# Code Management Issues

Jake Calcutt, Tom Junk

DUNE Computing Essentials

August 31, 2022

# Who We Are

- Code Managers:
  - David Adams (was: full time, now part time on DUNE)
  - Jake Calcutt
  - Tom Junk
  - Gavin Davies
- Many developers and contributors
  - Especially calling out Brett Viren, Tingjun Yang, Tammy Walton, Alex Himmel, Leigh Whitehead, Dom Brailsford, Wenqiang Gu, Nitish Nayak, Phil Rodrigues, Barnali Chowdhury, and many others I am forgetting here.
- Support from SciSoft/art/LArSoft developers and staff:
  - Lynn Garren, Erica Snider, Kyle Knoepfel, Chris Geen, Katherine Lato

## What We Do

- Design DUNE's software architecture
  - Well, okay, we actually work with stakeholders to do that; we don't just make pronouncements of how things are done.
  - Some software architecture decisions are made for expediency or compatibility with external products.
- Components we are directly involved with
  - FD, ProtoDUNEs, coldboxes, ICEBERG, ND-GAr
- Some DUNE software is developed by independent groups:
  - Most Near-Detector components  (SAND is factorized.  TMS and ND-LAr work together but manage their own software)
  - Beam Simulations
  - Most end-user physics:  LBL tools, BSM.

# What We Do

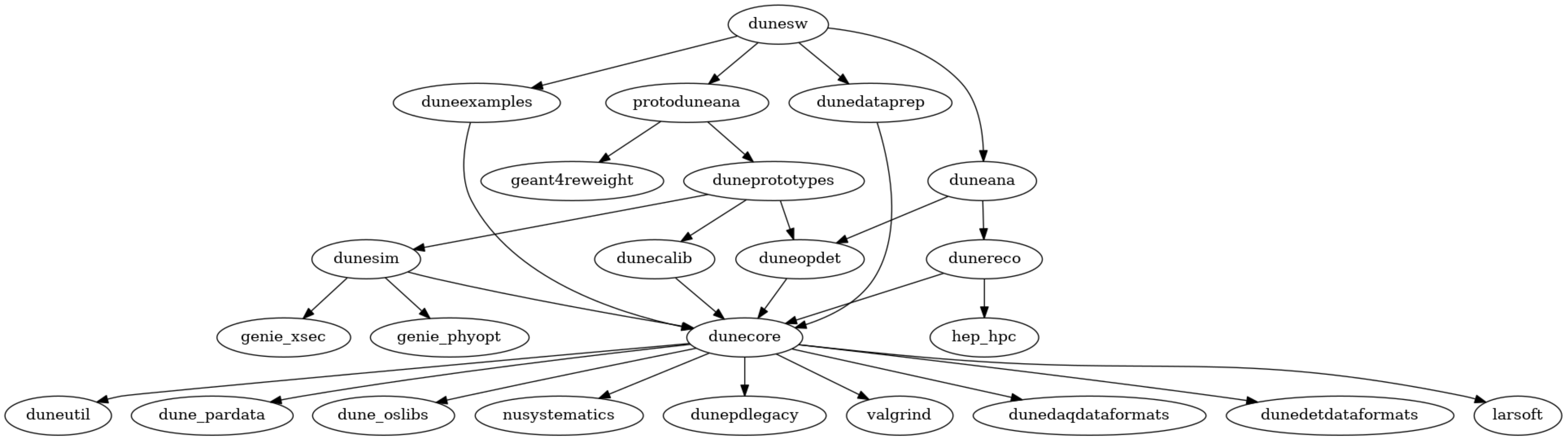
- Weekly releases of the *art*+LArSoft-based DUNE stack
  - following the LArSoft releases.  DUNE developers want the latest and greatest
  - build with Jenkins, publish in CVMFS and scisoft.fnal.gov
- Maintenance
  - changes required by the *art*+LArSoft ecosystem:  examples:  makefile upgrades, Jenkins build script upgrades, replacing getByLabel with getHandle
  - Fixes needed because of upgrades
    - New compiler versions flag various things as warnings which LArSoft's build system treats as errors.  Fixes for legacy code that fails to compile when new externals/compilers are brought in.
    - Handling deprecations  (SciSoft removed HDF5's C++ interface went away in the LArSoft distribution, for example, and we had to re-code some things)
  - Maintain the setup_dune.sh script

## What We Do

- Releases of DUNE-specific externals
  - dune_raw_data and dunepdsprce:   now dunepdlegacy – DAQ-group-provided code
    - ongoing maintenance of legacy software so we can read in ProtoDUNE-SP data for the foreseeable future.
  - dunedetdataformats, dunedaqdataformats  (installed UPS versions of dune-daq/detdataformats and dune-daq/daqdataformats)
    - These two involved some negotiations with the DAQ group to ensure portability.  DAQ-specific tools are not available offline, so we had to find solutions we could throw over the wall.
  - duneanaobj
  - edepsim:  a UPS version of Clark McGrew's product
  - garanaobj:  Chris Higenberg wrote it.

# What We Do

- Split dunetpc into 11 other repositories and move it all to GitHub
  Many thanks to Jake and Tom!
- Split occurred January 2022, and we expect further evolution of the product stack as DUNE's needs grow



# What We Do

- Interface with *art*, LArSoft and SciSoft teams
  - Attend *art* stakeholders' meetings
  - Attend LArSoft Coordination, Steering Group and Offline Leads meetings
  - Review and approve pull requests for LArSoft. Even those for DUNE software! The DUNE ones are not yet mandatory, but people are using them nonetheless.
  - Learn about necessary maintenance for DUNE software from evolution in the *art*/LArSoft stack
  - Provide feedback on physics needs (GEANT4, GENIE)
  - Tell the LArSoft team which versions need to be preserved
  - Clean up our own CVMFS area from DUNE software that corresponds to deleted LArSoft versions (Thanks, Gavin!)

## What We Do

- Monitor and maintain the DUNE Continuous Integration tests
- We get a lot of help from Vito DiBenedetto on the CI system
- CI tests are an important validation step for evaluating LArSoft pull requests.
- Check to see that the builds work and code still runs.
- Chase after users who break things.

- Test new systems like the Elastic Analysis Facility to see if they can build and run DUNE software. Fix problems uncovered in the process.

## What we do

- Documentation
  - Wikis on the DUNE wiki, Redmine, and GitHub
- Training
  - Collaboration-meeting training episodes and wikis
- User support
  - Respond to questions/requests for help on Slack
  - e-mail
  - walk-ins
  - SNOW tickets that get sent our way
- Architecture discussions with framework experts – membership on frameworks task force, monitor ongoing R&D projects and provide advice on DUNE's needs.

# The To-Do List

- Lock down repositories in GitHub so that only pull requests can be made
- Currently people in the DUNE organization can push to DUNE's repositories.  A risk if people force pushes that overwrite or delete other peoples' work.
- LArSoft model:  CMS bot evaluates pull requests with CI system and enforces review
  - L2 managers perform review and approve PRs
  - L1 managers merge PRs and make releases
- We can do this and even have a list of possible L2 managers.
- A concern – PRs can languish in review, especially if no one feels qualified to perform the review.  Reviewers need to be nagged sometimes.  Usually, they are responsive, and they know that without them, the code would be pushed anyway.

## The To-Do List

- Spack migration
  - Modernize CMakeLists.txt files. Currently passes the requirements of the migrate script, but there is more work to be done
  - Change find_ups_product to find_package (mostly done in GArSoft as an example)
  - ROOT library lists need to be separately specified. Currently we use ROOT_BASIC_LIB_LIST which goes away when we remove find_ups product
  - We tried out Marc Mengel's Spack instructions a couple of years ago.
  - We are keeping an eye on it.

## To-Do List

- Improve software organization, build out the package list
- Where does FD-specific code go?
  - duneprototypes is a catch-all for the ProtoDUNEs, coldboxes, and ICEBERG
  - Need FD-specific products for similar things.   Channel maps, data ingestion, interfaces with databases, analysis tools.
  - Several FD module types will make this more challenging
  - Would like to share code whenever possible.
  - Sometimes it helps to fork code for detectors as each detector evolves over time and we don't want to break one detector when another one needs a change.
- Improve fcl dependencies (services_dune.fcl and tools_dune.fcl still tie everything together).  May need a separate head product for each detector to straighten that one out.  Currenty need to set up dunesw regardless of what you're doing.

## The To-Do List

- More software-development tasks rather than software management tasks:
  - Reduce memory consumption of FD sim/reco!
  - Write code that runs on GPUs as well as CPUs
  - Write code that is thread safe and runs in multi-threaded mode

- Of these, reducing memory consumption is the highest priority. Needed for optimal use of multi-threaded applications anyway.

## The To-Do List

- Improve documentation
  - Centralize
    - People may question the newness of the Redmine wiki for dunetpc when the code has migrated away.
    - GitHub and the DUNE Wiki are currently being used.
    - GitHub wiki does not need authentication:  search engines can index it.
  - Prune out-of-date instructions

# The To-Do List

- Roll out a copyright policy
- Talked with Liz Sexton-Kennedy and Aaron Sauers about this.
- The FNAL Prime Contract actually has a few pages on this.
- We can choose any Free and Open Source license we like
- It is mainly there to protect us from ourselves. DUNE developers by default are granted copyright even if we do nothing. A developer may leave in a huff though, and deny us the use of DUNE software. Or try to sell it to us.
- LArSoft uses the Apache 2.0 license. Battle-tested in court.
- Need to assert copyright (FRA) over contributions.
- FNAL Prime Contract says we can do this for DOE-supplied code with 2 weeks notice. Silent on contributions from non-DOE-funded entities.