Introduction to NOνASoft and SVN

Jan Zirnstein

University of Minnesota

Software Tutorials, April 02, 2014

Where to begin

- Before you follow this guide:
  - Get FNAL computing access priviledges
  - `https://cdcvs.fnal.gov/redmine/projects/novaart/wiki/Fermilab_Computing_Access`
  - This will get you permission to access the NOνA General Purpose Virtual Machines (novagpvmxx.fnal.gov)
- What this guide covers:
  - How to connect to the NOνAGPVMs
  - How our software is organized
  - How to set up a frozen release and run a simple job
  - How to set up a test release and check out a package
  - How to build the packages in your test release and troubleshoot errors

Where to get more Information

- This tutorial will paraphrase and elaborate on `https://cdcvs.fnal.gov/redmine/projects/novaart/wiki/Documentation_FOR_BEGINNERS`
- It's a wiki, if you find things that are incorrect or unclear, please update the wiki

How to Connect - Part I

- Set up your local environment (`~/.ssh/config`) once

```
Host fermi
    User janzirn
    HostName nova-offline.fnal.gov
    ForwardAgent yes
    ForwardX11 yes
    ForwardX11Trusted yes
    Compression yes
    ServerAliveInterval 60
    GSSAPIAuthentication yes
    GSSAPIDelegateCredentials yes
    StrictHostKeyChecking no
    UserKnownHostsFile=/dev/null
    PasswordAuthentication no
```

- Remember to change the `User`

How to Connect - Part II

- Get a kerberos ticket
  `kinit -f -l 7d janzirn@FNAL.GOV`
- Again, you're not me so change the principal to your own
- Time to log into a NOνAGPVM
  `ssh fermi`
- Presto!
- There are 10 GPVMs dedicated to NOνA and the above `HostName` works as a round robin connection broker

What's in a GPVM

- Four cores, code development and quick test jobs only
- Running 64 bit Scientific Linux Fermi (SLF) 5
- Make yourself a directory in `/nova/app/users` for code development
- No need to run VNC sessions to these machines

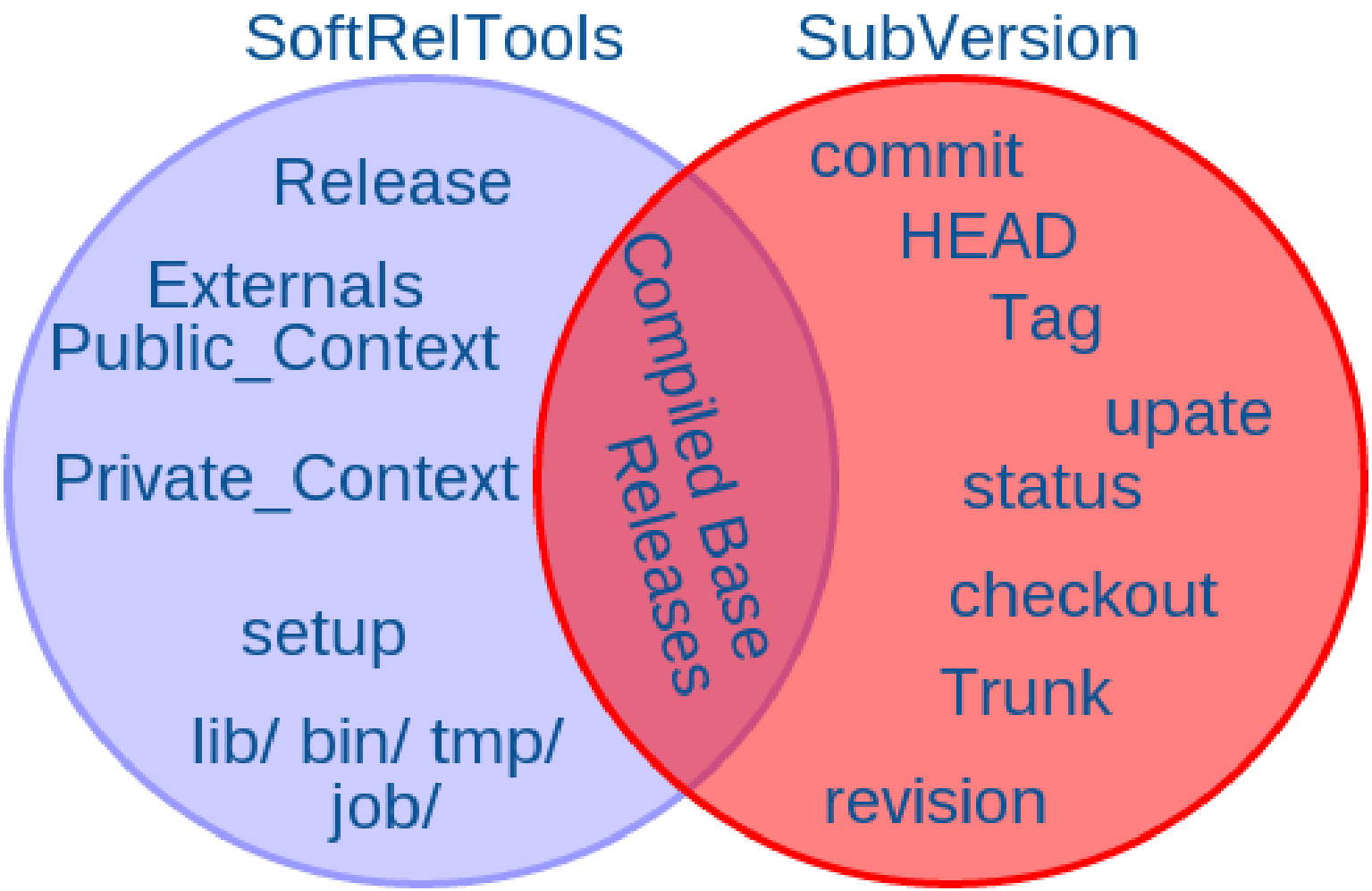
SoftRelTools (SRT)

- SRT is what handles the code on the machines
  - Sets up the environment
  - Assists in the build of packages
  - Ties in with our code repository
- Good idea to define setup function in `~/.bash_profile`

```
function setup_nova
{
 source /grid/fermiapp/nova/novaart/novasvn/
 srt/srt.sh
 export EXTERNALS=/nusoft/app/externals
 source $SRT_DIST/setup/setup_novasoft.sh "$@"
 cd /nova/app/users/janzirn
}
```

- Check the Beginner Wiki for a more elaborate definition

Peace and Harmony



Ready to Run out of the box (Spoiler)

- You are now ready to run code, as it exists in the development release

```
nova -c evd.fcl -s
/nova/data/novaroot/FarDet/S13-09-04/000124/12439/numi/
fardet_r00012439_s10_t00_numi_S13-09-04_v1.data.daq.root
```

- Chris' talk will cover ART and the things we do with it
- You can also run out of the box out of a tag, just specify during the setup stage `setup_nova -r S14-03-24`
- We also provide a `maxopt` version of the build, which you should use when you're not developing, but actually running the code
  `setup_nova -r S14-03-24 -b maxopt`
- Default is the debug version of the development release

What's a Test Release and how do I get one?

- You develop code in your own test release
- A test release is tied to a base release (development, tags)
- In your apps area do:
  `newrel -t development myTestRel`
- This creates a new folder called `myTestRel`, based on the development release
- Has the skeleton of a release
  - bin, lib, tmp, include, job dirs
- Only needs to be set-up once

How do I use my test release?

- You need to let SRT know where you're developing code, so it can set the correct library and include paths (among other things) under the hood

```
cd myTestRel
srt_setup -a
```

- Do this every time you are developing code from a new shell
- Be careful of release skew, your test release's base release needs to be the same as the release you specified when you set up NOνASoft

Where's the code at?

- The base release you're working off of is located in `$SRT_PUBCLIC_CONTEXT`
- This is where the compiled code is located and what you're running with if you don't use a test release
- Do not copy code from there into your test release to develop code
- Use `addpkg_svn` instead
- This will pull a version controlled copy from the repository into your test release

Quick SVN interlude

- We use svn for version controlling our code
- All changes go into the trunk of the repository
- The trunk is checked out nightly, compiled, and published to where you can use it as a base release
- **This means tomorrow's development release is slightly different than today's**
- We make copies of the trunk at certain intervals, these become the tagged releases
- More on this in a bit, now back to the main program

Tell me more about working with test releases

- Assuming you're developing code and have a test release based on the development release
  `addpkg_svn -h Demo`
- Pulls a copy of the `HEAD` (most up to date) version of the Demo package into your test release
- It also correctly identifies the `include` directory in the package and places a symbolic link in your test release's `include` directory to that location
- This is why you shouldn't just use the bare `svn checkout` command
- What if for some reason you didn't want the most up to date version
  - Run `addpkg_svn Demo <tag-name>` instead, where `<tag-name>` is something like `S14-03-25` for example
  - Alternatively if you know what revision you want exactly, grab the head version and then *update* it to the revision you want, `svn update Demo -r 8456`

That's exciting, then what?

- Edit the file you want to change in your favorite text editor
- Use `make` to compile the changes
- Assuming no errors during compilation, test the code and verify it does what you meant for it to do
- Commit the changes with an intelligent log message
- You're developing code!
- Well, almost. You should run some more checks before committing.

I wouldn't want to inconvenience anybody else with my commits, what should I check?

- Other developers might have committed changes in the meantime
- Check for those: `svn status -u Demo`
- If there are changes to files you're not working on, update your working copy
  `svn update Demo`
- If there are changes to the files you're working on, check what they are
  `svn diff -r HEAD Demo`
  - If they don't conflict with your changes, update your working copy
  - If they do conflict with your changes, coordinate with the other developer to understand what the two of you are trying to accomplish

That sounds important, what else should I look out for?

- Have you made changes to an existing function?
  - Check the rest of the code to see where else this function is used, `ack` is your friend
  - Check out those packages and make changes if necessary
  - Compile those packages and verify you didn't break anything inadvertantly
  - Use `novasoft_build -t` to build all packages in your release in the correct order to avoid linking issues during compilation
- Did you modify a data- or reco-object stored in an ART event (Evan's talk)?
  - Update the class version in the `classes.h` file of the package.
- These are more advanced topics and you should let Kanika know if you're changing much of the code so she can keep an eye on the build

I'm developing code, everything worked fine yesterday, but nothing works today

- Check the build log from last night: `http://nusoft.fnal.gov/nova/novasoft/logs/debuglogs/latest_build.html`
- If there are errors, in red at the top, and Kanika hasn't sent an e-mail around saying she has things under control, shoot her a quick e-mail and let her know that there were errors in the build
- If there are no errors with the build, it's likely theat one of the packages in your release got updated and you are running behind.
  - Clean your release: `novasoft_build -t -clean`
  - Update the packages in your release `update-release -t`
  - Compile the packages in your release `novasoft_build -t`
  - These steps will only work if all packages you're working with are part of a base release

I'm still having trouble getting my code to compile/run

- Check the troubleshooting section of our wiki: `https://cdcvs.fnal.gov/redmine/projects/novaart/wiki/Trouble_Shooting_and_Gotchas`
- Search the email archive to see if someone else has encountered, and found a fix to, your problem `http://listserv.fnal.gov/archives/nova_offline.html`
- Use a debugger to narrow down the problem, assuming your code compiles
- If you can't figure it out, and have made a decent effort to solve the problem, send an email to the listserv with the following information
  - What job, exact incantation, are you trying to run?
  - Have you cleaned, updated, and built your test release?
  - What's the problem you encountered?
  - What have you already tried to fix it?

Other Useful Things

- Sign up for the `nova_offline` listserv
- Sign up for the `novasvncommit` listserv
- Browse code via the redmine repository interface `https://cdcvs.fnal.gov/redmine/projects/novaart/repository`
- Browse code via the Doxygen source code browser `http://nusoft.fnal.gov/nova/novasoft/doxygen/html/index.html`
- Follow best practices and conventions `https://cdcvs.fnal.gov/redmine/projects/novaart/wiki/Editing_Code`
- SVN documentation `http://svnbook.red-bean.com/en/1.6/index.html`
- And of course many more tutorials to follow!