Fermilab U.S. DEPARTMENT OF ENERGY Office of Science

Liquid Argon Software Toolkit
LArSoft

Ruth Pordes, Erica Snider

For the LArSoft Collaboration – contributed to by many

http://www.larsoft.org

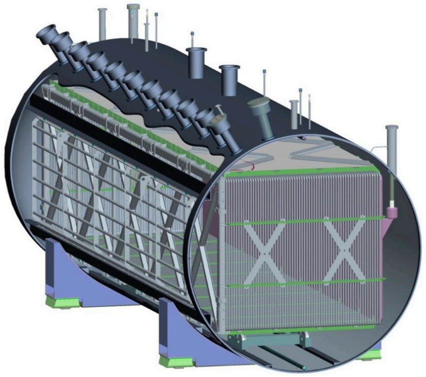
August 2016

Contributors to the talk : Jonathan Asaadi, Vito DiBentto, Lynn Garren, Jim Kowalkowski, Marc Paterno, Brian Rebel, Gianluca Petrillo, Saba Sehrish…

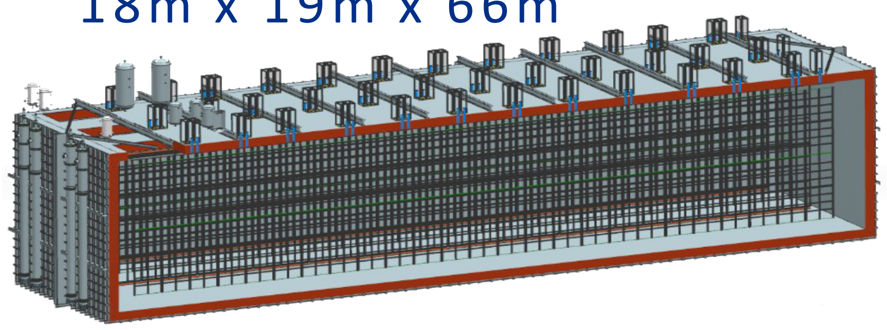
LArSoft

- C++ based infrastructure and algorithms for the reconstruction, simulation and analysis of data for and from Liquid Argon Time Projection Chambers
- Aim is more (as complete as feasible) automated reconstruction of LArTPC data.
- Includes one or multiple algorithms for signal processing, hit finding, cluster finding, showers, track finding, vertex finding, particle identification, deconvolution…

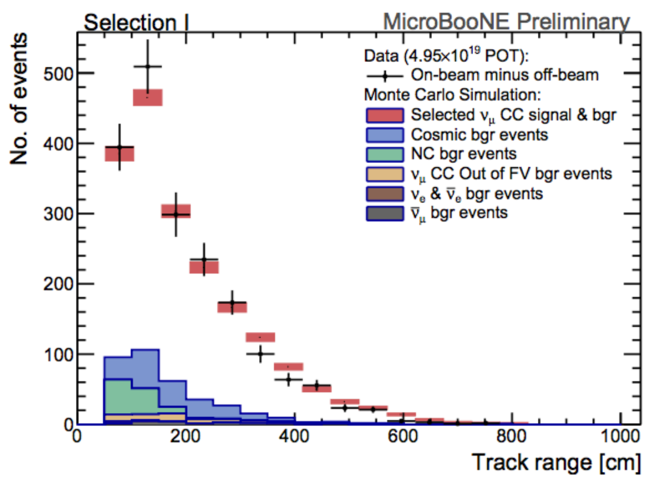
MicroBooNE LArTPC:
2.2m x 2.5m x 10m



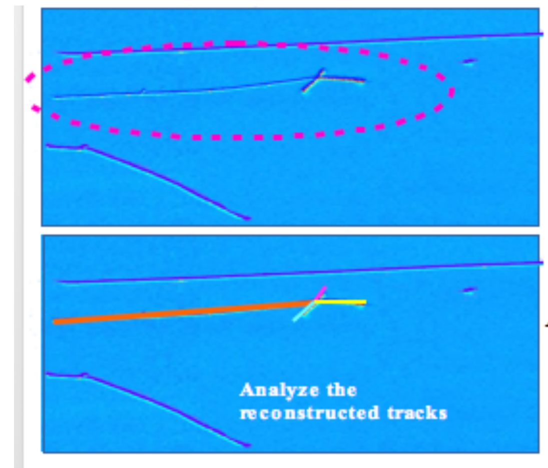
One DUNE LArTPC Module:
18m x 19m x 66m



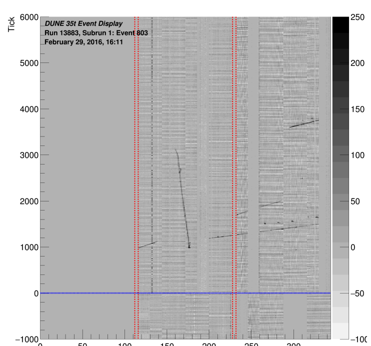
Science output using LArSoft



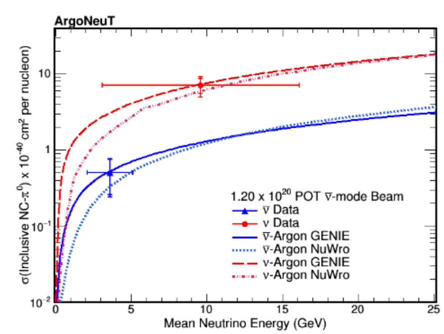
Courtesy MicroBooNE collaboration http://www-microboone.fnal.gov/publications/publicnotes/MICROBOONE-NOTE-1010-PUB.pdf,



Courtesy LArIAT Collaboration π - Ar Event Selection, FNAL Wine and Cheese Seminar



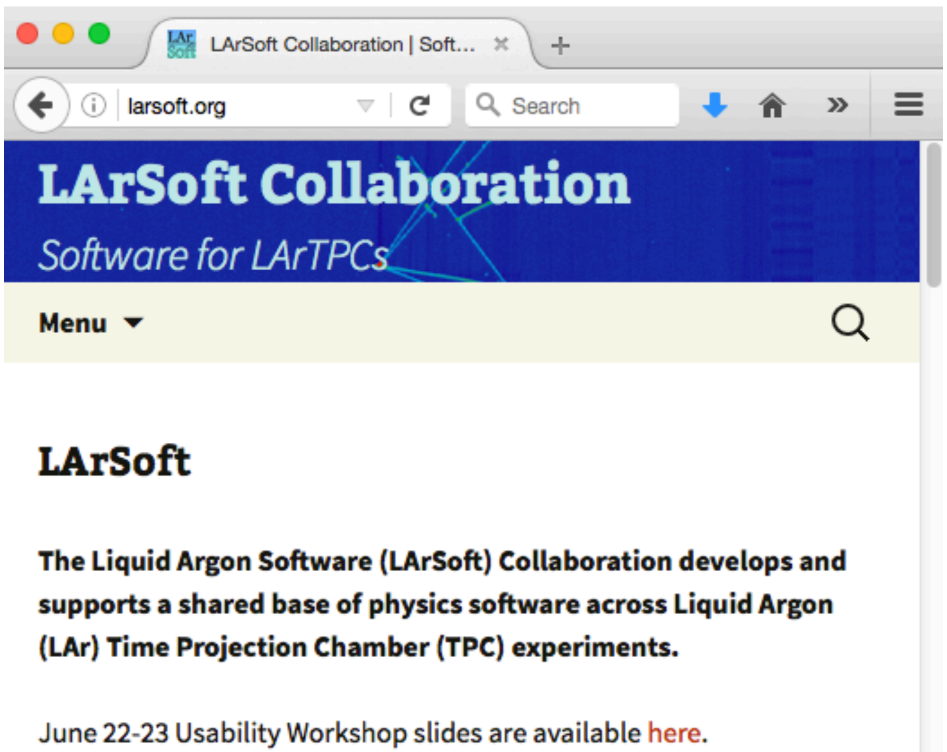
Courtesy DUNE Collaboration http://lbne-dqm.fnal.gov/ArchiveEventDisplay/ArchiveEVD_xaa.html



Courtesy ArgoNeuT Collaboration: http://arxiv.org/pdf/1511.00941.pdf

The LArSoft Collaboration is:

- Experiments, Labs and University groups who contribute to and use the LArSoft software
- The set of projects that contribute to the LArSoft executables used for processing data.



The LArSoft Project

- Means to share expertise and software across experiments.
- Provisioning and support for the core framework, architecture, design, release, testing and roadmap activities across the experiments.
- Provide “crowd source” “open source” value including:
  - Increase quality and effectiveness of algorithm code,
  - Provide clean integration with other products,
  - Reduce total effort needed across the experiments,
  - Support of new ideas/proposals who can build out from existing capabilities.
- One of Fermilab’s centralized activities towards common software and computing services across experiments (synergistic with P5 report guidance)

Scope of this talk

The framework, structure and project.

(Does not include science, algorithms, physics inputs and outputs.)

Outline

Background

Architecture

Code

Future Plans

Background: History

- 2008: First code repository by Brian Rebel to share code for LArTPCs.
- 2010: Eric Church joined common LArSoft effort; both scientists members of ArgoNeuT and MicroBooNE.
- 2013: Fermilab Scientific Computing Division took on coordination, sustainability, support for build, release and maintenance.
- 2014: Collaboration driven by experiment spokesperson steering group defining the roadmap and priorities of the collaboration and future work.

Background: Requirements

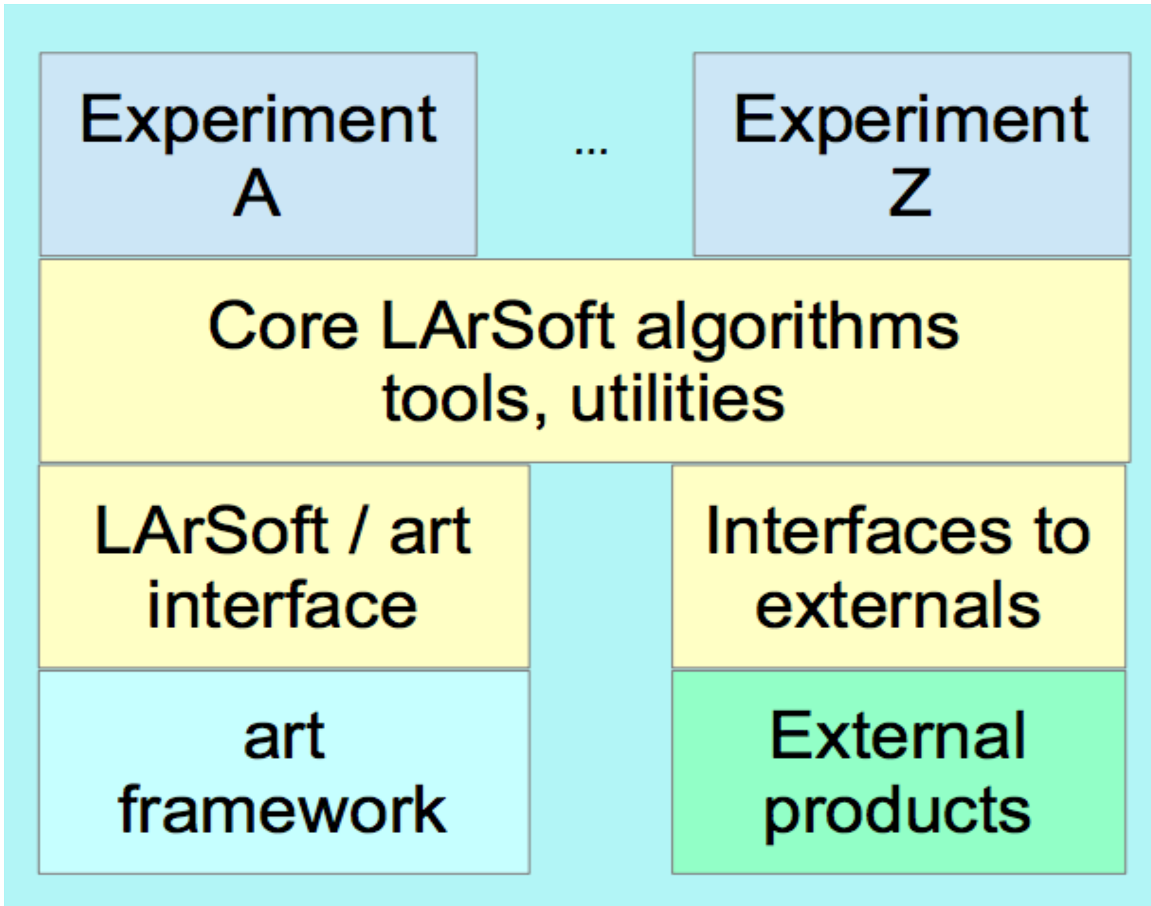
- 2015: LArTPC Reconstruction workshops
  - delivered requirements document
  - > 40 Authors

2

Open Architecture

- The LArSoft software is based on the HEP *art* event processing framework, used by and supported for most Fermilab based experiments. *art* includes facilities to:
  - define a variety of experiment-written modules that perform the steps in a workflow
  - configure the coordinated execution of these modules
  - handling experiment-defined descriptions of experimental data
  - read and write files containing these data
  - keeping track of the provenance of data generated during execution of the program

Layered Architecture

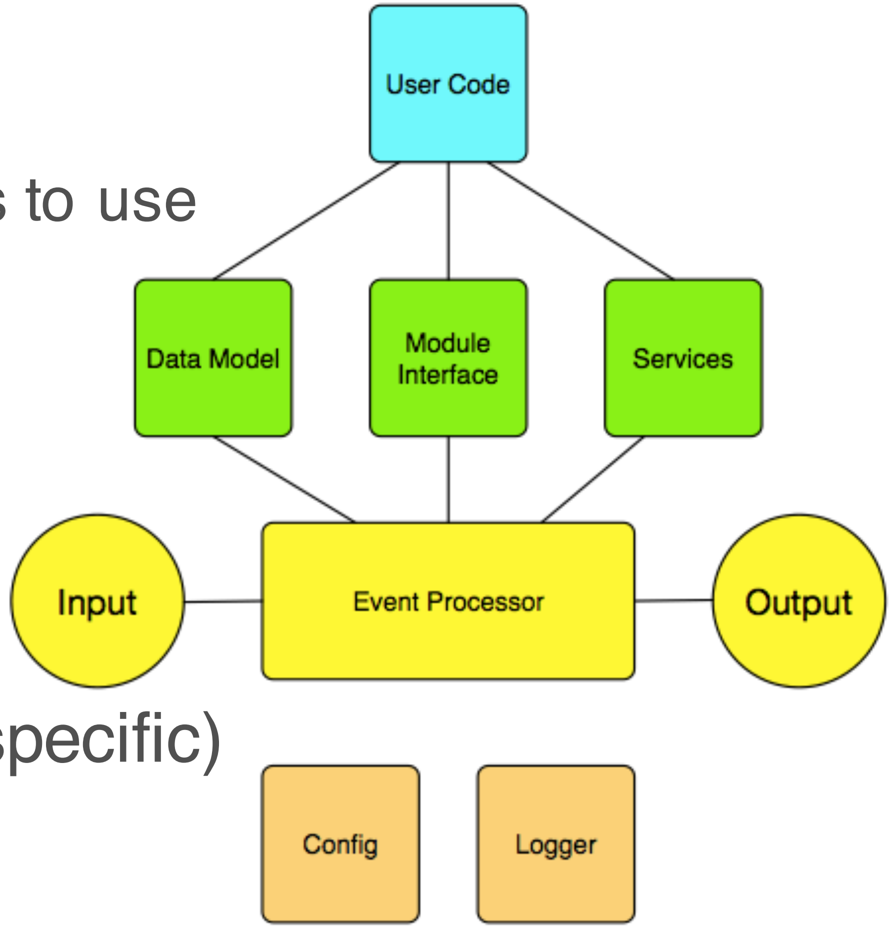


Data Products

- Classes that can be saved into *art* ROOT output files.
- Communication protocol between modules.
- Translations between this and external software packages protocols provide for data exchange and module integration/interaction.
- Core data products cover simulation, detector output, reconstruction and analysis information.
- Users, experiments, external providers, define extensions that can be shared through contributing to core LArSoft.
- Connections between data products are defined/used through associations.

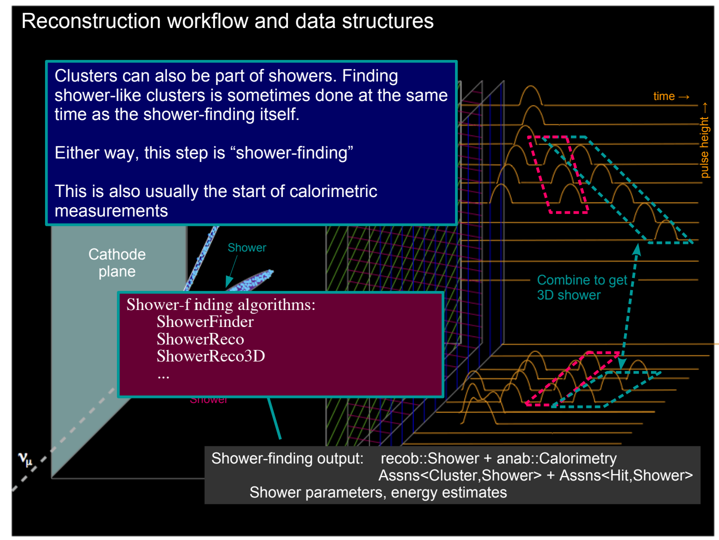
Services

- Provide common resources or tools available to all modules
  - Manage the resource
  - Allow modules and other services to use the resource.
- *art* services include:
  - Random number generator,
  - memory tracker,
  - message logger etc.
- LArSoft (shared and experiment specific) services for:
  - geometry,
  - conditions,
  - databases etc.



Modules, Algorithms, Workflows

- Modules include the algorithms
- A module "plugs into" a processing stream and performs a specific task on data obtained through the data products, independent of other running modules.
- Well-specified algorithm interfaces allow different algorithms to address any particular step/scope.
- Configuration files define and manage the workflow, execution sequence of the modules, experiment specific parameters etc.



Showing a module activity

Interfaces to External Software Products

- LArSoft Core modules provide centralized common data objects, physics utilities, and shared algorithms.
- APIs and data products provide interfaces to external software packages provided by other projects:
  - including Pandora software for pattern recognition, Geant4 simulation, Genie neutrino monte-carlo, and LArLite light analysis framework.
- Experiment specific algorithm implementations rely on the common modules and are moved into the common repository as they are shared.

Detector interoperability

- Important design objective for the toolkit/code suite.
- Drives guidelines for using and developing services and coding algorithms
- Encourage developers to define (and use!!) common interfaces for accessing detector-specific configuration information e.g. detector geometry
- Also avoid implied geometrical assumptions in algorithms e.g. position of the first plane or wire, the wire spacing, etc,
  - structure data products/modules to facilitate generic loops over geometrical elements
  - Define detector and data element IDs at all levels
- Similarly for calibration data, electric field map, database metadata etc.

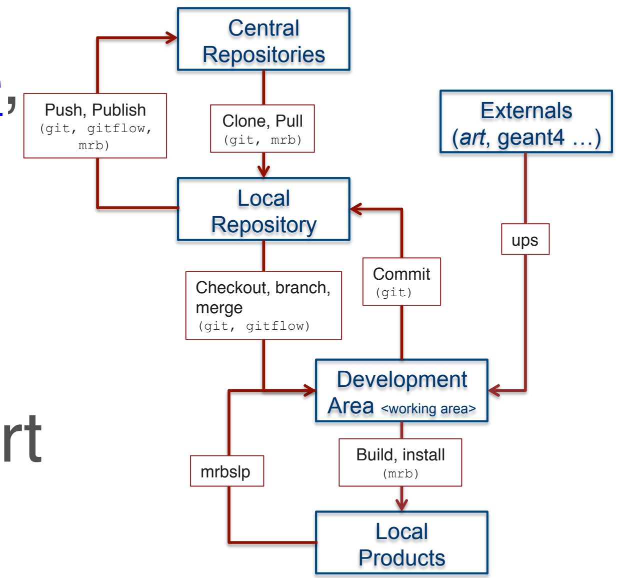
The Code:

- Number of different authors: 110 from more than 25 institutions.
- number of LarSoft code modules: 247
- Total lines of code (excluding configuration)

Language	Files	Lines:	blank	comment	code
C++	905		58,389	53,350	190,199
C/C++ Header	758		21,314	40,791	47,141
CMake	164		783	597	4,605
Perl	12		890	438	3,984
XML	17		157	174	1,823
Python	14		435	393	1,210
Bourne Shell	18		151	126	647
make	10		97	79	249
SUM:	*1,898*		*82,216*	*95,948*	*249,858*

The Code: Development Environment

- Redmine repository open to all.
- Source code build infrastructure based on:
  - ups (Fermilab code versioning), cmake, cetbuild/mrb (*art* build system)
- Wiki pages, Doxygen, LXR for documentation.
- LArSoft examples and *art* workbook support learning for development, patterns.
- Experiment-specific components live in experiment repositories: detector-specific geometry descriptions, electronics response functions, calibration functions, specific algorithms etc.

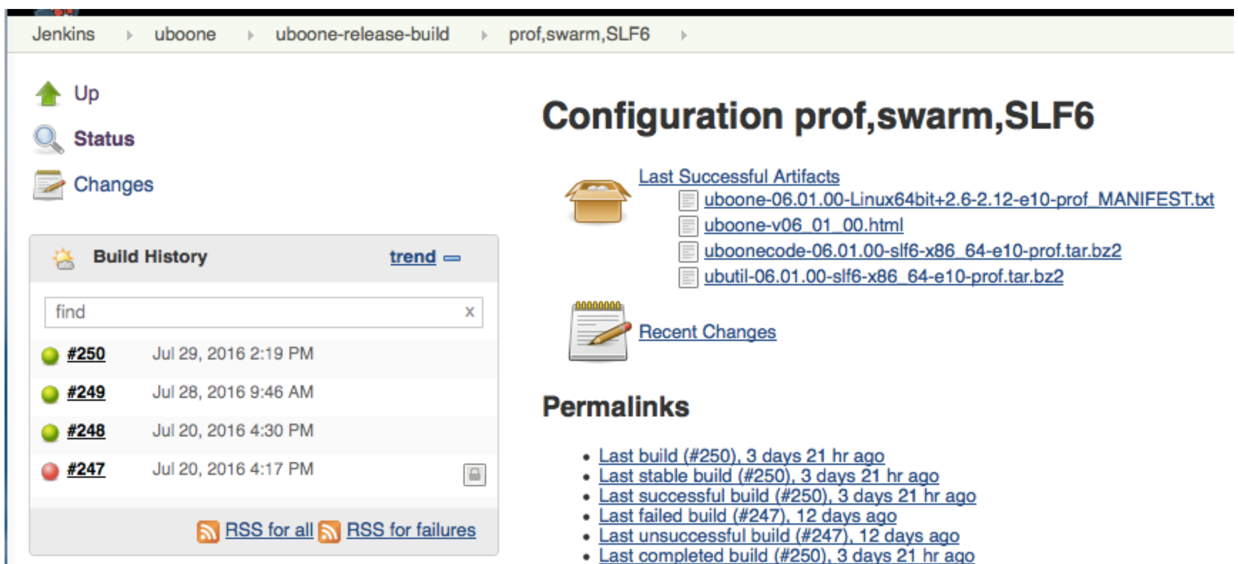


The Code: Release Management

- Core project provides integrated, tested, supported releases with new versions of and new modules for dependent and external products:
  - contributed algorithms.
  - ROOT 6, art V2.0, Geant4 V10
- Centralized release management for LArSoft core (Fermilab) and (separately) for Experiments (related git repositories)
- Multiple releases and branches supported simultaneously.
- Centralized distribution from web site and CVMFS
- Releases available for:
  - Scientific Linux (6, 7),
  - Ubuntu (14, and soon 16),
  - MacOSX (Mavericks, Yosemite)

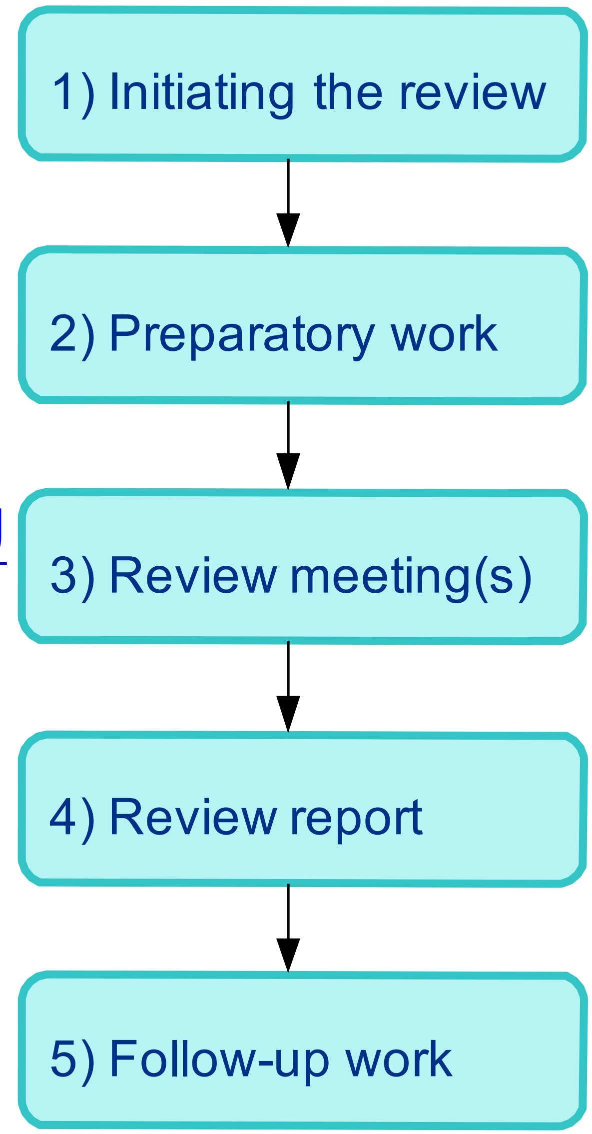
The Code: Continuous Integration Testing

- Centralized Jenkins framework and systems supports
  - Automated build and test program execution after each central repository commit
  - Automated email to Module owners of errors and warning
  - Recording of memory and CPU usage and comparisons between versions.
  - Support for distributed/remote hardware for further testing



The Code: Peer Analysis

- Review of contributed code through Coordination meeting discussions:
  - Proposals, architecture, design, implementation.
  - Read through by core developers.
- Support for performance measurement tools (igprof, valgrind, *art* memory and CPU trackers) and interpretation of their output.
- In depth code analysis including C++ experts.
- Have done 3 module analyses to-date with constructive and well received outcomes.
- Process includes commitments to time and follow up.



Short/Long Term Future (1 of 3)

- Continue to respond to immediate experiment requests, bug fixes etc.
- Continue to improve usability
  - Development project to use SPACK for software build/distributions
  - Deployment of light framework integration into MicroBooNE
- Extensions to integration with Pandora
  - Allow multiple trips to/from algorithms in LArSoft as part of end to end experiment workflow/chain.
- Foster easier use/configuration of development event display, analysis event display and other visualization tools
  - extend use of Paraview, Root, 2D and 3D and virtual environments.

Short/Long Term Future (2 of 3)

- BNL WireCell 3-d reconstruction package)– LArSoft integration
- FLUKA detector simulation – LArSoft integration
- Support for ProtoDUNE Dual Phase experiment
- Update interface (based on new *art* modules) for Geant4 and discuss GeantV when requested; consider Marley inclusion in Genie and/or LArSoft
- Include architecture extensions for current/new machine learning algorithms under active development in multiple experiments.
  – e.g. Extend data objects to better support standard image formats used by such methods

Summary

- LArSoft provides an architecture and software based on a common event framework, together with shared and experiment specific algorithms and tools for the simulation, reconstruction and analysis of LArTPC experiment data.
- An ultimate goal is to develop fully automatic processes for reconstruction and analysis of LArTPC events.
- The Collaboration includes the ArgoNeuT, LArIAT, MicroBooNE, DUNE and SBND experiments as well as Laboratory and University software developers and scientists.
- The project supports a common environment for and contributions of the use and development of algorithms aimed for a single or multiple experiments
- The collaborations are increasingly engaged and there are many plans for future work

Additional Slides

simb::MCTruth	the interaction generated by event generators like GENIE, Corsika, etc.; usually, one for each generator.
simb::MCFlux	the flux of particles toward the detector (neutrinos from the beam, cosmic rays, etc.); usually, one for every *simb::MCTruth*.
simb::MCParticle	a single generated particle, either by an event generator (GENIE, Corsika, …) or by the detector simulation (GEANT4).
sim::SimChannel	the electrons deposited on one TPC readout channel, as function of time, and connected to the generated particle that produced them.
sim::SimPhotons	the photons reaching one optical detector readout channel.
sim::SimPhotonsLite	the count of photons reaching one optical detector readout channel as function of time.
sim::MCHit	charge from a single particle seen by a TPC readout channel.
sim::MCTrack	the observable energy deposit coming from a single particle.
sim::MCShower	the observable energy deposit coming from a electromagnetic shower of particles.
raw::BeamInfo	beam status data.
sumdata::POTSummary	Protons On Target information (stored once per run).
raw::RawDigit	digitized signal on a TPC readout channel as function of time.
raw::OpDetWaveform	digitized signal on a optical detector channel as function of time.
raw::AuxDetDigit	digitized signal on a channel from an auxiliary detector as function of time.
raw::Trigger	a single trigger.
raw::ExternalTrigger	a single trigger from a source external to the TPC.
recob::Wire	calibrated signal from a TPC readout channel (the name is misleading!).
recob::Hit	signal from a single charge cluster on a TPC channel.
recob::OpHit	single from a scintillation event on a optical detector readout channel.
recob::Cluster	projection of a particle energy deposit on a single view, as a set of geometrically related hits.
recob::EndPoint2D	point on a TPC view pinning an extreme of a cluster.
recob::SpacePoint	point reconstructed in the cryostat volume.
recob::Vertex	point representing an interesting physics reaction (e.g., decay, creation, emission of a δ ray).
reco::Cluster3D	cluster of geometrically related, reconstructed space points.
recob::Track	a particle manifesting with a track-like trajectory (e.g. from muons, protons, etc.).
recob::Shower	a particle manifesting as a cascade of daugghter particles (e.g. from electrons and photons).
recob::PCAxis	3-D axis as extracted by a principal component analysis.
recob::Seed	a short 3-D segment, useful to start tracking.
recob::OpFlash	a scintillation flash reconstructed with the optical detector data.
recob::PFParticle	a reconstructed particle as member of a hierarchy describing the evolution in time of a physics event (*particle flow*).
recob::Event	identification of a single physics event (as opposed to the readout/*art* event).
anab::Calorimetry	energy of a reconstructed physics object.
anab::FlashMatch	connection between a light flash and a physics event in the TPC.
anab::T0	the time an interaction happened in the detector (commonly called t_0).
anab::CosmicTag	hypothesis on the nature of a physics object as a cosmic ray.
anab::MVAPIDResult	particle identification output from a multivariate analysis.
anab::ParticleID	particle identification hypothesis.

List of Currently Publicly Published Algorithms on Larsoft.org

Algorithm name	Author name	one line description
BlurredCluster	Mike Wallbank	2D cluster reconstruction technique which specialises in clustering hits from shower deposits by first applying a weighted Gaussian smearing to the hit map in order to more accurately distribute the charge and form more complete clusters.
ClusterCrawlerAlg	Bruce Baller	Reconstructs line-like 2D clusters, 2D vertices and 3D vertices.
EMShower	Mike Wallbank	3D shower reconstruction algorithm which takes 2D clusters in each view and produces 3D shower objects with all relevant properties
Fuzzy Cluster	Benjamin Carls	A 2D clustering algorithm that attempts to ID shower and track like objects
NucleonDecay	Tingjun Yang	A module to simulate nucleon decays.
Projection Matching Algorithm	Robert Sulej, Dorota Stefan	Reconstructs structures of 3D tracks interconnected with vertices; the input is 2D clusters.
Track3DKalmanHitAlg	Herbert Greenlee	Reconstructs tracks applying Kalman filter on hits.
TrackContainmentAlg	Wesley Ketchum	Groups tracks by containment.