LArSoft News and Announcements: Project Status

R. Snider
Fermilab

LArSoft General Meeting
Oct. 2, 2013

Topics for today

- Status of git transition work
- cvmfs and software distribution

The “git transition” plan

- The objectives:
  - To create a development environment that:
    - offers greater stability
    - eliminates the problem of inconsistent builds associated with SRT test releases
  - To improve and facilitate distribution and remote build capabilities
- The means:
  - Migrate LArSoft from svn into git repositories
  - Migrate from SRT to the cmake build system
  - Use “git flow” product to facilitate development workflows
  - Create a “multi-repository build” tool (mrb) to facilitate building the code
  - Package releases as “re-locatable” ups products
  - Document the new system and development workflow

General status of transition work

- Progress since last general meeting (8/21)
  - Populated a set of “alpha” repositories updated to mid-Sept snapshot
    - For prototyping the build system and re-factoring
  - Successfully built LArSoft using cmake (no mrb yet)
  - Created LArSoft ups products
  - Initial “release” created (but not intended for general user or testing!!!)
- Work in progress
  - Create “final” alpha repositories
    - Initial validation
      - Focused on demonstrating that the builds / ups packaging is fully functional
    - Complete any LArSoft re-factoring needed
      - Some depends upon results of initial validation
  - Implement “basic” mrb functionality
  - Produce documentation required to use / test alpha release

General status of transition work

- List of major tasks and milestones (not entirely time-ordered)
  - Create initial alpha release (git + cmake build + ups)
  - Final LArSoft re-factoring
  - Validate alpha release
    - verify that products work
    - validate LArSoft output
  - Define final LArSoft re-factoring
  - Create LBNE and uBooNE release areas
  - Create final alpha release
  - Create beta release transition (pre-production transition)
  - Beta release validation

  - Implement basic mrb functionality
  - Implement extended mrb functionality

Completed
In progress
Not started

General status of transition work

- List of major tasks and milestones (not time-ordered)
  - Produce expert documentation for initial alpha release testing
  - Produce documentation for alpha release validation
  - Produce extended git transition documentation for users

  - Perform final git transition
    - Freeze SVN repository
    - Create final SVN release from head
    - Transition to git from SVN head
    - Create release from git
    - Validate git release against final SVN release
    - Go live

  - Automated LArSoft release / binary distribution
  - Demonstrate automated remote build

Completed
In progress
Not started

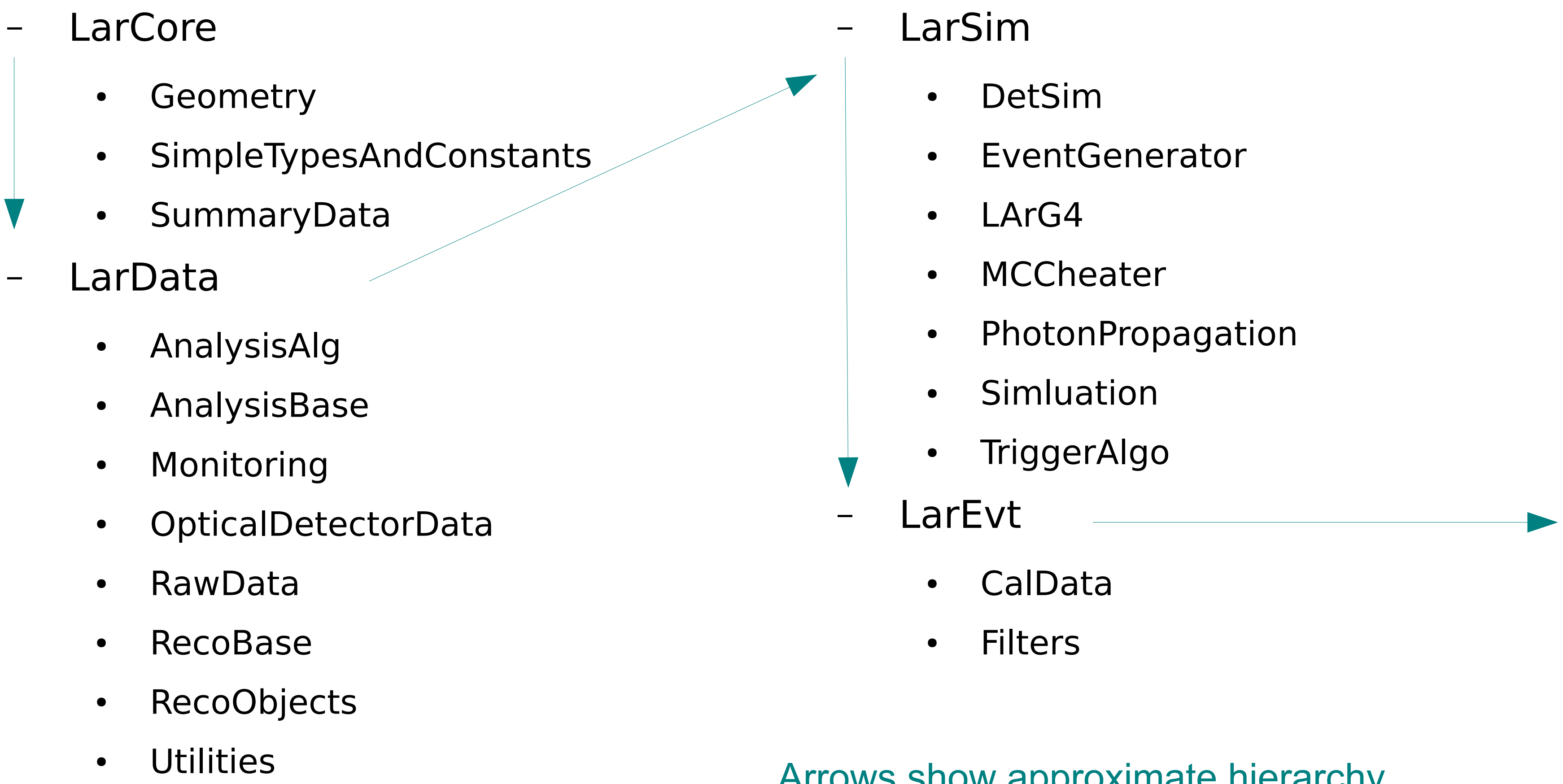
General status of transition work

- Where to find the alpha repositories, release, documentation
  - See redmine project “LArSoft Alpha Test” and sub-projects
    - https://cdcvs.fnal.gov/redmine/projects/larsoft-alpha
    - Alpha repositories live there. Work notes are on the wiki page.
    - Instructions for building / using alpha release are on Alpha Test wiki

    (Note: everything will live on “LArSoft” redmine project when done)
  - LArSoft products (“release”) area
    - /grid/fermiapp/larsoft/products
    - Mounted on all LBNE, uBooNE, LArRandD, T-962 and Darkside GPCF nodes
  - Plan documented and tracked on LArSoft sharepoint site
    - Official task list and milestones maintained there
    - https://sharepoint.fnal.gov/project/LArSoft

Alpha release components

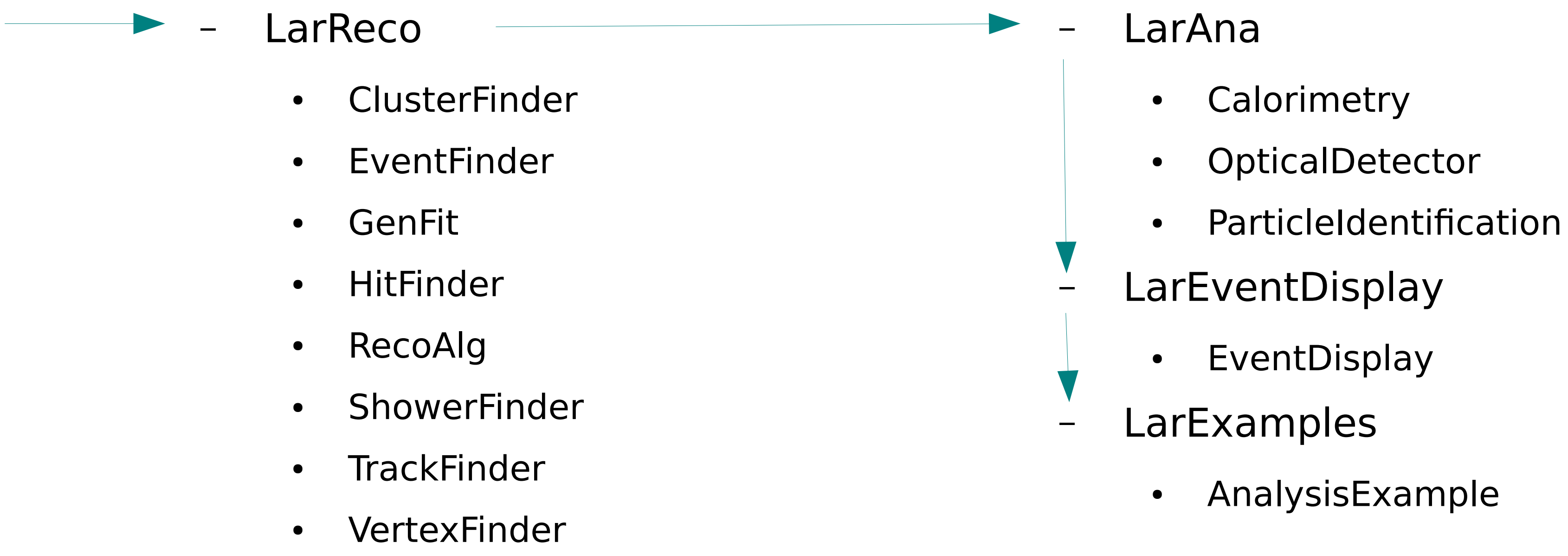
Full dependency tree: https://cdcvs.fnal.gov/redmine/projects/larsoft-alpha/wiki/LarSoft_Dependency_Tree



- LarCore
  - Geometry
  - SimpleTypesAndConstants
  - SummaryData
- LarData
  - AnalysisAlg
  - AnalysisBase
  - Monitoring
  - OpticalDetectorData
  - RawData
  - RecoBase
  - RecoObjects
  - Utilities
- LarSim
  - DetSim
  - EventGenerator
  - LArG4
  - MCCheater
  - PhotonPropagation
  - Simluation
  - TriggerAlgo
- LarEvt
  - CalData
  - Filters

Arrows show approximate hierarchy
(not full dependencies)

Alpha release components



- LarReco
  - ClusterFinder
  - EventFinder
  - GenFit
  - HitFinder
  - RecoAlg
  - ShowerFinder
  - TrackFinder
  - VertexFinder
- LarAna
  - Calorimetry
  - OpticalDetector
  - ParticleIdentification
- LarEventDisplay
  - EventDisplay
- LarExamples
  - AnalysisExample

Arrows show approximate hierarchy
(not full dependencies)

Alpha release plans

- Some re-factoring of code will be needed
  - To address physical design issues (only one case so far)
  - To extract experiment-specific code
    - Will go into experiment repositories maintained by the experiments
  - Other changes that will enhance maintainability (and are tolerable)
- Validation
  - Will enable and exercise unit testing (via “make test”)
  - Will also run some example programs, then compare results
    - Importing tools to compare output of one release to another
    - Would like to have comparisons at binary level
    - Will be needed to test final alpha, beta, and final migration releases
  - User testing will be opened once mrb works and documentation is ready

All work, instructions will be documented on the Alpha Test wiki

Workflow implementation

- For users

```
setup <experiment>
<(possibly experiment-specific) command to create work area>
cd <work_area>
<check out experiment software>
<...work...>
<build>
<run>
...      .
```

- For developers

```
setup <experiment>
<(possibly experiment-specific) command to create work area>
cd <work_area>
<check out experiment software>
mrb checkout <LArSoft component> [<tag>]
git flow feature start
<...work...>
mrb setup
mrb build
<run>
git flow feature finish
...
```

Code distribution

- Ability to distribute, install and run the software on off-site machines is a requirement
- Will use three basic approaches
  - Binary distributions
    - Suitable for “supported” platforms that we build on-site
  - Source distributions that are built locally at remote sites
    - The project will provide assistance with building on any unix-based platform the experiments decide is needed
  - Distribution via cvmfs

cvmfs

- CernVM File System
  See http://cernvm.cern.ch/portal/filesystem
  and http://www.slideshare.net/traylenator/cvmfs-workshop
  - An http-based, network file system
    - Appears to applications as a file system mounted on a local disk
  - Optimized to deliver experiment software in a fast, scalable, reliable way
    - Expects to see lots of little files
    - Additions on the time scale of hours, days
    - “Aggressive caching” and transparent downloading on demand
  - Files written to a single repository node
    - Can then be read in “100,000's of locations”
  - Has been used for distribution to both batch and interactive clients
  - Has been adopted by Fermilab as part of grid-enabling IF experiments and managing disk resources

cvmfs

- Required infrastructure
  - A repository server that store copies of all files to be distributed
    - OSG maintains one at Indiana for use with OSG grid sites (Oasis)
  - Fermilab may have servers for use on Fermigrid
    - Part of the strategy to remove BlueArc mounts from Fermigrid nodes
  - Client infrastructure already in place on all OSG grid sites
    - Software components distributed with all linux distributions
    - Requires local squid cache, which most places have (though maybe not at the appropriate scale)
    - Most institutions with any significant computing will have the client infrastructure
- LArSoft and cvmfs
  - Will upload all releases and nightly snapshots to Oasis server
    - Already started working on this
    - Will assist with configuring clients for access to LArSoft

The end