

Splitting dunetpc

Tom Junk

DUNE Software Architecture Meeting

July 7, 2021

We have been promising to split dunetpc into pieces for a long time now

- Time to get it done!
- dunetpc takes a loooong time to build, even on a build node
15 minutes (16 cores)
- longer if the build node is being used, or if you're using a less-powerful computer



Goals

- Split dunetpc so the builds are faster.
- Preserve functionality. setup a new top-level product and be able to do everything you used to do with dunetpc
- Make unit tests work
- Move to GitHub (I can be talked out of this one if need be)

Not Current Goals

- Make completely standalone products
 - see later discussion on fcl files
 - Dependencies should be a DAG so the build can be done consistently.
 - Unit tests should work, but not every fcl file may work unless you set up all of what was dunetpc
 - setup is much faster than building, so setting up everything is not a problem.
 - Want to do a protodune analysis? Set up dunesw, not just dune prototypes.
- Make DUNE code any better/prettier
- Incorporate Spack
- Further sub-splitting is possible as the software grows

dunetpc Has Grown Over the Years

- 58 subdirectories, in various states of maintenance
- dunetpc is a catch-all repository – 867 MB for source + git history
- Already have some dune UPS products split off:
 - protoduneana
 - duneutil
 - dune_raw_data
 - dunepdsprce
 - dune_oslibs
 - dune_pardata
- Not all are built from git source (dune_pardata is a standalone, and dune_oslibs is just a collection of files copied into it)

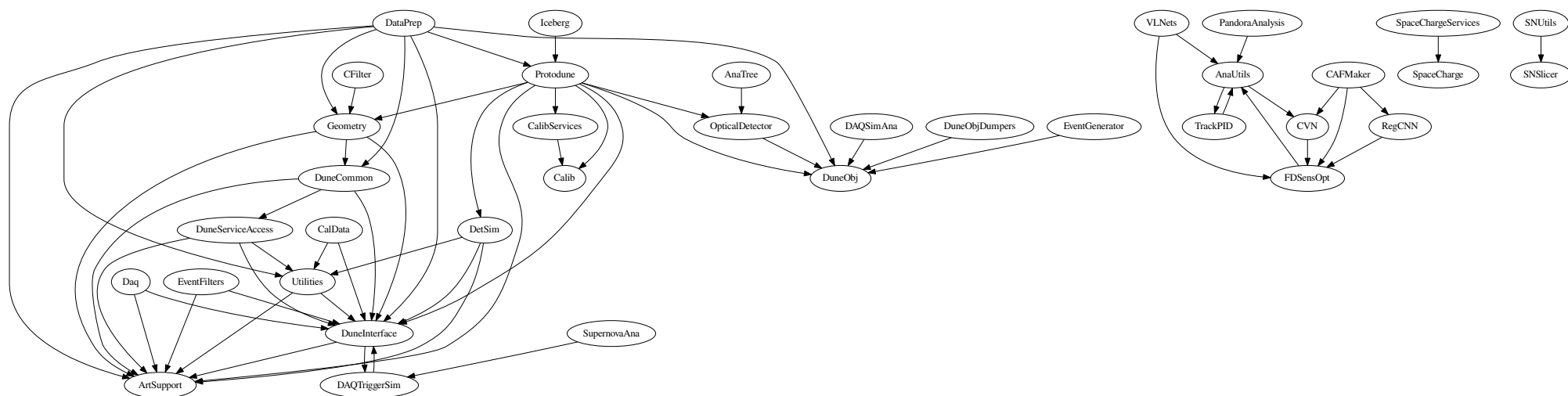
dunetpc's Environment

From `dependency_graph.sh`
which uses `ups depend`



larsoft is split into more pieces for build speed

Inside dunetpc -- #include dependencies



Only directories containing source files that #include files in other directories are shown.

Grouping Directories

- Proposed new products:
 - dunesw
 - dunecore
 - duneopdet (has sim, calib, reco and ana components)
 - dunesim
 - dunecalib
 - duneprototypes
 - dunedataprep
 - dunereco
 - duneana
 - duneexamples
 - duneconfig

The Grouping

dunetpc Directory

product

3x1x1dp	duneprototypes
APAIO	do not migrate
AnaTree	duneana
AnaUtils	dunereco
ArtSupport	dunecore
BeamData	duneprototypes
CAFMaker	duneana
CFilter	do not migrate
CVN	dunereco
CalData	dunedataprep
Calib	dunecalib
CalibServices	dunecalib
ClusterFinderDUNE	dunereco
DAQSimAna	duneana
DAQTriggerSim	dunecore
DUNEPandora	dunereco
DUNEWireCell	dunereco
Daq	do not migrate
DataPrep	dunedataprep
DetSim	dunesim
DetectorVariations	dunesim

The Grouping, cont'd

DuneCommon	dunecore
DuneExample	duneexamples
DuneInterface	dunecore
DuneObj	dunecore
DuneObjDumpers	dunecore
DuneServiceAccess	dunecore
EnergyStudies	duneana
EventFilters	duneana
EventGenerator	dunesim
FDSensOpt	dunereco
GalleryScripts	duneexamples
Geometry	dunecore
HitAnalysis	duneana
HitFinderDUNE	dunereco
Iceberg	duneprototypes
InfillChannels	dunereco
LArG4	dunesim
OpticalDetector	duneopdet
PandoraAnalysis	duneana
PhotonPropagation	dunesim
ProductFilters	duneana
Protodune	duneprototypes

The Grouping, cont'd

RecoAlgDUNE	dunereco
RegCNN	dunereco
SNSlicer	dunereco -- but not built (as per dunetpc)
SNUtils	dunereco -- but not built (as per dunetpc)
ShowerAna	duneana
SimFilter	dunesim
Simulation	dunesim
SpaceCharge	dunesim
SpaceChargeServices	dunesim
SupernovaAna	duneana
TrackFinderDUNE	dunereco
TrackPID	dunereco
TrackingAna	duneana
Utilities	dunecore
VLNets	dunereco

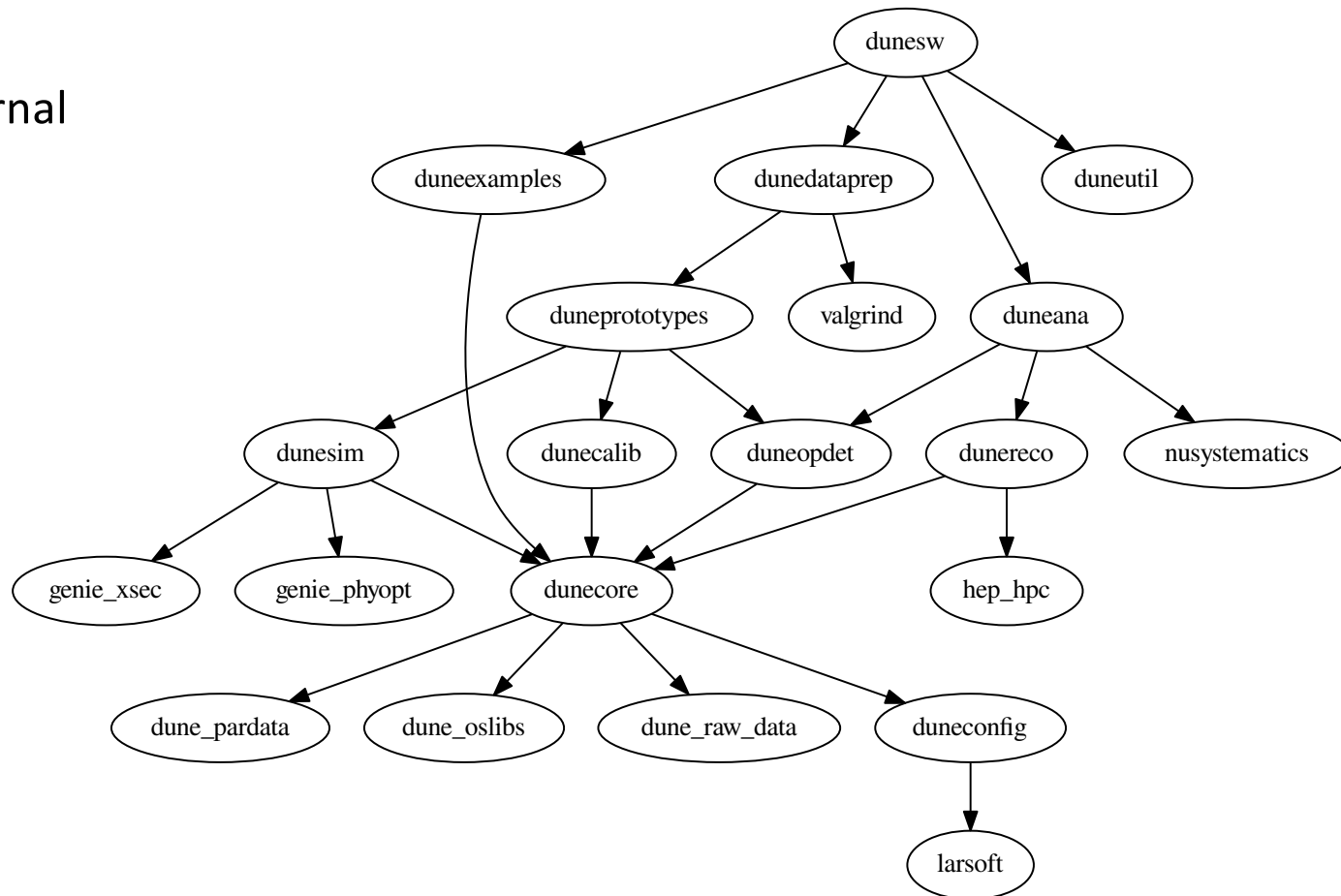
Split Scripts

- Copy code from a checked-out test release of dunetpc into a separate test release
- Supply new CMakeLists.txt files (for the main directory – subdirectories are fine as they are)
- Supply new product_deps files
- Edit `#include` statements to point to header files' new locations
- Edit CMakeLists.txt files to point to libraries' new names (they have their UPS product names in front)

These editing steps are done with sed, in scripts. Can be redone if we change our minds. And kept around to fix protoduneana and user code.

Proposed Split UPS Dep. Tree

Also shows external dependencies



Only one point of dependency on LArSoft – one number to update in product_deps, but need to update the version numbers of the dune products on each release.

To decide: Version numbers! Currently they are all the same: v09_25_00 for dune products except legacy ones like dune_oslibs and duneutil

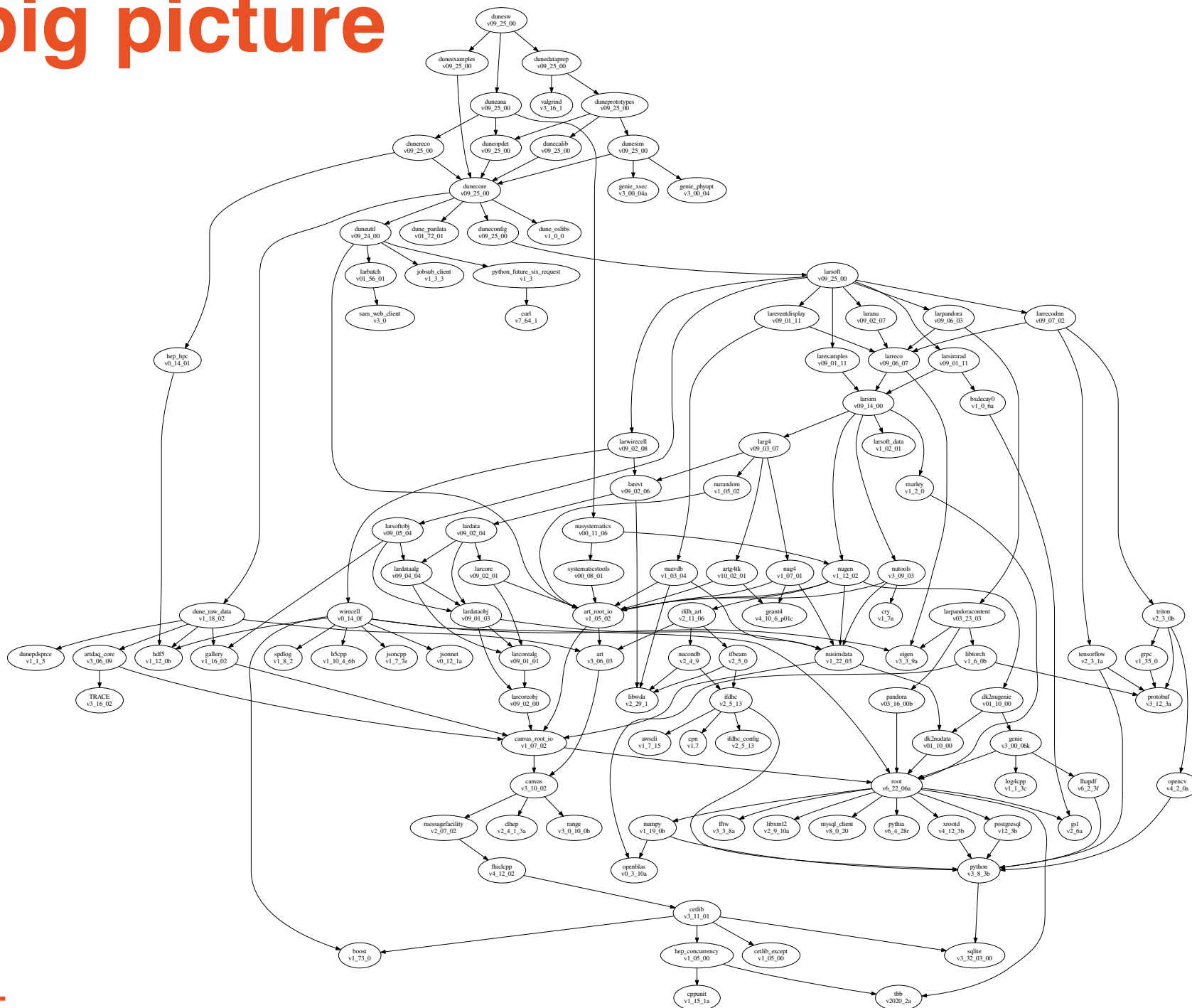
Splitting and Building

- I tested the build as I went along.
- Built from bottom to top.
- Installed products in a local PRODUCTS directory, set them up, and build split products based on them
- I found that some things didn't compile, and I had to fix some CMakeLists.txt files (missing libraries). Checked those changes in to develop
- SNAna and SNUtils were not built in dunetpc, left them unbuilt in duneana (I tried, but not too hard)

The Top-Level Product

- What to call it? dunesw? dunesoft? dunecode?
- dunetpc was good, but I'd like to get rid of it entirely
- Like larsoft or uboonecode, an "empty" product for setting up the whole stack
- Has some uses: fcl files (?), test suite, environment variable setup
- Where does protoduneana go?
 - Under dunesw: We'd then include it in our regular weekly release.
 - Above dunesw: Then it can be released on its own schedule.

The big picture



Would like to deploy to GitHub

- Collaborators prefer GitHub to Redmine
- At least the ones who express their preference to me.
- Redmine has blocked anonymous https clones from offsite. Need VPN for that. Or use the ssh clones.
- I believe there's nothing secret in our code, but I haven't looked at every line!
- LICENSE.txt file (copy LArSoft's?)
- Wiki pages (we use the Redmine wiki for a lot of things – now password-protected). Some things we may want to put behind a password wall.
- Disable pushes to Redmine, create repos in GitHub
- Allow collaborators to push (pull-request model? Need L2 and L1 managers).
- Not every DUNE collaborator is a member of the DUNE GitHub organization. Need to join. Or add users one by one.
- Also need to update the Jenkins build script. Use buildfw perhaps?
- Move duneutil and protoduneana to GitHub? dune-raw-data? dunepdsprce already is.

Progress since June 28

- I successfully ran tests on all the directories (but had set up the entire tree)
- Found that the 35t channel map fcl file was still being included by services_dune.fcl and removed it, making tests succeed.
- Found that many tests required fcl files in products above the test. services_dune.fcl includes simulationservices_dune.fcl for example, so a test under dunesim that includes services_dune.fcl will fail.
- I ran an event through the protodune refactored gen/g4/detsim/reco stages successfully.

Where to put fcl files?

- dunetpc/fcl has these subdirectories:
 - common
 - dune35t
 - dune fd
 - dune fddp
 - protodune
 - protodunedp
 - 3x1x1dp
 - evd

Fun fact: dunetpc has 1534 fcl files.

fcl files → duneconfig?

- Some tests in dunecore depend on `dunetpc/fcl/common/backtrackerservice_dune.fcl`
- fcl structure is not commensurate with proposed split structure.
 - dunesim
 - dunereco
 - duneana
- But `dunetpc/fcl/dunefd` has these directories: `gen`, `g4`, `detsim`, `reco`, `mergeana`.
- Would be nice to keep the fcl files together.
- Propose a new product, `duneconfig`, for storing fcl files not tied to a specific product, or which steer general workflows.
- Can put it at the bottom of the dependency graph. Under `dunecore`.
- Some CMake modules are in `canvas_root_io`, so I put `duneconfig` between `dunecore` and `larsoft`, to pick up all the art/canvas stuff

The problem with putting fcl files at the bottom of the dependency graph

- You end up with fcl files that call for running code that may not be set up. `module_type`, `service` etc. may correspond to code in products above the fcl file.
- fcl files in `duneconfig` might want to `#include` fcl files in products not set up. This in fact makes some unit tests in `dunecommon` fail. `services_dune.fcl` (`dunecore`) includes `simulationservices_dune.fcl` (`dunesim`).
- Setting up just `duneconfig` by itself would set up nothing else in `dune`, so none of the fcl files would work.
- "Don't do that then"

The problem with putting fcl files at the top of the dependency graph

- Then you cannot use them unless you set up the whole tree.
- If we want UPS products to stand by themselves, then some fcl files are impossible to place.

services_dune.fcl

tools_dune.fcl

- Unit tests are run on a per-product basis, without setting up products above.
- I had to put (at least some) fcl files at the bottom to make unit tests work.
- Possible to write a test that fails due to this but which tests a common workflow (`#include services_dune.fcl`, `#include tools_dune.fcl`)
- A solution: tests must provide configurations for those components that are to be tested and that are available, and not other things.
- Need to test the `services_dune.fcl` functionality? Put that test in `dunesw`.
- 188 fcl files not in `dunetpc/fcl` `#include services_dune.fcl` -- big job to replace that with just the needed services, or move functionality up the tree. Don't want to have everything all put back in one repository, though.

dunetpc

[+](#)[Overview](#)[Activity](#)[Issues](#)[Gantt](#)[Calendar](#)[News](#)[Documents](#)[Wiki](#)[Files](#)[Repository](#)

root / fcl / common @
develop

[Statistics](#)

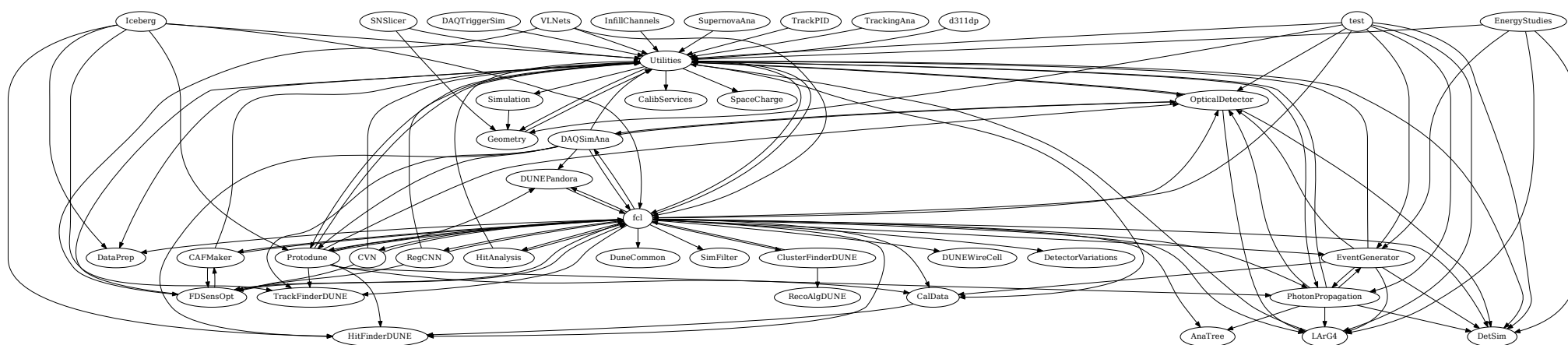
| Branch:

develop



Name	Size	Revision	Age	Author
CMakeLists.txt	237 Bytes	ab0ab08e	about 4 years	David Adams
SpacePointSolver_dune.fcl	450 Bytes	b2be7ddf	about 2 years	Tingjun Yang
backtrackerservice_dune.fcl	347 Bytes	be63b297	4 months	Tingjun Yang
calibration_dune.fcl	1.11 KB	87c6d425	about 2 months	Tingjun Yang
calorimetry_dune10kt.fcl	716 Bytes	30d2d912	almost 2 years	Tingjun Yang
calorimetry_dune35t.fcl	1017 Bytes	30d2d912	almost 2 years	Tingjun Yang
calorimetry_pdune.fcl	4.38 KB	87c6d425	about 2 months	Tingjun Yang
featurelabelingmodules.fcl	1.47 KB	62a277d4	4 months	Tingjun Yang
particleinventoryservice_dune.fcl	398 Bytes	78b6b149	almost 2 years	Tingjun Yang
time_memory_tracker_dune.fcl	272 Bytes	9a58e903	almost 2 years	Tingjun Yang
tools_dune.fcl	212 Bytes	ceaae40d	over 1 year	Vyacheslav Galymov

The FCL #include dependency graph



Another proposal for fcl files

- Put them where they already are
 - top-level fcl directory just goes in dunesw, or put duneconfig on the top.
 - Keep existing fcl files in their directories
- But ... This requires setting up the entire stack in order to be able to run many workflows.
- CI system already does this (to check). Entire stack is built before unit tests and integration tests are run.
- One advantage: fcl files should be in places you expect to find them. Less hunting around and less maintenance than if we split up services_dune and included just the necessary pieces.

Extras

Stoner Architect Drafts All-Foyer Mansion

MINNEAPOLIS—In the oft-overlooked field of stoner architecture, new talent often goes unnoticed. But that hasn't been the case for Minneapolis stoner architect Richard "Dick" Donovan, whose groundbreaking design for an all-foyer mansion is earning slack-jawed admiration from some of the most respected members of the Twin Cities stoner-architecture community.

