



Working with the SWIM Code Repository

David E. Bernholdt

Oak Ridge National Laboratory

bernholdtde@ornl.gov

<http://www.csm.ornl.gov/~bernhold>



Why Use Centralized Version Control?

- A centralized repository makes it possible for any (authorized) person to obtain the code
 - No waiting on someone to send it to you
 - No question of which is canonical version
- Systematic version control gives you a complete history of changes to the code
- Centralized version control allows joint development without forking code



Subversion Version Control System

- Subversion (SVN) is the successor to CVS
- Extends CVS functionality modestly
- Similar command set to CVS
 - `svn checkout repositoryURL [path]`
 - `svn status, svn update`
 - `svn commit --message message [files]`
 - `svn revert file`
 - `svn add, svn delete, svn mkdir, svn move`
 - `svn import`
- Commits are atomic
- Version number increments with each commit
- See cswim.org → [Guides](#) → [Subversion Howto](#) for more info and pointers



Subversion Supports Multiple “Repositories”

- SWIM philosophy is to use separate repositories for separate items
 - `ips` – integrated plasma simulator
 - `doc` – project documents (papers, talks, posters, reports, etc.)
 - `LinSysAnalyzer` – Randy’s Linear System Analyzer
- See cswim.org → *Guides* → *SWIM Code Repositories* for list of repositories and locations
- Some codes already have repositories on other hosts
 - Moving to cswim.org is an option, but not a requirement
 - We may eventually mirror them on cswim.org



SVN Terminology and Conventions

- The top level of any SVN repository contains three subdirectories
- **Trunk (e.g. `ips/trunk/`)**
 - Tree where primary development takes place
- **Tags (e.g. `ips/tags/`)**
 - Specific (fixed) versions, marked for separate checkout
 - For example, mark public releases of code, or versions known to work correctly
- **Branches (e.g. `ips/branches/`)**
 - Places where private development can take place
 - Merging of changes between trunk and branch are controlled by user



A Tour of the IPS Repository

```
svn checkout https://cswim.org/svn/cswim/ips/trunk IPSDev
```

IPSDev/

```
STATE_batchelor/  
  mod/  
  src/  
  obj/  
doc/  
  2006/  
    design/  
      Example_PS_and_PS_rf/  
    papers/  
    presentations/  
    reports/  
    workshops/  
    DOE/  
  framework/  
  components/  
src/  
utilities/  
  src/
```

IPSDev/components/

```
state/  
  batchelor/  
    src/  
  mccune/  
    mod/  
    src/  
    obj/  
  rf/  
  toric/  
  aorsa/  
    legacy/  
    src/  
  shared/  
    src/  
framework/  
  utils/  
  lsa_scripts/
```



Key Points of IPS Repository Structure (1)

- Organize components by class, then instance (specific code they wrap)
- Documentation of components should parallel actual components
- It is better to capture is *somewhere* rather than *nowhere!*
 - If you're unsure where something belongs, ask Randy or David
 - Things can be moved within an SVN repository
 - Polite to announce the move, in case others are working on the same part of the tree



Key Points of Repository Structure (2)

- Directories shown are where source code (for component wrappers) & documentation should be developed and built
- Most actual physics codes will live outside the IPS tree
 - Come from other version control repositories
 - For Fast MHD campaign should not need to be modified

```
MyIPS/  
  IPSDev/  
    aorsa  
    cq13d/  
    m3d/  
    ...
```



IPS "Installation" Directories

- IPS tree also encompasses directories that hold the scripts/binaries/libraries as they are installed for use

IPSDev/

bin/

etc/

lib/

libexec/

man/

share/

- These directories should **NEVER** have anything check into them directly (in SVN), only installed from the place where they are built!



IPS Development Workflow

- Develop components in IPS tree
- Modify physics codes in their own trees
- Install all codes into `IPSDev/bin`, ...
- Put `IPSDev/bin` in path (or copy) to run
- See cswim.org → *Guides* → *IPS Development Workflow* for more detailed explanation



Integrating a Large System is an Evolutionary Process

- These guidelines draw on experience developing large, integrated, simulation software
- Organizing large codes of this type is a “research” question, not a textbook question
- Flexibility to change and adjust as we gain experience
- If you don’t understand something (or the motivation for it) just ask
- If you don’t think something is “working” and you have a better approach – speak up!



Guides on SWIM Web Site

- Subversion Howto
- SWIM Code Repositories
- IPS Development Workflow
- *All can be found at cswim.org → Guides*
- Please suggest improvements
 - Or make them yourself!