Software Provisioning for ProtoDUNE-SP

Tom Junk (FNAL), Christoph Alt (ETHZ), David Adams (BNL)

DUNE Collaboration Meeting

February 2, 2018

More information: See the Wednesday tutorial – Robert, Dorota, and Xavi's talks
https://indico.fnal.gov/event/16218/

Fermilab DUNE

The DUNE Software Management Group

- Co-conveners: Tom Junk, Christoph Alt, David Adams
  - dunetpc librarian: Christoph Alt
    - Merges branches
    - Tags releases
    - Review Continuous Integration test results
  - Release manager: David Adams
    - Build official releases
    - Install in /grid/fermiapp, CVMFS, and scisoft
  - Coordination: Tom Junk
    - Liaison with *art*, LArSoft, *artdaq*, and ProtoDUNE-SP DAQ group
    - Fill gaps and respond to problems and user requests
  - Active members: Tingjun Yang (former librarian), Gavin Davies

Software Distribution on DUNE

- Preferred Methods:
  - **CVMFS** -- static (tagged and installed releases)
  - **User Tarballs** for user-defined code that may change from job to job
    - copy to the worker node (some scaling issues here)
    - wget them from a web server using a local squid cache
  - **http://scisoft.fnal.gov** -- Tarballs of tagged releases. Pre-built and source code tarballs. Not meant for distributing code to batch workers, but people use it to install code on their own computers.
  - **Shared Disk Mounts:** /afs/cern.ch/user/<letter>/<name>
    Old: /dune/app and /grid/fermiapp at Fermilab – dismounted from batch workers Jan 2018
  - **StashCache** for larger static data files – locally cached at the grid site.
  - **Check out the source code from git repositories**

CVMFS – Distributed filesystem

https://wiki.dunescience.org/wiki/DUNE_Computing/Access_files_in_CVMFS

- Documentation and best practices:
  https://cdcvs.fnal.gov/redmine/projects/fife/wiki/Introduction_to_FIFE_and_Component_Services#OASISCVMFS

  https://indico.fnal.gov/getFile.py/access?contribId=26&resId=0&materialId=slides&confId=9737

  https://cernvm.cern.ch/portal/filesystem/repository-limit
- Only authorized people can add files to CVMFS. Talk to Tom, David, or Christoph to get on the list if you need it.
- Files in CVMFS are publicly accessible
- Two repositories
  dune.opensciencegrid.org for DUNE-specific code, and
  fermilab.opensciencegrid.org

CVMFS Features

- Files are written to the DUNE repository by cvmfsdune@oasiscfs02.fnal.gov
- Need to be in the .k5login to have access
- CVMFS stores only one copy of files across all releases and distributions. If a release has a copy of the same file as another, only one copy is stored
- Files generally persist "forever" in CVMFS
- We generally put UPS products in CVMFS. Source, include files, built objects, and configuration files.
- Makefiles generally not included in built UPS products – get out of git.
- Not restricted to just UPS products in CVMFS

Some CVMFS Operational Issues

- Need to wait ~20 minutes for files to propagate – remote sites must be made aware of new files
- Anecdotal issues Tom sees:
  - poor performance on Mac OS 10.12.6. cvmfs2 process looks CPU bound. I hear it might be better on Mac OS 10.13.x.
  - fermilab.opensciencegrid.org was not readable on my laptop in the CERN hostel until I turned on VPN. (i/o errors). It was readable on lxplus.

User-Defined Tarballs

- Recommended way for users to communicate their own code to batch workers
- For distributing changed pieces of the software stack. Most software should still come from CVMFS
- Put tarballs in dCache at Fermilab or a web server or anyplace your job can access them
- Issue is performance when many thousands of jobs download the same tarball all at once
- wget with local squid cache is optimized for performance
- Fermilab SCD is looking at alternatives, such as temporary CVMFS space

Shared Disk Mounts

- AFS (CERN), and up until recently, /dune/app (Fermilab)
- cernbox
- /mnt/nas00 (neut cluster)
- DAQ software is all locally built and run as far as I know
- The easiest way to distribute software
- No version control built in – do it yourself. Files can change on disk while jobs are queued up.
- Performance issues when many jobs access the same disk
- For distributing changed pieces of the software stack. Most software should still come from CVMFS

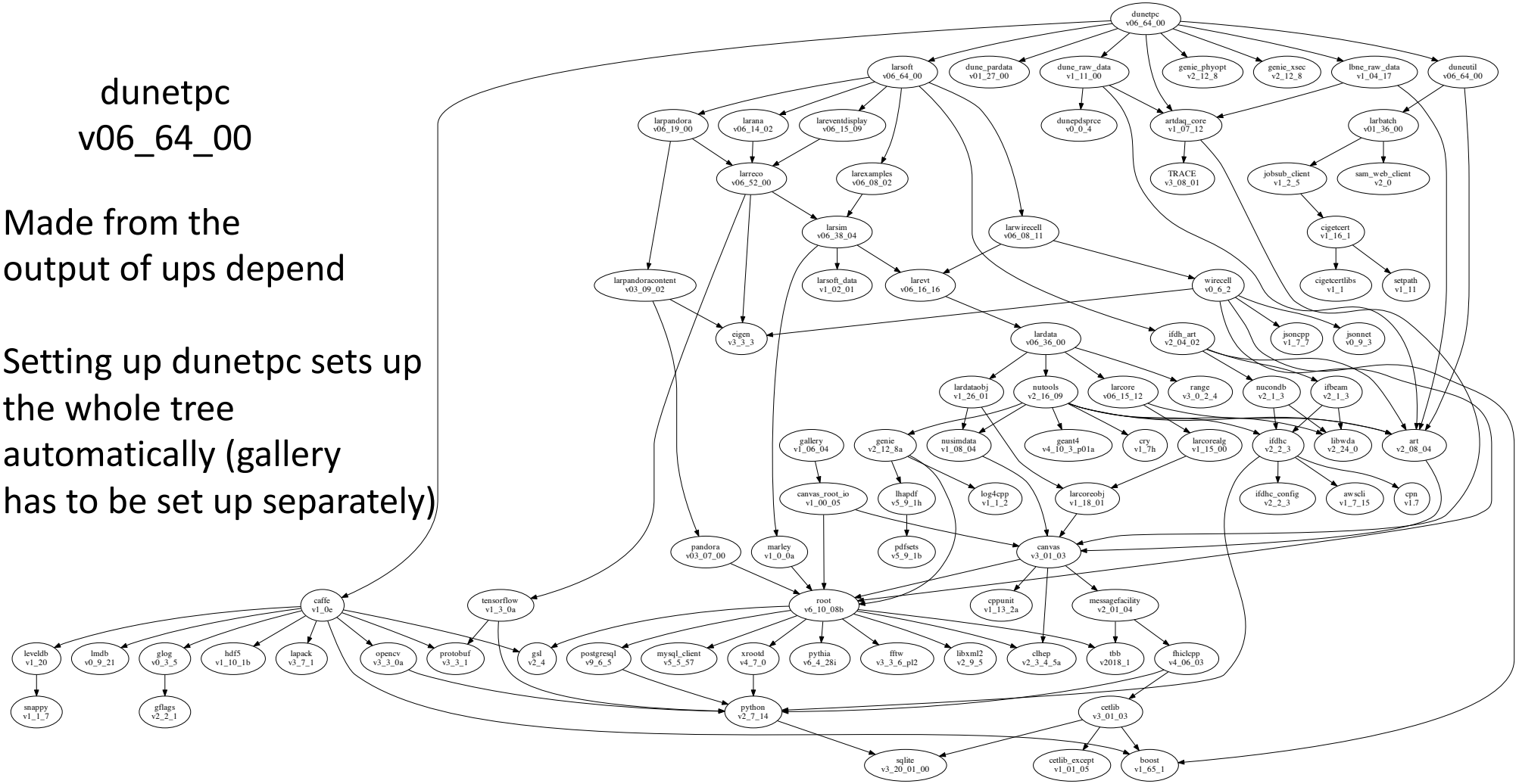
- This is how Dorota built dunetpc for DQM purposes.

dunetpc Dependency Diagram

dunetpc
v06_64_00

Made from the output of ups depend

Setting up dunetpc sets up the whole tree automatically (gallery has to be set up separately)



*GEANT4 dependencies not shown. gcc and cetbuildtools also not shown. Otherwise too messy

Release Schedule

- LArSoft releases Weekly (!)
- Intention is to reduce the instances of develop heads of many repositories depending on each other. Get it released!
- Since UPS sets up dependent products automatically, and setting up dunetpc sets up all of LArSoft, we need a new version of dunetpc for every version of LArSoft we want to work with
- Release numbers are synchronized between LArSoft and dunetpc just to keep users from having to know the map and giving us a basis for communication
- We can release more often as needed. Just not less often
- It takes ~1 day to tag, build, and deploy releases.
- **A release of dunetpc takes about 3 GB of space**

dunetpc, dune_raw_data, lbne_raw_data dunepdsprce

- lbne_raw_data was designed to house code that was used online and offline, such as data format definitions and unpackers.
- separate builds for online and offline
  - online requires stable software environment – usually built with older releases of *art*, *artdaq_core*, etc.
  - offline releases update quickly to new versions of products.
- lbne_raw_data is only used for unpacking 35-ton prototype data. No more online builds for it are necessary.
- Every time *art* or *artdaq_core* gets a new release, we need a new releas of dune_raw_data and lbne_raw_data for offline use
- So far we've been lucky and seprate code hasn't been needed to satisfy an API change in a dependent product

dunetpc, dune_raw_data, lbne_raw_data dunepdsprce

- Decoder modules live in dunetpc
- Channel map lives in dune_raw_data
- dune_raw_data and lbne_raw_data have separate Jenkins builds.
- dunetpc's Jenkins build also builds lbne_raw_data and dune_raw_data, but these can be removed. Probably historical.
- Need coordination for each release that requires a new dune_raw_data (or lbne_raw_data). Happens when any dependent product gets updated.

dunetpc, dune_raw_data, lbne_raw_data dunepdsprce

- dunepdsprce contains JJ Russell's RCE unpacker code
  - unpacks WIB frames into wavefroms
  - Need a version that handles compressed data
- The decoder modules (Written by Jingbo Wang) started out in an online monitoring feature branch of dunetpc.
- This feature branch is built online with an old set of dependencies (*art, artdaq_core*).
- Raw data need to be decoded for offline use too
- Versions of the decoder modules now live in the develop branch of dunetpc
- But *artdaq_core*'s API has evolved. So some minor changes had to be made to the offline versions of the modules.
- Separate maintenance of online and offline raw decoder modules: RCE, FELIX, SSP, Timing

Developing with dunetpc

- See the tutorials at the August 2017 collaboration meeting satellite meeting on Computing

https://indico.fnal.gov/event/14943/other-view?view=standard

- Two kinds of code -- two ways to get it for developing
  - set up pre-built code (it takes a long time to build all code yourself)
    - done with ups.
      source /cvmfs/dune.opensciencegrid.org/products/dune/setup_dune.sh
      setup dunetpc v06_66_00 –q e14:prof
  - check out source code you would like to modify
    - done with mrb (multi-repository build) mrb g dunetpc – more later

Examples

- Set up larsoft, check out dunetpc, modify a module, build it, and make a local ups product for dunetpc

https://wiki.dunescience.org/wiki/DUNE_Computing/Analysis_Module_From_Example_August2017

- Instructions for committing your code back to dunetpc (or a larsoft repo). Feature branches, pulling and pushing

  https://cdcvs.fnal.gov/redmine/projects/dunetpc/wiki/_Tutorial_

Example script for setting up a new development directory

```
#!/bin/sh

LARSOFT_VERSION=v06_66_00
COMPILER=e14
DIRECTORY=develop_dunetpc_${LARSOFT_VERSION}

source /cvmfs/dune.opensciencegrid.org/products/dune/setup_dune.sh

touch ${DIRECTORY}
rm -rf ${DIRECTORY}
mkdir ${DIRECTORY}
cd ${DIRECTORY}
setup larsoft ${LARSOFT_VERSION} -q ${COMPILER}:prof
mrb newDev
source localProducts_larsoft_${LARSOFT_VERSION}_${COMPILER}_prof/setup
mkdir work
cd srcs
mrb g -t ${LARSOFT_VERSION} dunetpc

cd ../build_slf6.x86_64/
mrbsetenv
mrb i -j2
```

Relocating Local Products

- Your build of dunetpc will need three directories
  - source
  - build -- we do an "out-of-source" build
  - LocalProducts -- installed built objects
- There's a setup script in LocalProducts that adds to LD_LIBRARY_PATH, FW_SEARCH_PATH, FHICL_FILE_PATH, and defines things like DUNETPC_DIR
- Problem is, this script has hardcoded path names (made by mrb when it is created).
- mrb mp makes a relocatable tarball that can be unwound on a batch worker and set up.
- If you build on AFS but need to run out of eos, this will be needed.
- Project.py takes another approach to tarring up local products if --localtar is specified.– it edits the Local Products setup script with sed

Work to Do for DQM

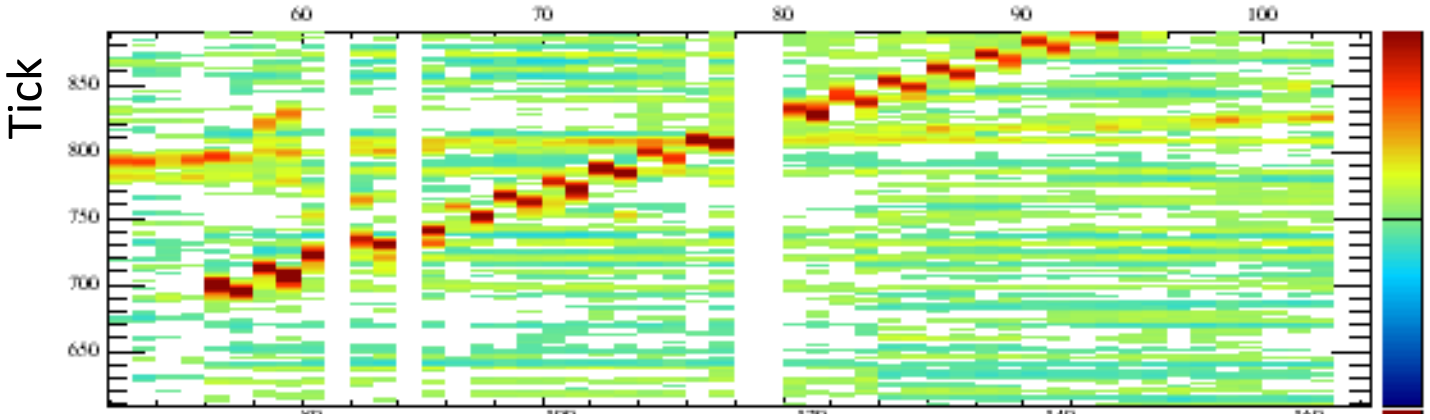
- Data size and integrity (are all expected data products present?) Which subdetectors are contributing?
- Compression ratio and CPU time summaries
- Modules for making histograms of ADC mean and RMS (anode & PD)
  - Histograms arranged in "physical" wire arrangement (collection, U, V wires together)
  - Histograms in "hardware" space. Channels grouped by ASIC, FEMB, WIB
  - How to show 15K channels' data effectively (one set of histos/APA)
- Pedestal calculation (median, or mean with signal rejection) and auto upload to a database
- Gain calculation for each channel
- FFT plots
- Event displays
- Stuck-bit Summaries (stuck codes and individual stuck bits). Recoverable and unrecoverable runs of stuck code rates.

Work to do for DQM

- Beam instrumentation summaries
- Trigger summaries (trigger bits)
- More ambitious modules:
  - hit wire and time distributions – wire occupancies
  - Induction hit charge asymmetry
  - Flash time distributions
  - hit charge distributions
  - track distributions. Length, position, angle, sum charge
    Gain calculations may require track finding
  - shower distributions. position, angle, sum charge
  - Bruce's purity module
  - Really ambitious: beam/TPC association. Maybe just timing differences (hit/beam) at first.

Validating the Channel Map

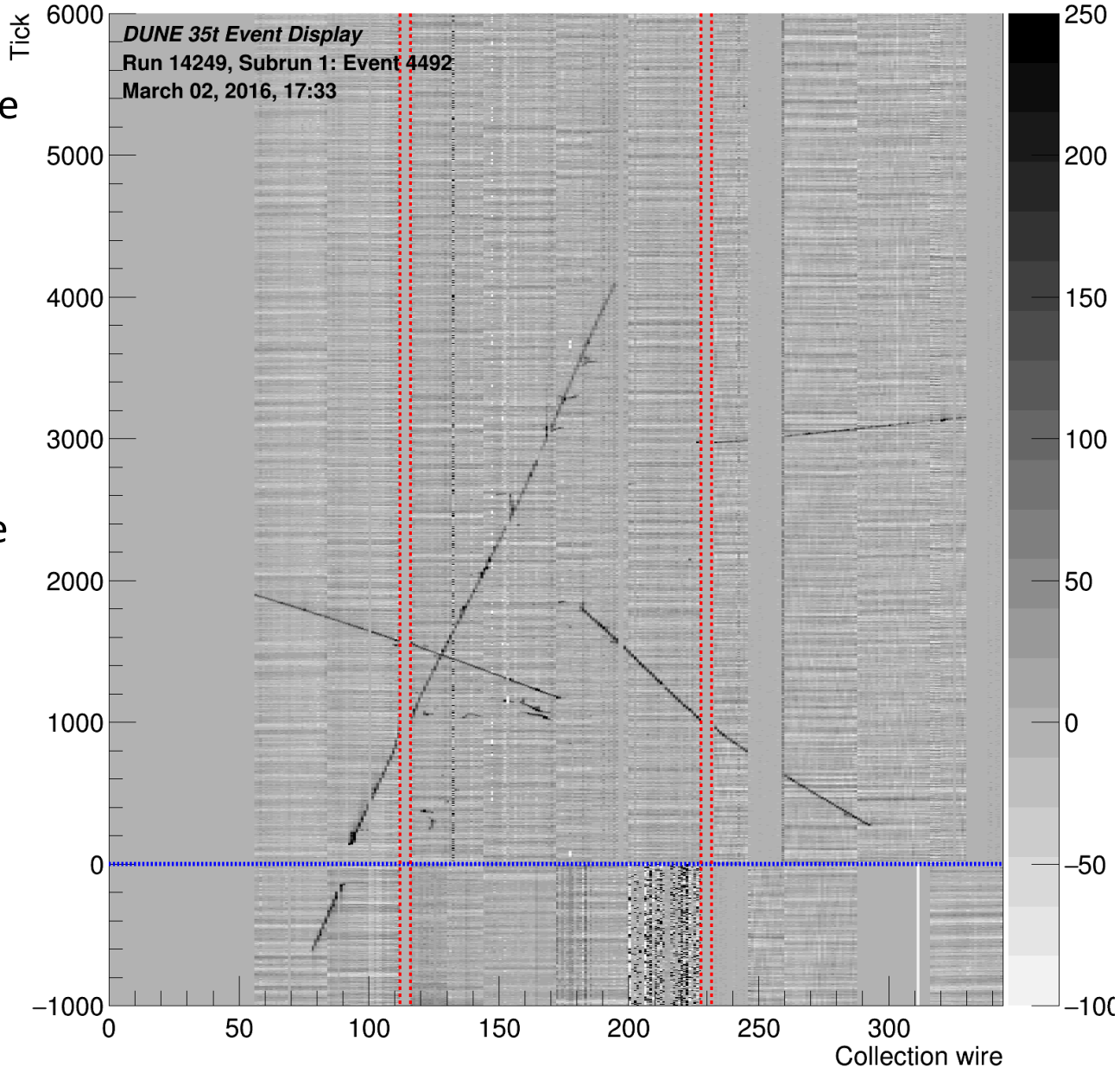
- Example from 35t: even and odd collection-plane channels were swapped in the channel map. Zig-zag tracks (visible for non-isochronous tracks). Maybe a ribbon cable was plugged in backwards?
- Other swaps possible. APA put in backwards (symmetrical, but bad channels). Channels numbered backwards.



A 35-ton Event Display With an APA Crosser and three Z Crossers

Induction-plane S/N was poor so we didn't get to align across the only horizontal gap in 35-ton

We studied the APA crossers and the Z-gap crossers



We won't get APA crossers in ProtoDUNE-SP (well, maybe...)

We will get them in the central APA's in the DUNE SP FD

Extras

Note on v06_65_00

- LArSoft v06_65_00 was built on art v2_09_06, nutools v2_17_02 with LArSoft v06_64_00's code.
- It is not last week's weekly release, and is just a test of building LArSoft with the upgraded dependent products.
- We therefore skipped it when building dunetpc v06_66_00. No new LArSoft features were available in v06_65_00.

LArSoft Priorities for the Year

- Investigate having a new Event Display framework
  - Multi-TPC detectors
  - Zoom and pan
  - Interfaces to art and larsoft services. E.g. interactively rerun reco, or get run conditions info.
- Add support for LAr TPC's
- SIMD Vectorization
- Optimization and Profiling Work
- Error Handling Policy
- Global Wire Coordinates
- Support for computing hardware architecture-dependent libraries (e.g. GPU or no, SIMD or no)

LArSoft Priorities for the Year cont'd

- LArG4 Refactoring Work
  - Expose the GEANT4 steps as persistable data products
  - Abstract the anode-plane simulation so single-phase, dual-phase, and pixel detectors can share simulation code
- Migrate to SPACK

Software Provisioning on Batch Workers

- The problem: tens of thousands of reads of the same tarball in dCache can cause performance problems that affect all experiments.
- FNAL's SCD is exploring solutions. They are collecting requirements first
  - DUNE requires at least a much functionality as we have now with CVMFS and tarballs. Code must be accessible and run on batch workers the same way as on interactive nodes.
  - Ease of use and documentation are required.
- Another problem: CVMFS is not allowed on NERSC HPC's and presumably others have similar policies.
  - Solution here: Docker containers + tarballs. It would be great to have a uniform interface however.