The New protoduneana Repository

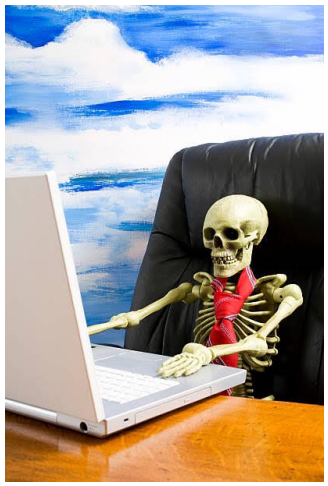
Tom Junk

ProtoDUNE Sim/Reco Meeting

Oct 16, 2019

The Problem

- dunetpc builds take of order 1 hour of CPU.
- Spread across 16 cores of a build node, this isn't so bad, but get two people using a build node at the same time, it is painful
- Initial git clone of dunetpc is slow (~5 minutes, worse when using read-only https URL), and depends on load on redmine server.
- cmake step is single-threaded and slow (~5 minutes)



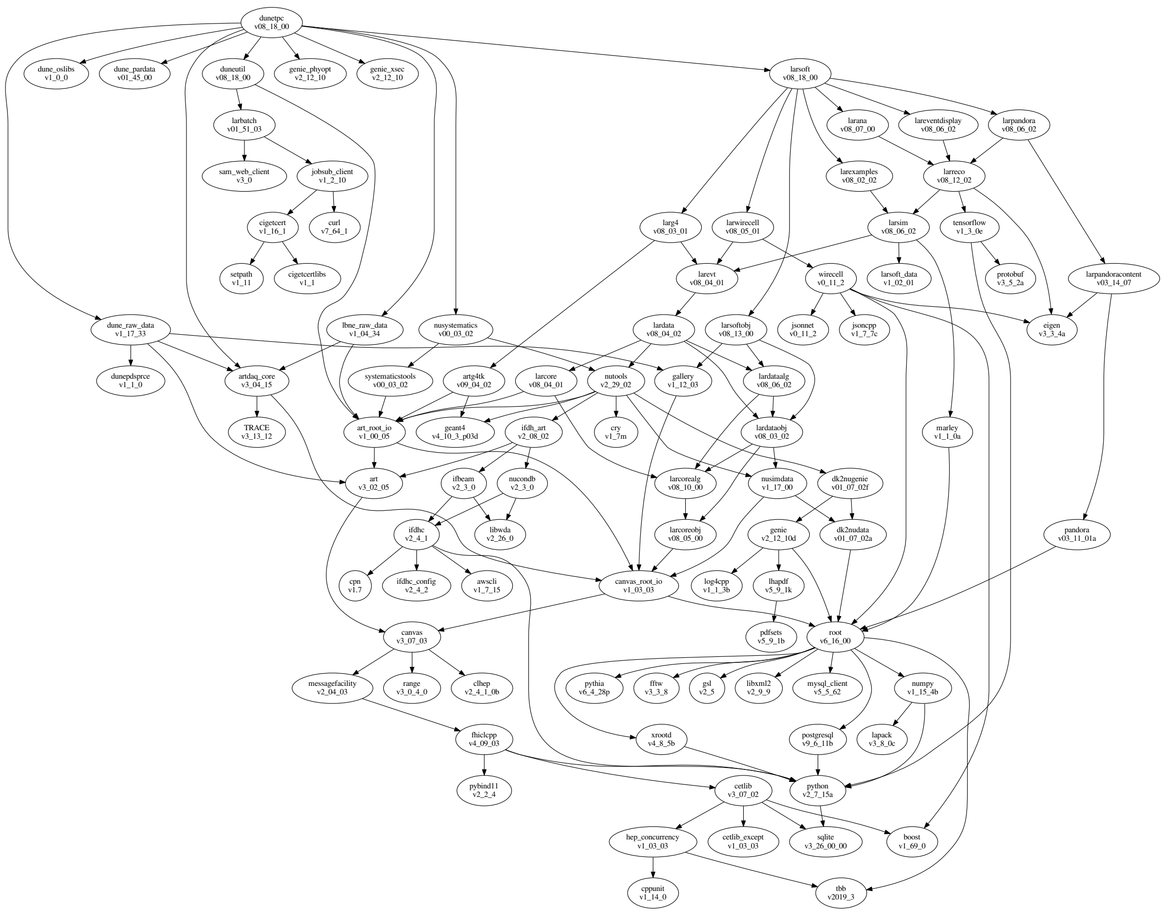
ProtoDUNE analyzer waiting for dunetpc to finish building

LArSoft's Solution

- LArSoft was split into many repositories even before migrating from SVN to git long ago. One UPS product per repo.

dunetpc already has a few pieces:

dunetpc
dune_raw_data
lbne_raw_data
duneutil
dunepdsprce



Fermilab DUNE

A Way Forwards

- We discussed this last week with LArSoft, SciSoft, and MicroBooNE experts.
- Splitting up dunetpc into fine-grained repo structure is the right thing to do
  - Remove unused code
  - Untangle circular dependencies
  - Migrate to buildFW for Jenkins build
- But involves lots of work. The above, plus an ongoing chain of tagging and releasing, keeping an entire tree up to date.
- David Adams's proposal is to split off pieces that are urgent and build them as we have been building our pieces.

protoduneana

- We took the expedient way forwards, lacking a large, committed personpower base to do things right.
- New redmine project protoduneana created as a subproject of dunetpc.
- Preliminary checkout and build of empty repo works.
- To do: Migrate analysis code into it and add its build to the dunetpc Jenkins script so we get consistent builds.
- dune software management proposes to do the initial migration of code in the develop branch
- Feature branches however are the responsibility of individual users.
- CI tests..