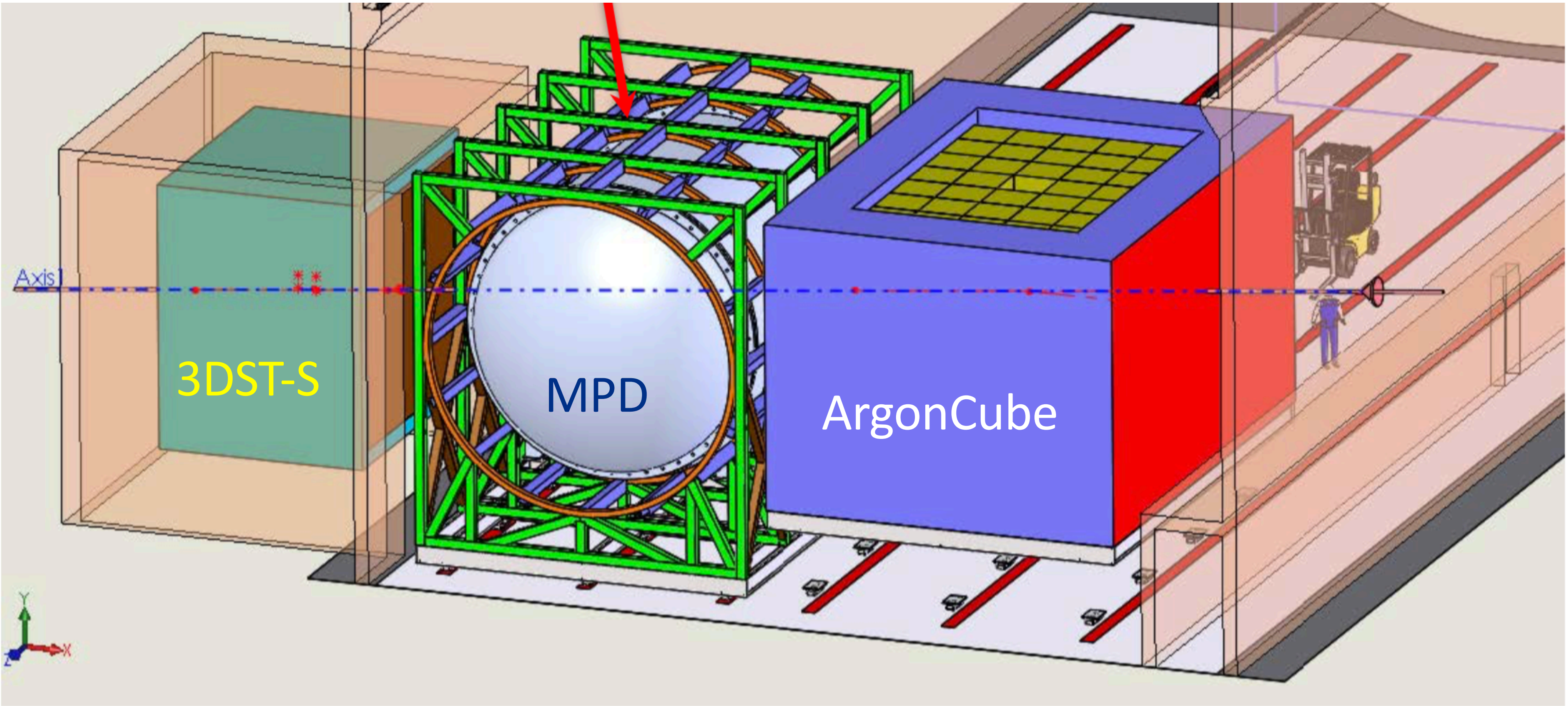
DUNE Perspective on Future LArSoft

Tom Junk

LArSoft Workshop

25 June 2019

The DUNE Near Detector Complex



ArgonCube: Pixel-based LArTPC, unmagnetized (150 Tons)

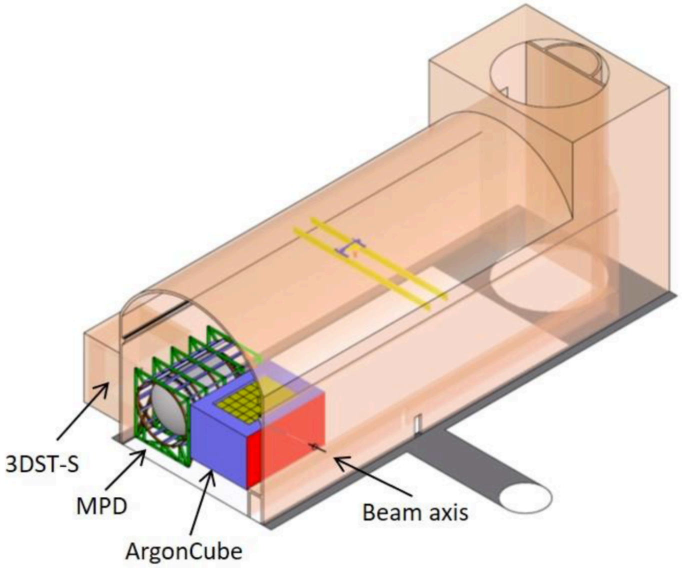
MPD: "Multi-Purpose Detector": High-Pressure Gas TPC, solenoid, ECAL, muon stack

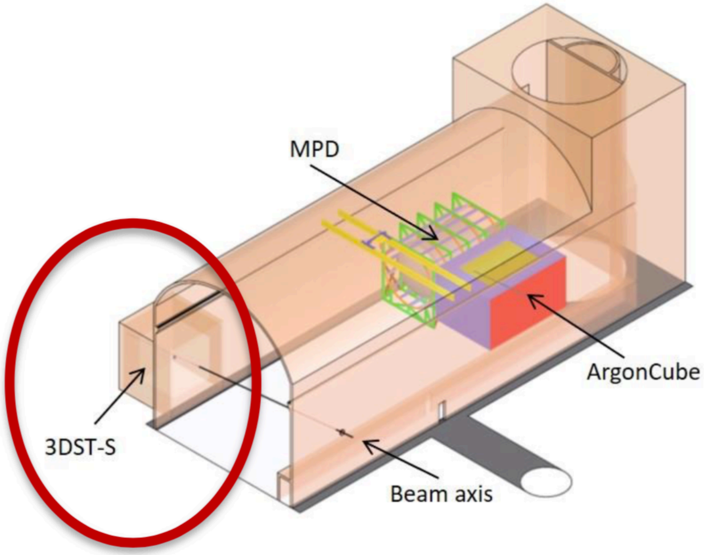
3DST-S Plastic scintillator, gas TPC, magnet, and ECAL

DUNE ND Prism Hall

MPD and ArgonCube plan to move up to 35m off axis.

3DST-S stays on axis in an alcove

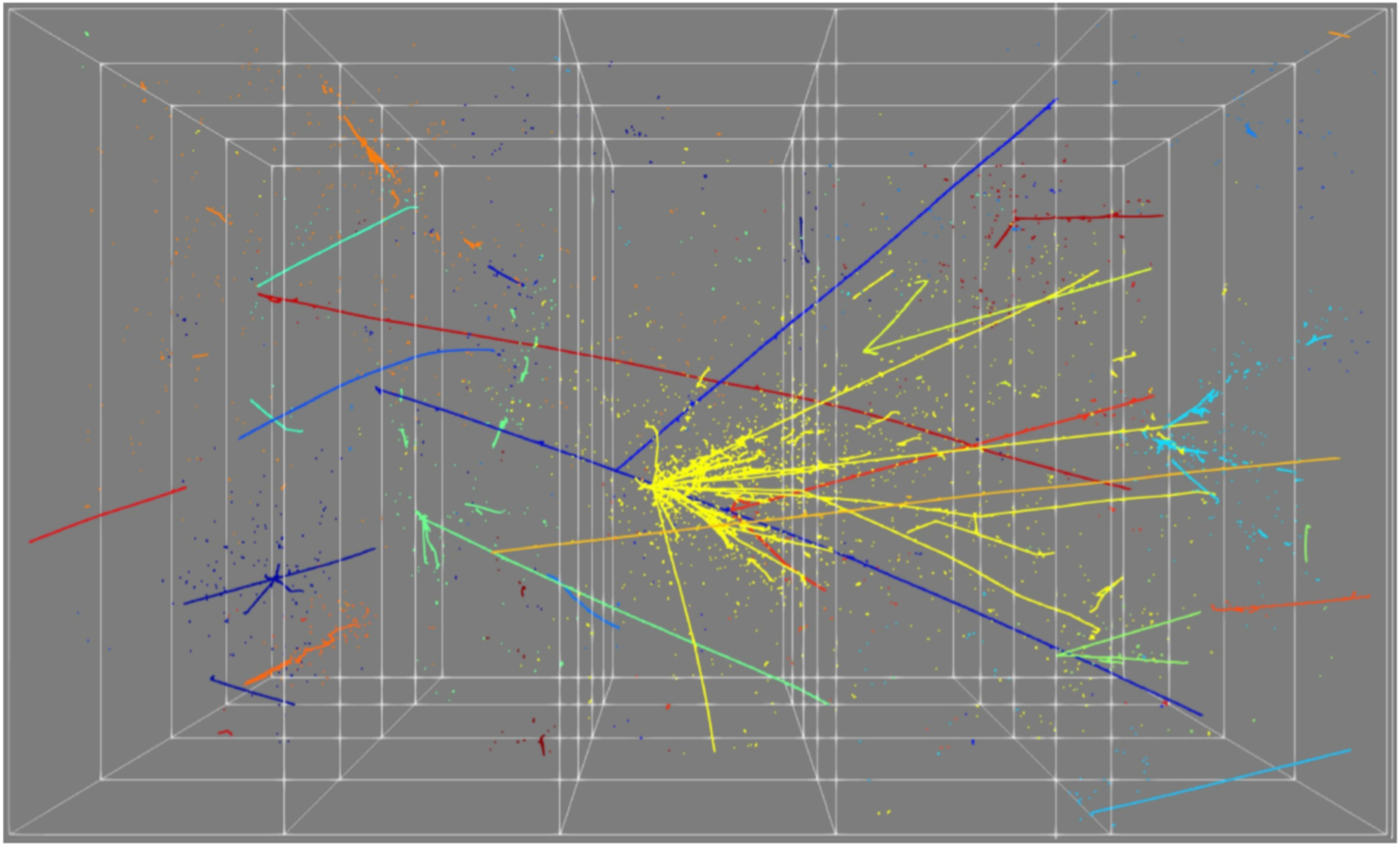




An Event in ArgonCube

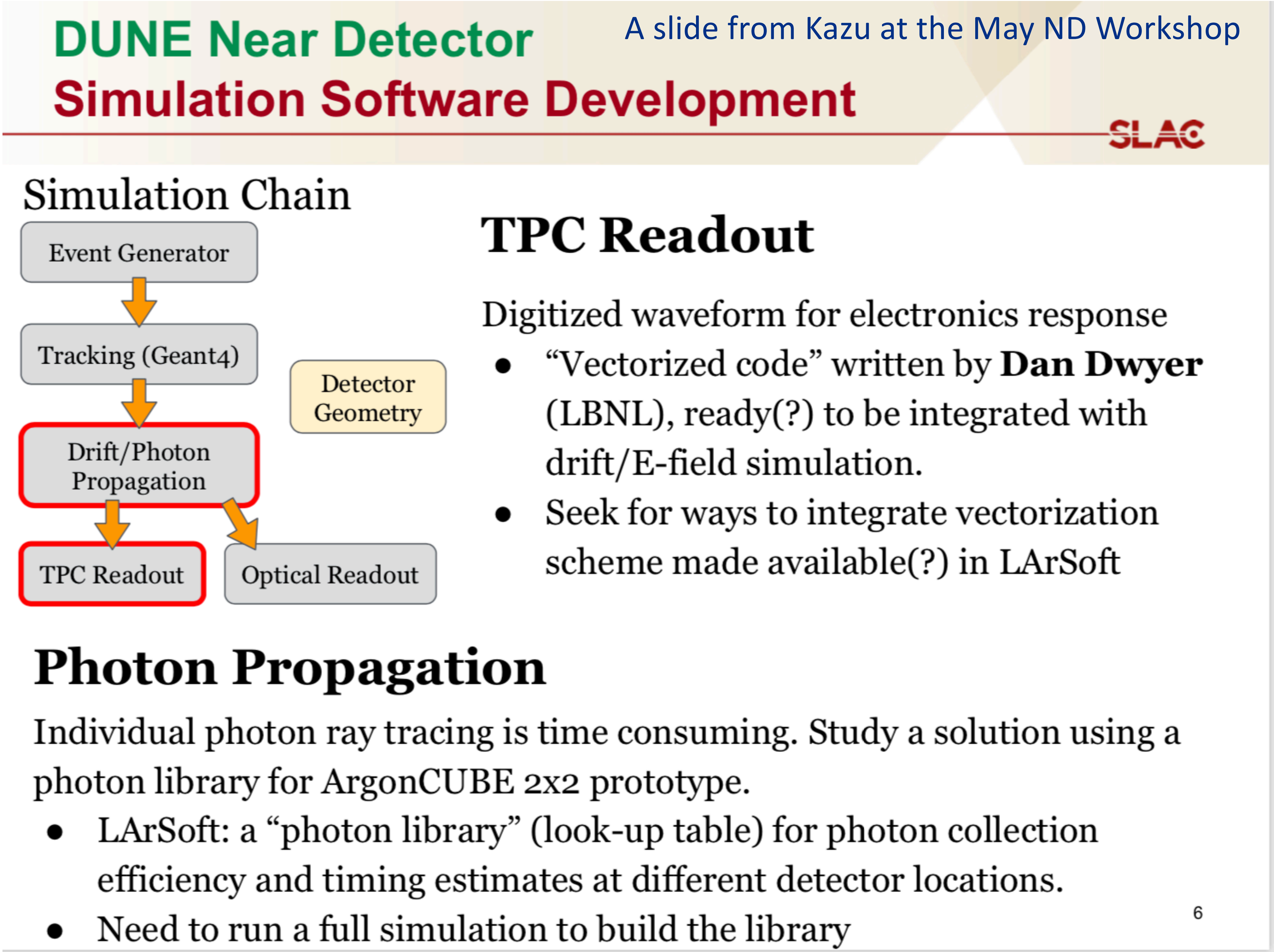
J. Sinclair

K. Terao
is the ArgonCube
software coordinator



1 MW 3 horn optimised spill, FHC, including rock. 4x5 geometry.
Colouring by nu.

A slide from Kazu at the May ND Workshop



DUNE Near Detector
Simulation Software Development

SLAC

Simulation Chain

Event Generator

Tracking (Geant4)

Detector Geometry

Drift/Photon Propagation

TPC Readout

Optical Readout

TPC Readout

Digitized waveform for electronics response

- "Vectorized code" written by **Dan Dwyer** (LBNL), ready(?) to be integrated with drift/E-field simulation.
- Seek for ways to integrate vectorization scheme made available(?) in LArSoft

Photon Propagation

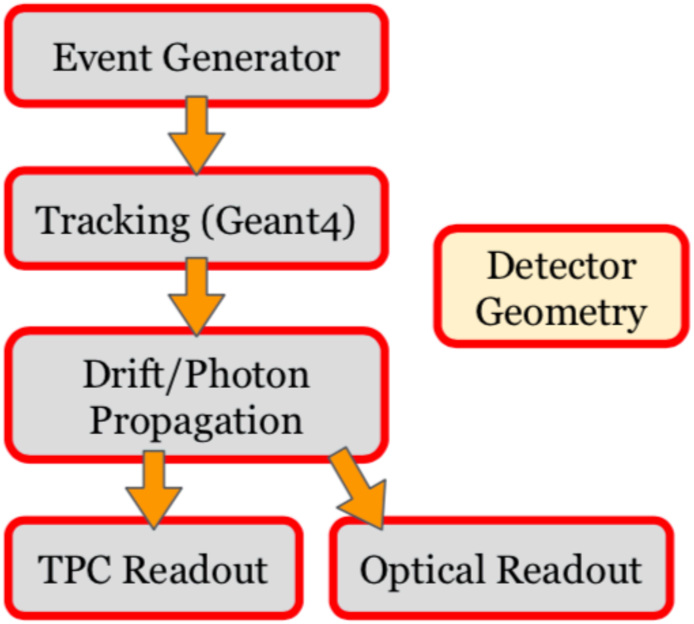
Individual photon ray tracing is time consuming. Study a solution using a photon library for ArgonCUBE 2x2 prototype.

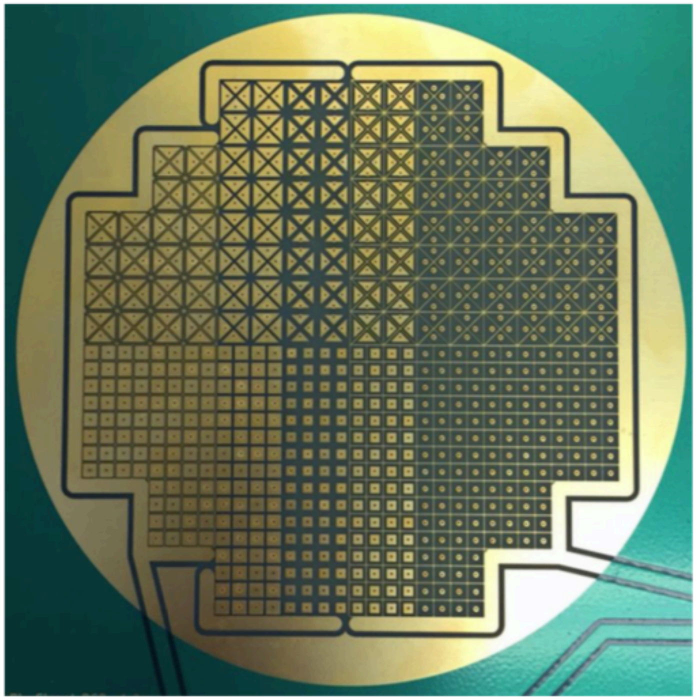
- LArSoft: a "photon library" (look-up table) for photon collection efficiency and timing estimates at different detector locations.
- Need to run a full simulation to build the library

A slide from Kazu at the May ND Workshop

DUNE Near Detector
Simulation Software Development

Simulation Chain



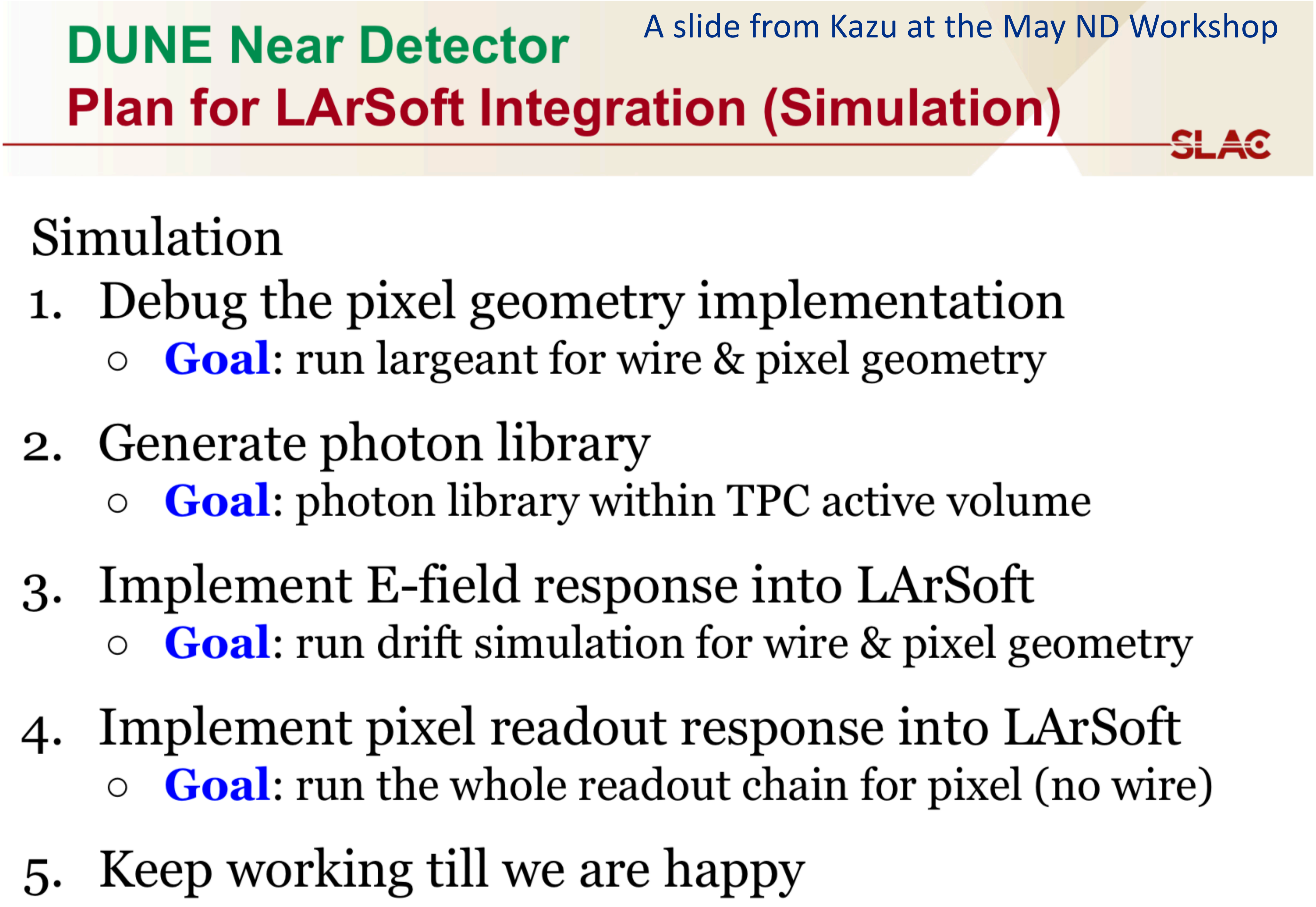


Pixel Geometry in LArSoft

LArSoft assumes “wire” in the core of Geometry design & APIs to query geometry information (PlaneGeo/WireGeo). Needs to be changed.

- Designed a generic “charge-sensitive element” to replace the current implementation in non-distruptive manner (by **Gianluca** @ SLAC)
- Now in testing stage: goal is to run largeant for wire & pixel geometry

A slide from Kazu at the May ND Workshop

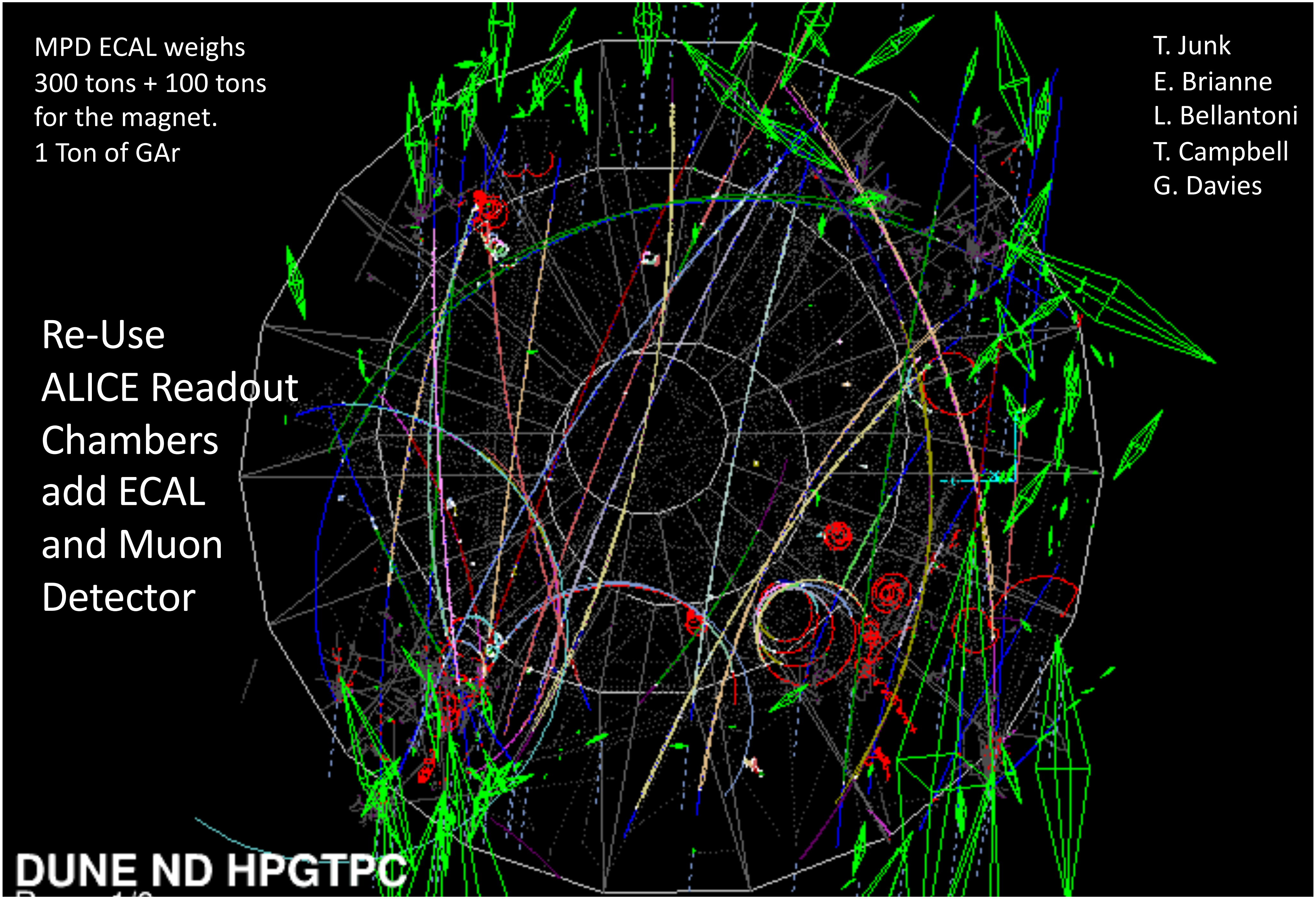


DUNE Near Detector
Plan for LArSoft Integration (Simulation)

SLAC

Simulation

1. Debug the pixel geometry implementation
   - **Goal**: run largeant for wire & pixel geometry
2. Generate photon library
   - **Goal**: photon library within TPC active volume
3. Implement E-field response into LArSoft
   - **Goal**: run drift simulation for wire & pixel geometry
4. Implement pixel readout response into LArSoft
   - **Goal**: run the whole readout chain for pixel (no wire)
5. Keep working till we are happy

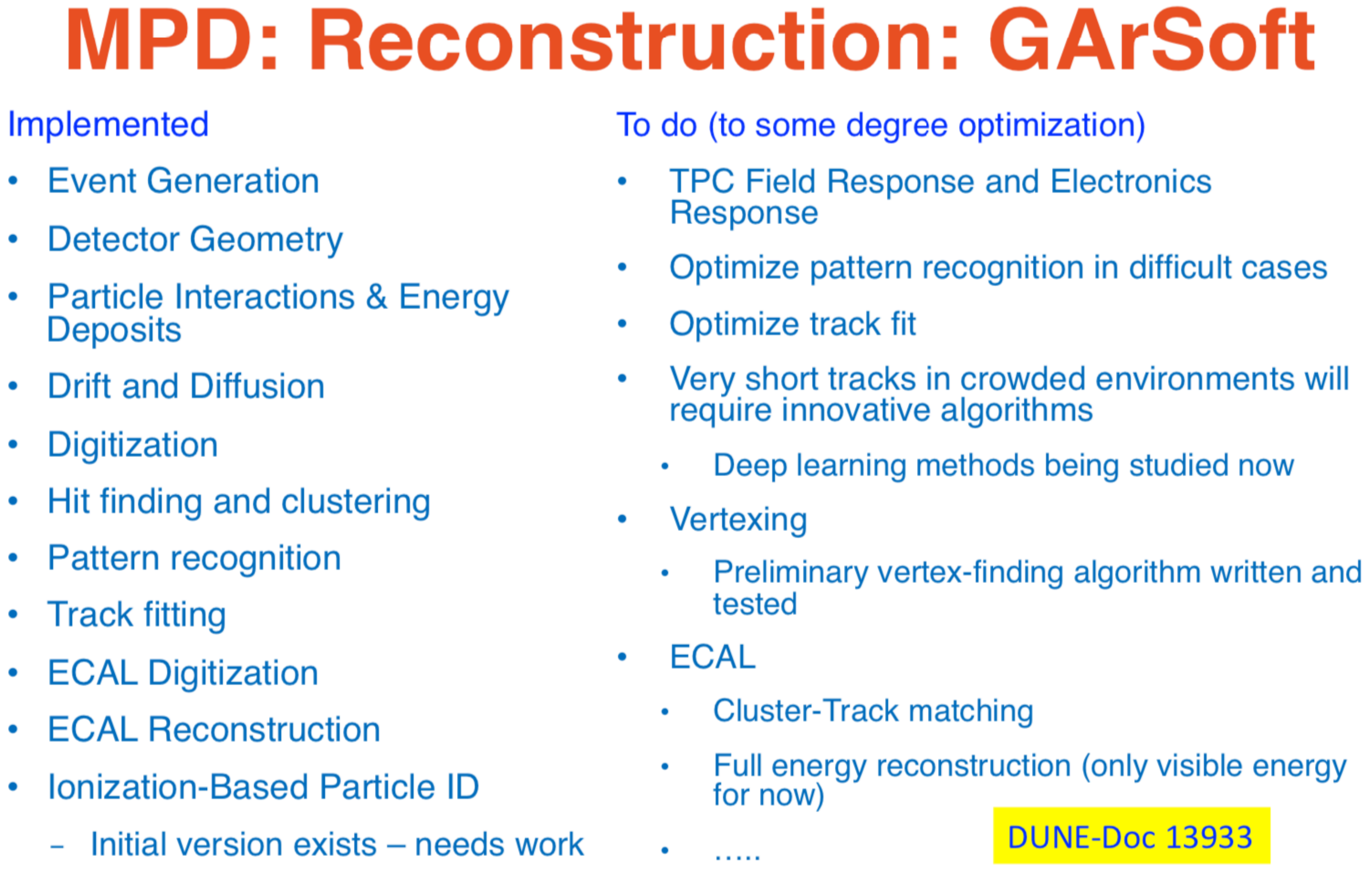


MPD ECAL weighs
300 tons + 100 tons
for the magnet.
1 Ton of GAr

T. Junk
E. Brianne
L. Bellantoni
T. Campbell
G. Davies

Re-Use
ALICE Readout
Chambers
add ECAL
and Muon
Detector

A slide from AlanBross at the June LBNC Review



MPD: Reconstruction: GArSoft

Implemented

- Event Generation
- Detector Geometry
- Particle Interactions & Energy Deposits
- Drift and Diffusion
- Digitization
- Hit finding and clustering
- Pattern recognition
- Track fitting
- ECAL Digitization
- ECAL Reconstruction
- Ionization-Based Particle ID
  - Initial version exists – needs work

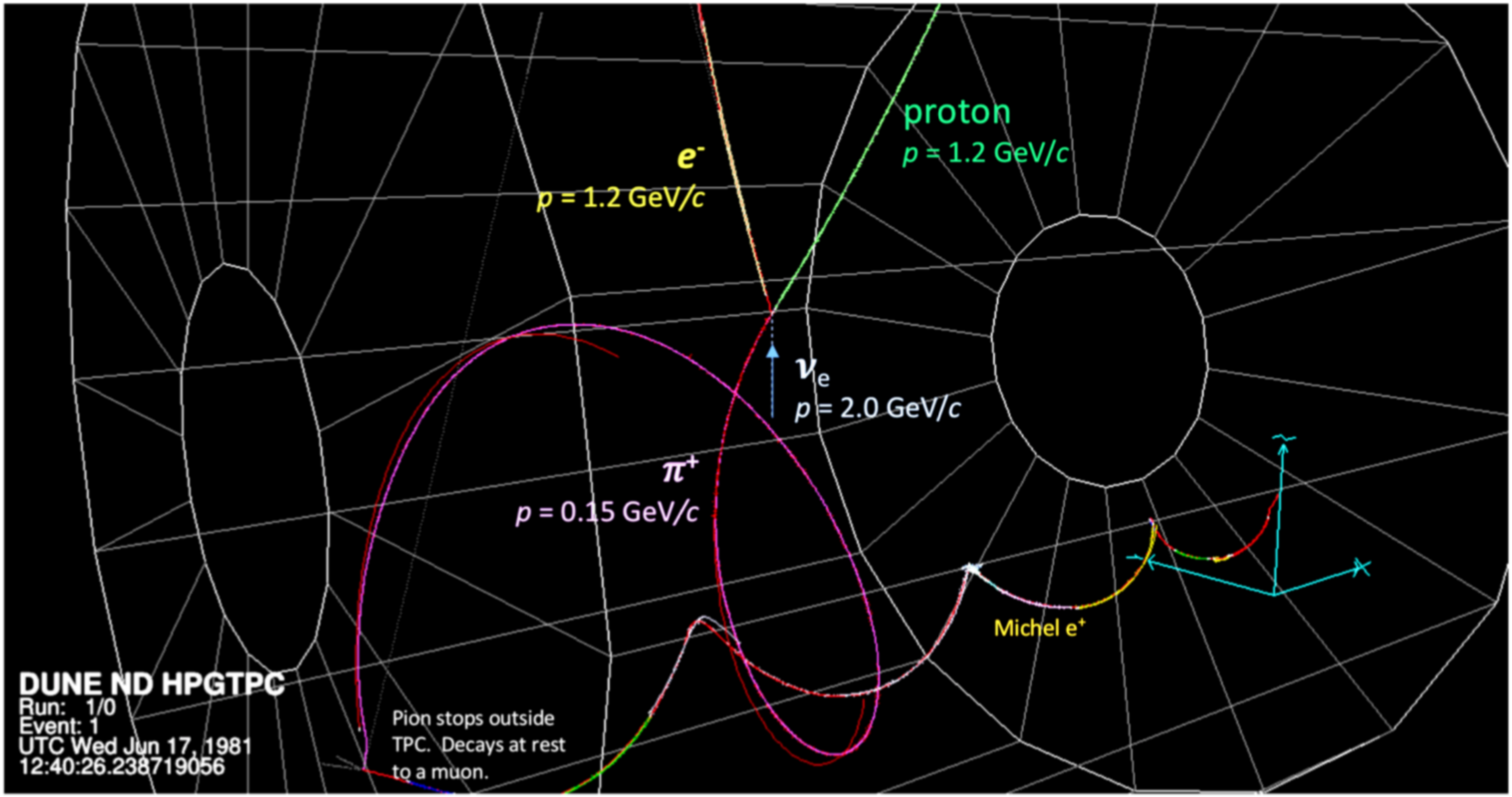
To do (to some degree optimization)

- TPC Field Response and Electronics Response
- Optimize pattern recognition in difficult cases
- Optimize track fit
- Very short tracks in crowded environments will require innovative algorithms
  - Deep learning methods being studied now
- Vertexing
  - Preliminary vertex-finding algorithm written and tested
- ECAL
  - Cluster-Track matching
  - Full energy reconstruction (only visible energy for now)
  - …..

DUNE-Doc 13933

A Simulated and Reconstructed ν_e Charged Current Event in the HPGTPC

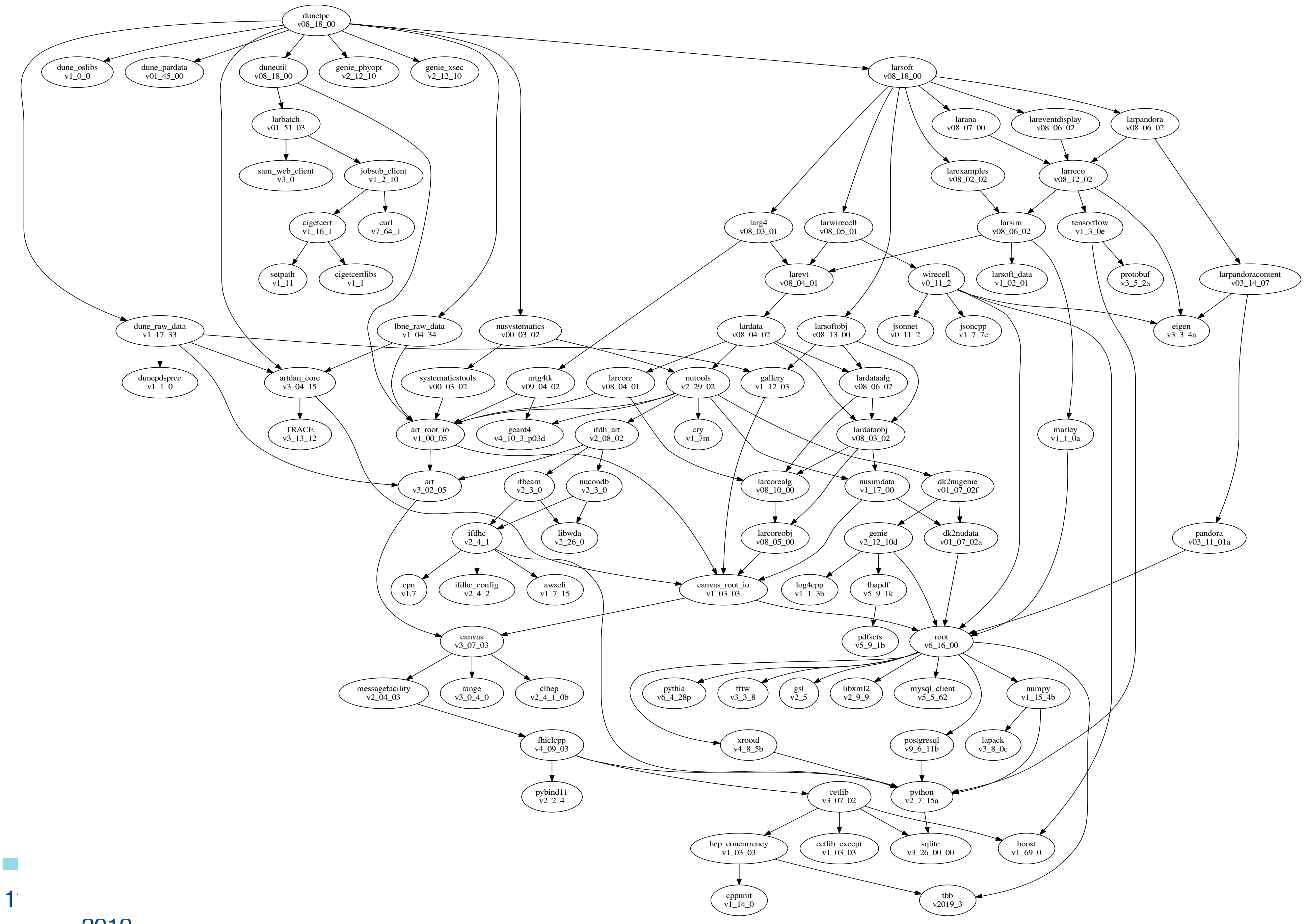
$\nu_e + \mathrm{Ar} \rightarrow \mathrm{e-} + \pi + + \mathrm{p} + \mathrm{n}$



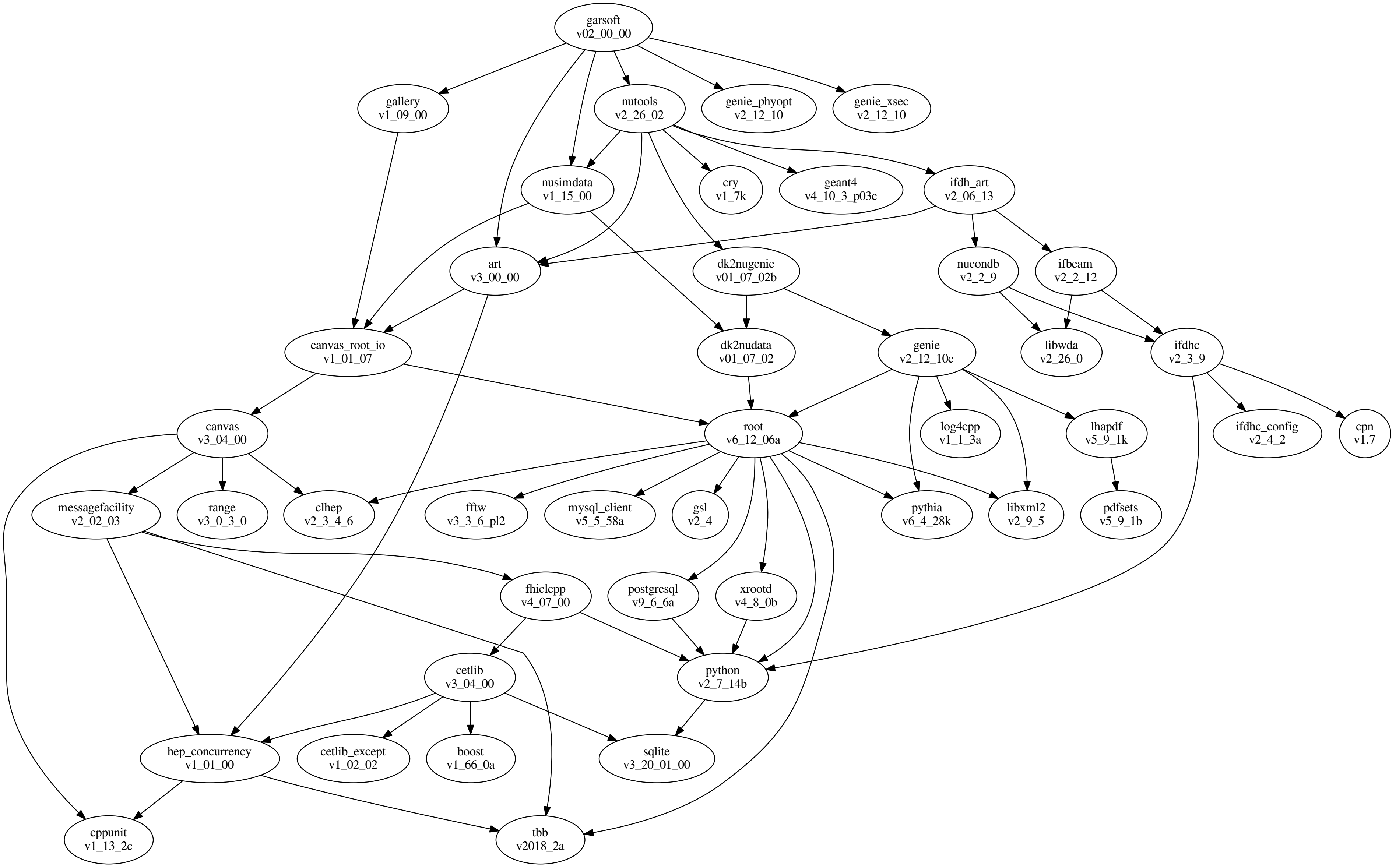
Pion stops outside TPC. Decays at rest to a muon.

Neutron with p = 0.23 GeV/c at the P.V. not shown

The dunetpc Dependency Tree (v08_18_00)



The GArSoft Dependency Tree (depends on art, nutools)



garsoft v02_00_00

gallery v1_09_00

nutools v2_26_02

genie_phyopt v2_12_10

genie_xsec v2_12_10

nusimdata v1_15_00

cry v1_7k

geant4 v4_10_3_p03c

ifdh_art v2_06_13

art v3_00_00

dk2nugenie v01_07_02b

nucondb v2_2_9

ifbeam v2_2_12

canvas_root_io v1_01_07

dk2nudata v01_07_02

genie v2_12_10c

libwda v2_26_0

ifdhc v2_3_9

canvas v3_04_00

root v6_12_06a

log4cpp v1_1_3a

lhapdf v5_9_1k

ifdhc_config v2_4_2

cpn v1.7

messagefacility v2_02_03

range v3_0_3_0

clhep v2_3_4_6

fftw v3_3_6_pl2

mysql_client v5_5_58a

gsl v2_4

pythia v6_4_28k

libxml2 v2_9_5

pdfsets v5_9_1b

fhiclcpp v4_07_00

postgresql v9_6_6a

xrootd v4_8_0b

cetlib v3_04_00

python v2_7_14b

hep_concurrency v1_01_00

cetlib_except v1_02_02

boost v1_66_0a

sqlite v3_20_01_00

cppunit v1_13_2c

tbb v2018_2a

Near Detector Integration Thoughts

- Running GArSoft and LArSoft modules in the same job "should" be possible
  - Both are based on the *art* framework
  - *art* loads modules dynamically based on FHiCL configuration
  - Data products for GArSoft have names in the gar namespace. e.g. gar::raw::RawDigit, so as not to collide or be confused with raw::RawDigit in LArSoft
- But there is some work to do to keep it all together
  - Dependency trees have to match. Must use same version of *art* for example. "A tree with two trunks"
  - GArSoft is updated to *art* V3. LArSoft has followed a few point releases since then but they involve few breaking changes.

Integration: Easy Issues First

Running detector-specific simulation and reconstruction are all independent pieces – modules work on independent data.

- channel response
- data output from sim job and readin in reco job
- noise filtering
- deconvolution
- TPC clusters and hit-finding
- tracking
- shower reco
- calorimetry

- Some modules and services may duplicate names with those in LArSoft. Can fix those easily.

Integration: Harder Issues

- Unified GEANT4 simulation
  - Current modules: LArG4 and GArG4. Consume MCTruth data products, make sim::SimChannel and energy deposits
  - particles produced in LAr -> GAr -- one can imagine running LArG4 first and then piping particles that come out of the LAr as MCTruth for GArG4, which gets run second.
  - Particles produced in GAr traveling back into LAr. Our CDR-Lite Executive summary mentions that backwards-going cosmic rays are an important calibration source for the LAr
    - Either need to iterate this, or run a unified GEANT4 step
- Unifying the GEANT4 step means having a single geometry description GDML file (or files), and calling GEANT4 once to follow particles back and forth.
- Hans Wenzel's new Energy Deposits in LArG4 look a lot like Brian Rebel's solution in GArG4.
- Data products have different names but that's okay

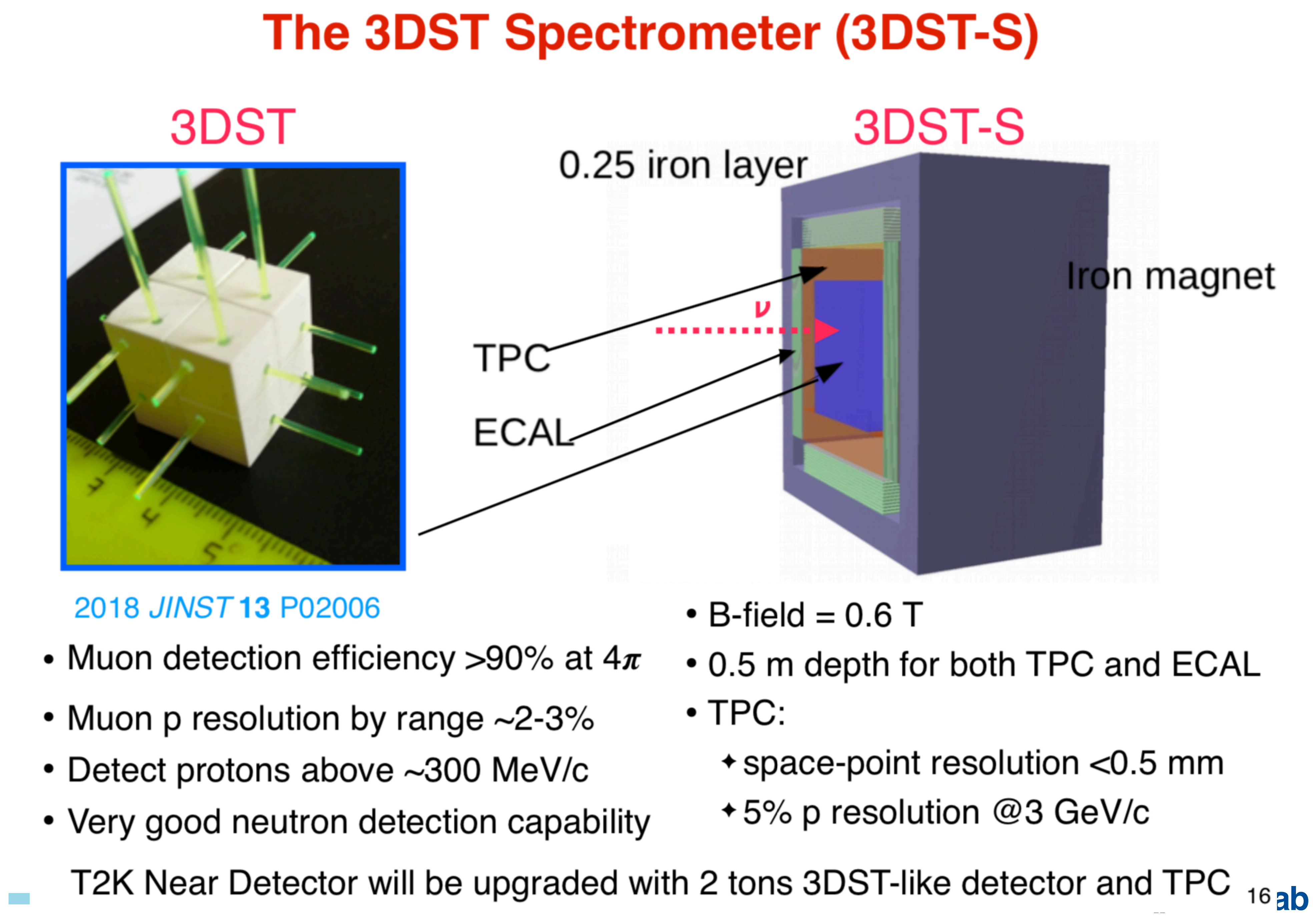
Integration: Event Display

- The three-detector ND Complex will have particles exiting one detector and possibly going into the other two.
- Visualizing the events will be useful in developing (traditional) reconstruction and track-matching algorithms
- Currently we are working independently
- How does MINERvA/MINOS deal with this?

Integration with 3DST-S

- Less understood on the MPD software side how the 3DST-S would fit in.
- GEANT4 step needs to be unified with ArgonCube and MPD for reasons explained before
- Off-axis positions are interesting – five combined geometry descriptions may be necessary, since 3DST-S does not move off axis.
- 3DST-S has gas TPC components. May want to re-use GArSoft algorithms, as they are intended to be homogeneous and isotropic. GArSoft assumes pixel readout however.

A slide from D. Sgalaberna at the DUNE ND Workshop May 2019



The 3DST Spectrometer (3DST-S)

3DST
3DST-S
0.25 iron layer
Iron magnet
ν
TPC
ECAL

2018 *JINST* **13** P02006

- Muon detection efficiency >90% at 4π
- Muon p resolution by range ~2-3%
- Detect protons above ~300 MeV/c
- Very good neutron detection capability

- B-field = 0.6 T
- 0.5 m depth for both TPC and ECAL
- TPC:
  - ✦ space-point resolution <0.5 mm
  - ✦ 5% p resolution @3 GeV/c

T2K Near Detector will be upgraded with 2 tons 3DST-like detector and TPC

Definition of a Near Detector "Event"

- *art* handles events as the smallest bit of independent data
- We associate these with triggered detector readouts.
- The entire complex will want to share a single trigger
  - beam spill signal from LBNF
  - Random triggers for background constraints
- We will also want to partition the DAQ for commissioning and tests

Supported Detectors in dunetpc

- 35-ton: Support is thin. Pandora stopped supporting 35-ton about a year ago. lbne_raw_data (DAQ interface) should be removed at some point. Data preservation?
- DUNE FD SP 10 kt
- DUNE FD SP 1x2x6 Workspace
- DUNE FD DP 10 kt
- ProtoDUNE-SP 6x6x6 meters cubed (+DAQ)
- ProtoDUNE-DP 6x6x6 meters cubed
- WA105 3x1x1 dual-phase prototype (+DAQ)
- ICEBERG (+DAQ)
- Coming: (?) ArgonCube ND. 2x2 ArgonCube Prototype in the NuMI hall near MINOS ND. (+DAQ)

Timing of a 7.5 ms ProtoDUNE-SP Reco Job

```
/pnfs/dune/scratch/dunepro/beam_prep/logs/protodune-sp_sce_sample_keepup_7.5ms_v07_08_00_05_snapshot_id_192222_slice_0_stage_500_17066634_0.out
```

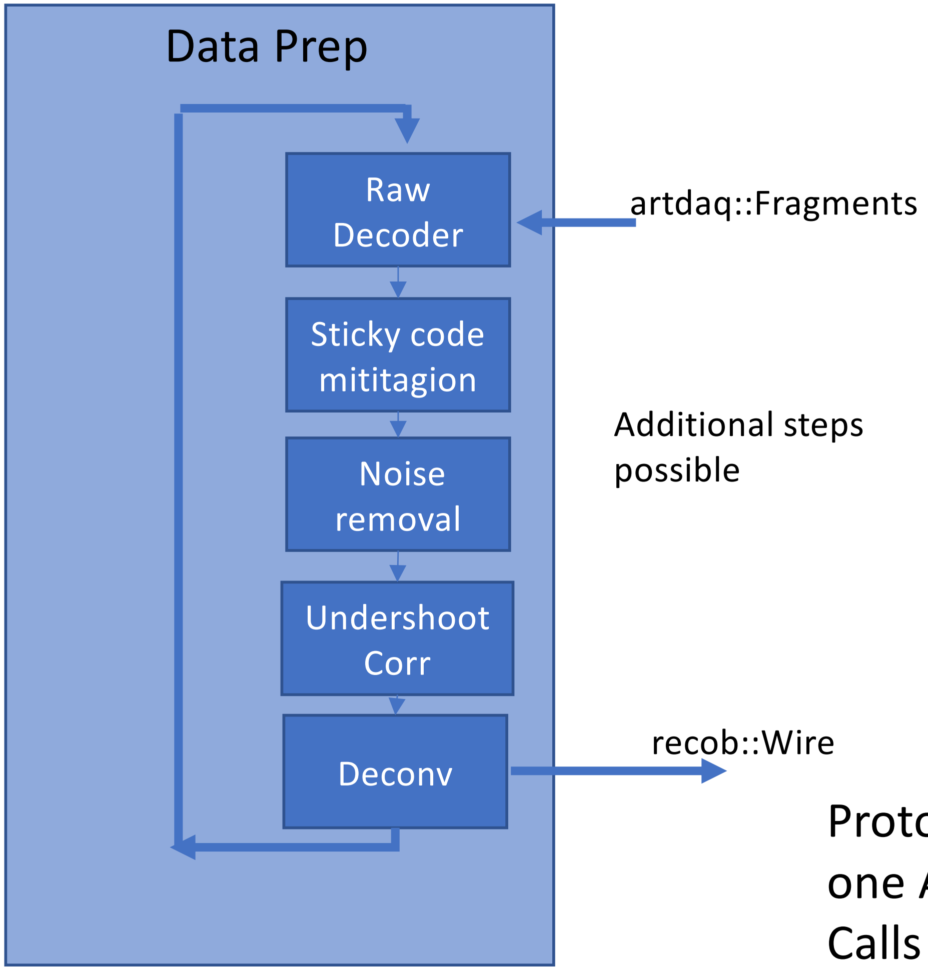
```
====================================================================================================================================
TimeTracker printout (sec)                                    Min           Avg           Max          Median         RMS         nEvts
====================================================================================================================================
Full event                                                  1402.6        2155.65       3403.72       2100.68       435.928        49
------------------------------------------------------------------------------------------------------------------------------------
source:RootInput(read)                                    0.000543632   0.00688882    0.111605      0.00189346    0.0191799      49
decode:timingrawdecoder:TimingRawDecoder                  0.00046275     0.0137007    0.178662      0.0115812     0.0252344      49
decode:ssprawdecoder:SSPRawDecoder                         0.233429      0.324645     0.806665      0.314574      0.0825951      49
decode:tpcrawdecoder:PDSPTPCRawDecoder                     24.8511       29.8284      33.0416       30.0935       2.21615        49
decode:ctbrawdecoder:PDSPCTBRawDecoder                    0.000214046   0.00367091    0.0828906     0.000400356   0.0124469      49
decode:beamevent:BeamEvent                                0.000726402   0.00678737    0.257794      0.00117761    0.036297       49
decode:caldata:DataPrepModule                              75.3518       104.001       135.345       103.056       12.6566        49
decode:gaushit:GausHitFinder                               34.2085       52.2964       70.2773       52.8011       7.57757        49
decode:reco3d:SpacePointSolver                             9.88494       30.8605       70.2438       25.6227       15.5368        49
decode:hitpdune:DisambigFromSpacePoints                    50.9646       101.859       156.355       99.9169       25.2145        49
decode:linecluster:LineCluster                             11.6407       22.6864       39.0839       22.2573       5.34067        49
decode:pandora:StandardPandora                             744.267       1274.57       2191.16       1228.64       306.203        49
decode:pandoraTrack:LArPandoraTrackCreation                57.3564       115.366       171.417       112.522       25.1478        49
decode:pandoraShower:LArPandoraShowerCreation              45.2669       92.431        152.72        91.4082       23.764         49
decode:pandoracalo:Calorimetry                             13.9568       29.1075       44.7453       28.942        7.08136        49
decode:pandorapid:Chi2ParticleID                          0.0072007      0.0137094    0.0312194     0.0134894     0.0043193      49
decode:pmtrack:PMAlgTrackMaker                             132.728       252.038       397.34        249.126       54.3535        49
decode:pmtrackcalo:Calorimetry                             11.1801       23.2617       44.6746       22.2582       6.54617        49
decode:pmtrackpid:Chi2ParticleID                          0.00731181     0.013917     0.0232595     0.0145476    0.00380676      49
decode:ophitInternal:OpHitFinder                          0.00872905     0.0135512    0.0184147     0.0138113    0.00228544      49
decode:ophitExternal:OpHitFinder                          0.00328762    0.00421628   0.00570462    0.00440462   0.000656042      49
decode:opflashInternal:OpFlashFinder                      0.00787199     0.0137992    0.0221914     0.0134439    0.0034087       49
decode:opflashExternal:OpFlashFinder                      0.000599987   0.000954489  0.00133126    0.000931951  0.000201029      49
decode:TriggerResults:TriggerResultInserter               2.3567e-05    3.7177e-05   7.0934e-05    3.7005e-05   1.15572e-05      49
end_path:out1:RootOutput                                   6.515e-06    9.1208e-06   2.4274e-05    8.068e-06    3.28833e-06      49
end_path:out1:RootOutput(write)                            21.5251       26.9227       45.9711       26.1481       4.15128        49
====================================================================================================================================
====================================================================================================
```

Scaling Resources to the Far Detector

- Pandora needs the most CPU in the 7.5 ms readout window ProtoDUNE-SP event
- But its CPU scales nonlinearly with the activity in the event
- Far Detector data will mostly be empty
- Need to run Data Prep on a 25—bigger detector
- Data unpacking and preparation scale linearly with data size (= detector size x nticks) FFTs scale a bit faster with nticks, but nchannels is the big scale factor here.

- Supernova burst: 30 seconds or more of non-zero-suppressed waveform readout
- All four detector modules processed separately (10 kt each)

Chunked TPC Wire Data Processing Chain, Option #1, Single Threaded



Additional steps possible

Loop over

- APAs, or
- DAQ chunks

Free up memory from artdaq::Fragments and raw::RawDigits inside loop. Storage of these is temporary

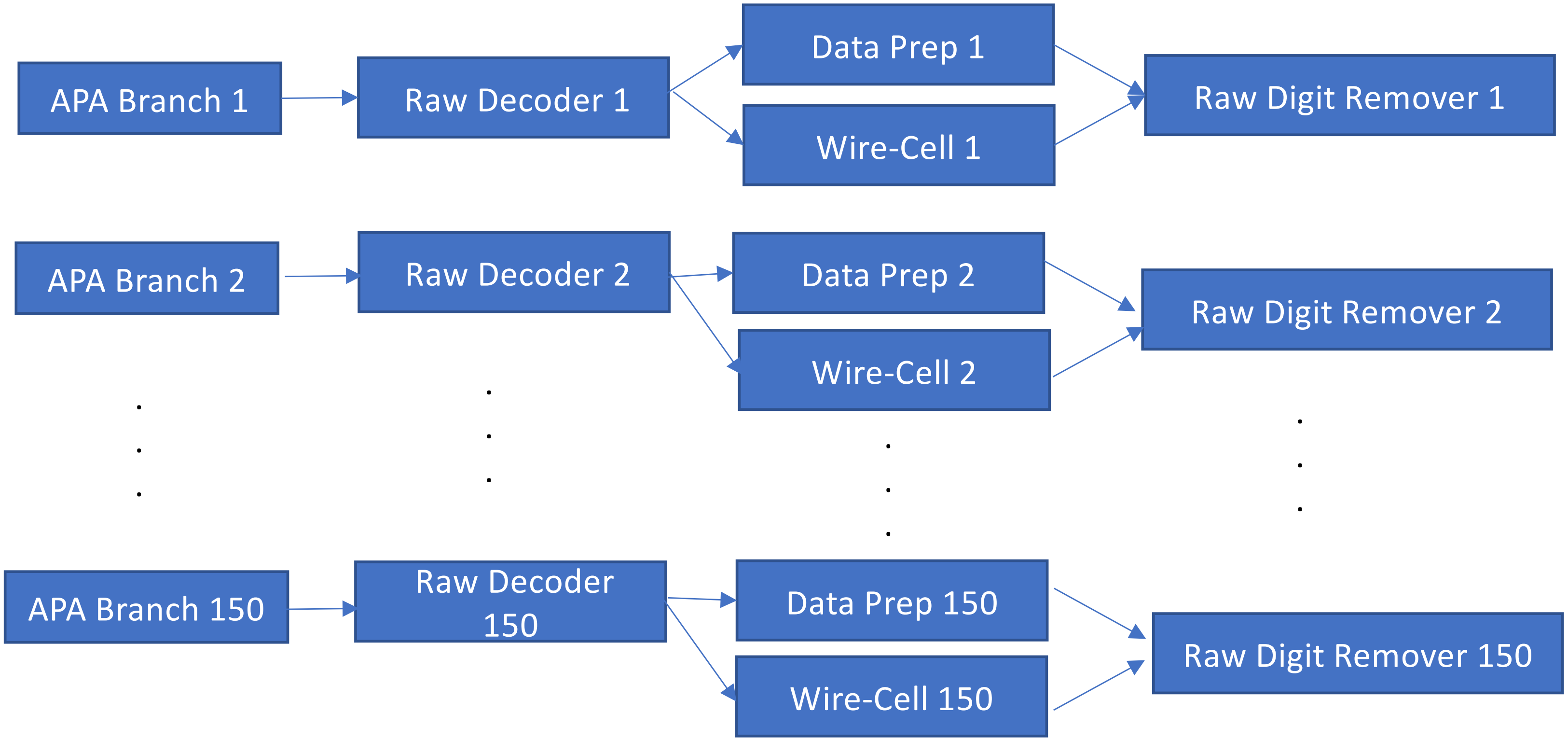
Is recob::Wire small enough to store the FD module's data in an event?

If not, then two options

- stream outputs to nAPA files like inputs, or
- include any processing needing recob::Wire in the loop

ProtoDUNE-SP Raw Decoder tool that unpacks one APA at a time now implemented.
Calls removeCachedProduct for the input artdaq::Fragments
Need to work on MC – break it into smaller pieces

Chunked TPC Wire Data Processing Chain, Option #2, Threadable Module Instances



Data Prep and Wire Cell may have to be serialized so that we are sure that everyone who needs raw digits for APA n is done and the raw digit remover can run.
ART does not currently support removing produced cached products however, only ones input from files.
-- may be an easy upgrade to allow produced cached products.

Threading processing of pieces of events is better than requiring multiple events to be in memory at a time.
Serial processing of APA's may be as efficient as parallel, perhaps more so.

Intra-event threading requires shared modules (replicated modules in art3 are made per schedule)

We care more about throughput than latency, though the Event Display has latency issues.

New DAQ Format Ideas

- DUNE DAQ Consortium is exploring ideas to not use *art*-formatted rootfiles as output from *art*DAQ.
- It is difficult to reshape *art* events after they are written. Possible, but difficult (LArIAT and 35t do this)
- It is difficult in *art* to process less than an event.
- It is impossible in ROOT to read in less than one entry on a branch (or leaf). (solution: just make more branches).
- Another solution: a file per APA – 150 Files per event.
  - The filesystem becomes part of the event builder.
- We'd like to keep them together. tar or something like it can keep the files on the same tape.

External Source Code (e.g. GPL3)

- Question on the DUNE Slack #larsoft-beginners channel:

What is the collaboration/LArSOFT/Art's position on copying from open source libraries? Specifically, I want to use a function released under LGPL 3.0 that I have modified, but left all the original inline documentation and author information, as well as my name and modification date

- Jeremy Hewes's request for central management of HighFive, a header-only convenience interface to HDF5. Lynn says DUNE would have to maintain it.

External Source Code Concerns

- What-if:
  - Original developers abandon project. We're left maintaining it
  - Developers take project in a new and interesting direction, leaving us behind or incompatible
- Concern if we modify external source
  - what if someone wants original, unmodified behavior
  - new names
- If code breaks, we can decide to maintain it or disable it. Simple updates are okay. Breakage is harder
- Tutorials?
- There's a lot of open-source code out there. Do we have to maintain every piece a DUNE collaborator wants to use?

GPU-Enabled Code

- We would like to make better use of GPU's of course
  - machine learning training (already did. Robert and Dorota's track/shower discriminator was trained using GPU's)
  - machine learning discriminant calculation (? less CPU intensive than training presumably)
  - Data prep
  - Event Display
- Development platform and examples would be welcome.
  - gpvms?
  - desktops and laptops?
  - Wilson Cluster?

Operating System Support

- We support
  - SL6
  - SL7
  - macOS 10.13 "High Sierra"
  - macOS 10.14 "Mojave"
- Many people currently use SL7. We just got a new build node, dunebuild02.fnal.gov, which runs SLF7.
- Probably not too disruptive to move away from SL6, though we do have a TDR to finish writing.
- I am also okay with containers replacing flavor support. Supporting one flavor and providing containers sounds good to me.

dunetpc Is Getting Too Big

- Slow git clone. Several minutes, even at FNAL. Lots of old history (GDML files came and went).
- build takes ~1 hour of CPU and around 3 GB of storage. On a build node, it's not so bad.
- Split into repositories and UPS products?
- Need a different build and release strategy (a la LArSoft and MicroBooNE)
- dunepdsprce source tarball? Not currently. We download from github
- We had been waiting to see how Spack(Dev) affects this model.
- At the LArSoft coordination meeting June 4, we found that Spack(Dev) will target replacing mrb and ups – we still need to build a repository at a time.
- Will need some refactoring of code to break this up.

- Pull requests a la LArSoft with GitHub as the service provider

Extras