LArSoft stakeholders meeting

E. Snider
L. Garren
G. Petrillo
R. Pordes

Fermilab

Feb 12, 2014

Today's agenda

- Transition follow-up
- Discussion of managing integration of code changes
- Compiler warnings
- Optimization (Gianluca)

Transition follow-up

- Treating the following open issues as highest priority to resolve
  - Remote source installation and build
    - Issues with encumbering low-level cmake configuration with cetbuildtools + ups
    - Project under way to test proposed solution (Brett Viren, Ben Morgan + art team)
  - Nightly release distribution
    - Eventually aiming for cvmfs solution. Not supported by existing cvmfs solution.
    - Working on an off-site nightly build. Requires changes to nightly build script.
  - Problems with ups initialization and some mrb under csh and zsh
    - Work planned, will start next week. Expect to be done in about 4 weeks
  - mrb bug reports / feature requests
    - Just released mrb v0_05_07 that fixes all outstanding issues
  - mrb enhanced features
    - Will identify when dependent products need to be included in the build
    - Work will take about one month. Will start after shell problems fixed.

Managing code integration

- Recommend we adopt a process to manage integration of code changes
  - LArSoft serving multiple experiments
    - Different priorities and lines of development for each
  - Currently 46 code authors, > 1000 files, ~150k lines of code
    - More authors coming on board as LBNE effort ramps up

Easy for one author to break the work of others

Managing code integration

- Several elements of a management system discussed previously
  - Package librarians / area coordinators
    - People who “own” the code integration process in one or more related packages
    - Responsible for making sure the code builds, works as expected.
    - Performs tagging or other “gate-keeping” functions if required
  - Continuous integration (CI) processes / infrastructure
    - A process for frequently merging changes into the main development line.
      - Low latency between commit and build / test
      - Verify that all changes pass a set of tests prior to integration
        - *eg, the feature being integrated + a standard suite of meaningful tests*
      - Trigger on commit or manually by librarian / area coordinator
    - Some advantages
      - Rapid detection of bugs, integration problems
        - *When problems found, can revert main branch before adversely affecting others*
      - Immediate feedback on code quality
      - Can make (at least some) of the resulting builds available
        - *Constant availability of high-quality release near the main development head*

Continuous integration

- An example: the CMS model
  - Build infrastructure
    - Utilizes Jenkins build scheduling / automation system
      - Required relatively little additional programming compared to other choices (eg, buildbot)
    - Executes the following workflow
      - Build + unit testing + integration tests and physics validation + quality assurance tests
    - Performs workflow 20-30 time per day; two delivered as full integration releases
    - Occasional builds of special branches to accommodate large-scale changes, product migrations, etc.
  - git version control system
    - Maintain “master” branch + many “topic branches”
    - Developers make integration requests for specific topic branches
    - Area coordinators review and approve changes based on test suite results (CI process) --- only area coordinators can merge into equivalent of “develop”
  - GitHub git repository hosting
    - Provides tools to automate the code integration process + other features

LArSoft code integration

- Recommend we adopt a CI approach
- CI model requires infrastructure we don't yet have
  - Integration build system
    - SCD currently planning to provide a solution for IF
      - Have experience with buildbot and Jenkins
      - Target timescale is approx 6-9 months (not official!!)
    - SCD gathering requirements for build platform
      - Trying to ensure that the system will meet the needs of CI
    - Brett V. has draft system requirements for LBNE. By my reading, very similar to those needed for LArSoft
      - A good start.
  - Integration workflow automation
    - May be part of above solution

  Goal will be to make it possible for a single release manager to perform integration

LArSoft code integration

- What to do in the mean time
  - Create frequent integration releases: vX_YY_XX
    - At least weekly? + as needed
    - More often?
  - Rely on librarians
    - Take responsibility for making sure code is ready for integration
  - Build out formal integration test suite
    - Project will coordinate this with the experiments
  - Coordinate all breaking changes with the release manager
    - This is important!!
  - Continue to develop policies, procedures to assist

Discussion on this plan...

Primary package authors

- Identified “primary” package authors (our guesses).

AnalysisAlg
baller
tjyang

AnalysisBase
andrzejs
bjpjones
tjyang

AnalysisExample
seligman

CalData
andrzejs

Calorimetry
tjyang

ClusterFinder
baller
bcarls

DetSim
andrzejs
greenlee

EventDisplay
brebel

EventFinder
??

EventGenerator
brebel

Filters
??

Genfit
echurch

Geometry
brebel

HitFinder
jasaadi
talion
wketchum

LArG4
brebel

LArPandoraAlgorithms
blake

LArPandoraInterface
blake

MCCheater
brebel

OpticalDetector
bjpjones

OpticalDetectorData
??

ParticleIdentification
tjyang

PhotonPropagation
bjpjones

RawData
brebel

RecoAlg
andrzejs
baller
bcarls
bjpjones
greenlee
talion
tjyang
wketchum

RecoBase
brebel

RecoObjects
bjpjones
greenlee

ShowerFinder
andrzejs

SimpleTypesAndConstants
brebel

Simulation
brebel

SummaryData
brebel

TrackFinder
bjpjones
brebel
echurch
greenlee
soderber

TriggerAlgo
kterao

Utilities
andrzejs
bjpjones
brebel

VertexFinder
jasaadi
wketchum

Compiler warnings

- Currently warnings are turned on, but are just “warnings”
  - Using:

    ```
    -Wall -pedantic -Wno-unused-local-typedefs -Wno-unused-variable
    ```
- Fixed warnings prior to cutting v1_00_02
  - Fixed warnings found with a more demanding configuration :

    ```
    -Wall -Wextra -pedantic -Wno-long-long -Winit-self -Wno-unused-local-
    typedefs -Wformat-y2k -Wswitch-default -Wsync-nand -Wtrampolines
    -Wlogical-op -Wno-ignored-qualifiers -Wno-error -Wno-overloaded-virtual
    -Wno-unused-parameter -Wno-switch-default
    ```
  - But did not enable all these flags in the repository
- Warnings identify lots of bugs, so want to make them errors
  - Use the more strict list of warnings
  - Then set `-Werror`

Target this change for a release within the next two weeks (??)

Code optimization

- Report from Gianluca