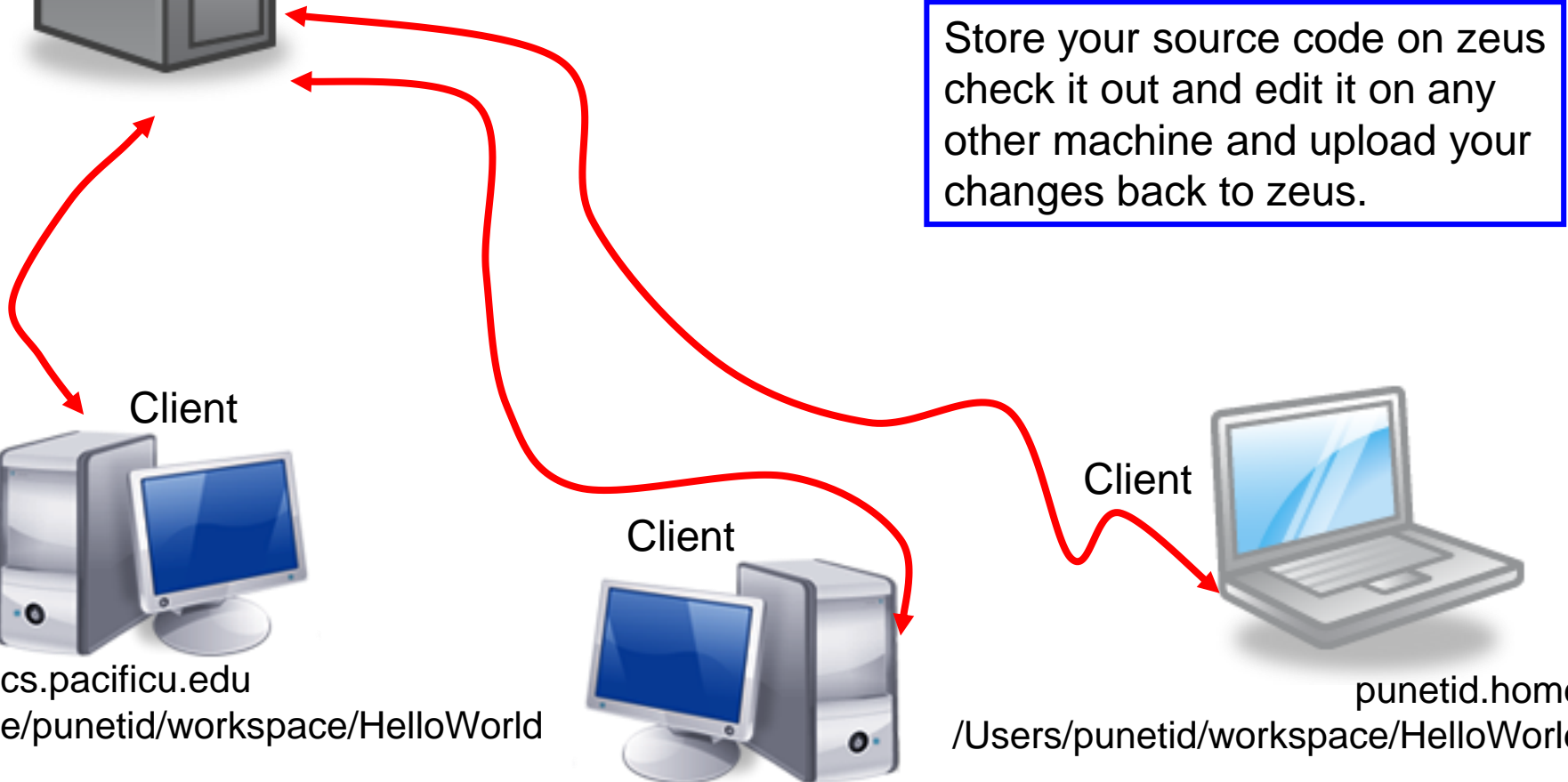


lisa.cs.pacificu.edu  
/home/punetid/workspace/HelloWorld

# Client



punetid.home  
/Users/punetid/workspace/HelloWorld



# Topics

- Subversion
  - Source Control
  - Check in
  - Check out
  - Update
  - Commit

# SVN

- Import the project **ArrayOfVoidStars** into Eclipse
  - Start Eclipse
  - Window->Preferences. Type SVN. Make sure that interface client is SVNKit (Pure Java)
  - File->Import->SVN->Projects from SVN
  - Select: Create a new repository location
  - Type in the following for the url:  
svn+ssh://zeus.cs.pacificu.edu/home/CS300Public/2017/SVNROOT\_CS300\_2017
  - Type in your zeus login and password
  - Click Browse then select **ArrayOfVoidStars**
  - Check out as a project with the name specified, then next, finish

# SVN

- Project has been imported into your workspace
- Right-click on project
  - Team->Disconnect
  - Make sure and delete meta-information
- Run the program and verify that it works
- Go to the file system and note that the project is in your workspace

# ArrayOfVoidStars

- Let's examine the code `arrayofvoidstars.c`
- How to use the debugger for a void \*
- Write the code asked for at the end of the program

# Your Own Repository

- Create a repository on zeus
  - do this exactly once!!!
  - use this one repository for all your projects

- ssh into zeus

- Type:

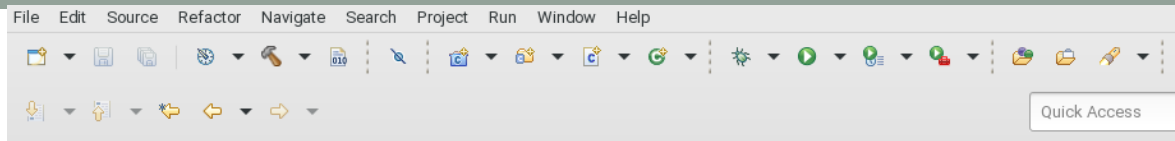
```
zeus$ svnadmin create /home/punetid/SVNCS300REPOS
```

- Replace punetid with your punetid which for me is  
**/home/ryand/SVNCS300REPOS**



# Check in ArrayOfVoidStars

- Problem: Let's check ArrayOfVoidStars into the repository
- How?
- Always clean project before checking in !!!!!
- Right click on the ArrayOfVoidStars project, then Team, then Share Project, then SVN, then Next, then Create a new repository location
  - See next slide



Project Explorer

- Airport [Airport]
- ArrayOfVoidStars
  - Binaries
  - arrays
  - Included Files
  - bin
  - included
  - src
    - arrays
  - Makefiles
- CS300y
- CS300y
- CS300Ex
- CS300Fu
- CS300Ge
- CS300Ge
- CS300Ge
- CS300Ge
- CS300Ra
- CS300Ra
- CS300Sk
- CS300St
- ExamStat
- ExamStat
- GenericS
- Included Files
- bin
- included
- infixe
- postfi
- src
  - sst
  - sst
- testdr
- Makefiles
- perf.d

arrayofvoidstars.c

```

1 /*****
2  File name:  arrayofvoidstars.c
3  Author:    Computer Science, Pacific University
4  Date:      9.11.16
              CS300
              Project: CS300 Lab
              Description: Show the use of an array of void stars including how to use
              the debugger with a void * variable. Finally, I have added
              a problem for you to code.
              *****/
<stdio.h>
<stdlib.h>

MAX_DATA 10

()

pData[MAX_DATA];

"PROGRAM START");

= 0; i < MAX_DATA; ++i)

i % 2 == 0)

data[i] = malloc (sizeof (char));
(char *) pData[i] = (char) ('A' + i);

data[i] = malloc (sizeof (int));
(int *) pData[i] = i;

= 0; i < MAX_DATA; ++i)

tf ("%5d", *(int *) pData[i]);

the code that sets the last void *
ge enough to hold your name. Then p
d star points to.
    
```

- New
- Go Into
- Open in New Window
- Copy Ctrl+C
- Paste Ctrl+V
- Delete
- Remove from Context
- Source
- Move...
- Rename... F2
- Import...
- Export...
- Build Project
- Clean Project
- Refresh F5
- Close Project
- Close Unrelated Projects
- Make Targets
- Index
- Build Configurations
- Show in Remote Systems view
- Profiling Tools
- Run As
- Debug As
- Profile As
- Restore from Local History...
- Run C/C++ Code Analysis
- Team
  - Apply Patch...
  - Share Project...
- Compare With

Share Project

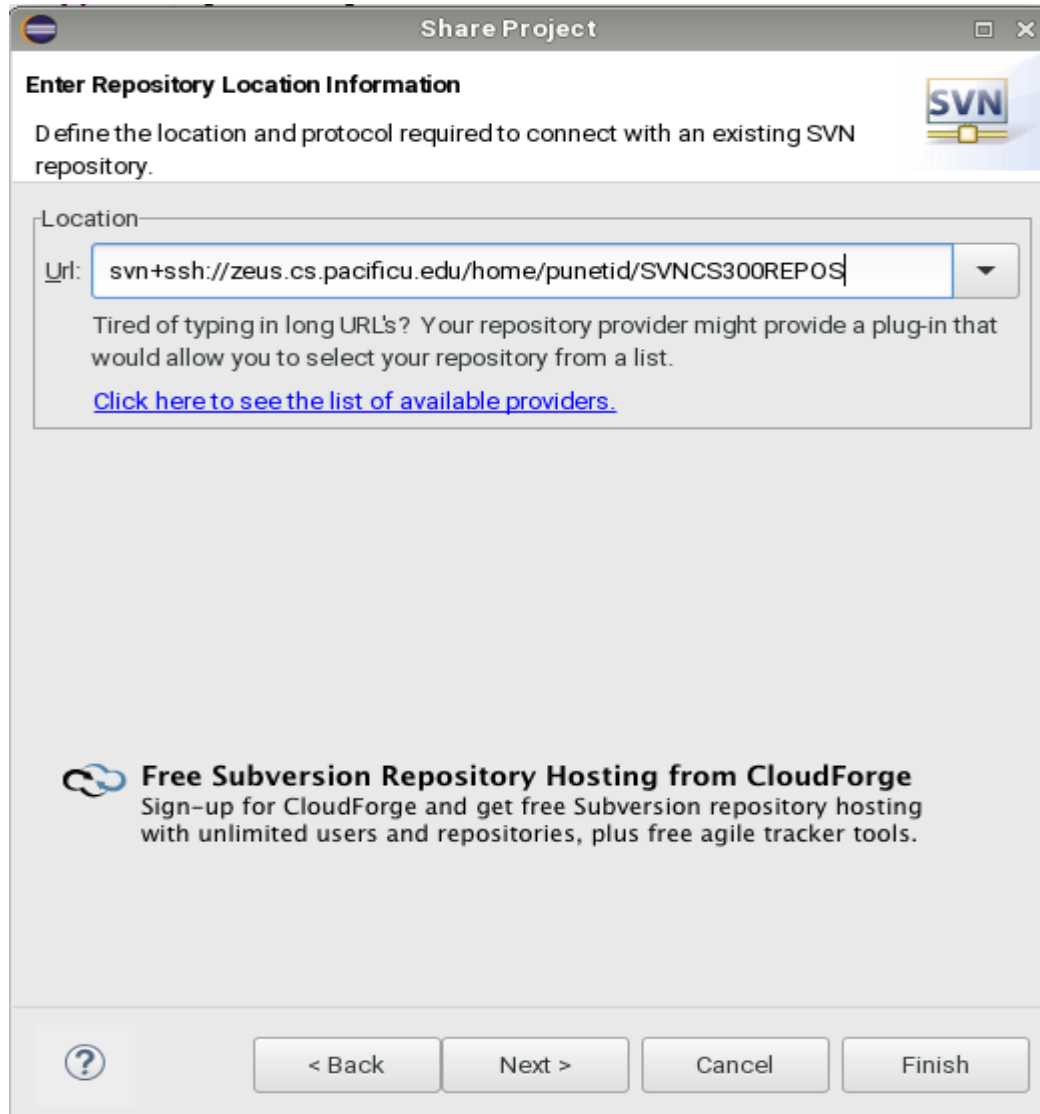
Select the repository plug-in that will be used to share the selected project.

Select a repository type:

- Git
- SVN

Select Create a new repository location

URL should use /home/punetid/SVNCS300REPOS on all subsequent slides



The screenshot shows a dialog box titled "Share Project" with a close button in the top right corner. The main heading is "Enter Repository Location Information" with an SVN logo to the right. Below the heading is the instruction: "Define the location and protocol required to connect with an existing SVN repository." A "Location" section contains a text input field labeled "Url:" with the text "svn+ssh://zeus.cs.pacificu.edu/home/punetid/SVNCS300REPOS" and a dropdown arrow. Below the input field is a note: "Tired of typing in long URL's? Your repository provider might provide a plug-in that would allow you to select your repository from a list." with a blue link: "[Click here to see the list of available providers.](#)". At the bottom of the dialog is a promotional banner for CloudForge: "Free Subversion Repository Hosting from CloudForge" with a CloudForge logo, followed by the text: "Sign-up for CloudForge and get free Subversion repository hosting with unlimited users and repositories, plus free agile tracker tools." The bottom of the dialog features a help icon (question mark in a circle) and four buttons: "< Back", "Next >", "Cancel", and "Finish".

**Share Project**

**Enter Folder Name**

Select the name of the folder in the SVN repository.

Use project name as folder name

Use specified folder name:

URL:

Click Finish and provide authentication information

# How to do a code commit

To commit a project, right click on the project folder →  
Team → Commit

Do not commit (i.e. uncheck) any binary files!!!!!! Always  
clean before committing.

Add very descriptive comments for EACH code commit.  
You will not be sorry.

Do not leave your computer for any reason before  
committing

# Version Control

- Each change you make to the source code is a **revision** stored in the repository
  - can annotate your change with a note
    - why did I do that?
  - you can browse back through the repository to find old revisions of file
    - changed a data structure and it did not work
    - rewrote an algorithm and it got slower!
  - check out the old (working) revision from the repository

# Hmmm....

- How often should I *update* and *commit*?
  - every major change
  - once every 15 minutes
  - right before you do something you think may be a bad idea
  - be sure to update and commit before you log off of a lab machine!
    - Or before you leave the lab
    - Someone may reboot your machine!

# Make a change in Eclipse

- Do multiple commits before today is over
- Build and run (just to be sure)
- Commit to SVN:
  - Right Click SVNTest | Team | Commit
- Do NOT commit .o or executable files!

## Show History

- In Eclipse
- Right Click a File
  - Team | Show History