Hall C 12 GeV Software Progress
Software Meeting
Nov 5, 2012

Milestones

Present

- Set-up Management structure
- Monte Carlo simulation is ready
- Decided on Git for code management of C++ analyzer

2012

July : Define reference HMS data for testing code

Sep : Documented non-tracking HMS detectors code in Fortran Analyzer

Oct : Make DAQ decoding in C++ Analyzer object-oriented

Oct : Ability to analyzed Hall C data at the raw data level in C++ Analyzer

Dec : Documented the drift chambers and tracking code in Fortran Analyzer

Dec : Verify HMS hodoscope analysis in C++ Analyzer

Milestones (Part 2)

2013

Jun : SHMS code added to Fortran Analyzer.

July : Full analysis of HMS data with C++ Analyzer ready

Sep : C++ Analyzer ready for SHMS calorimeter tests.

Dec : Full analysis of HMS data with C++ Analyzer verified by comparison to Fortran analyzer.

2014

Jan : Scalar and BPM analysis code in C++ analyzer

Feb : Calibration codes ready.

Jul : Analyze cosmic ray data in SHMS with both Analyzers

Sep : First beam, analyze data with both Analyzers

Review Hall C Recommendations

- With the somewhat aggressive schedule leading up to December 2013, make sure to engage a reasonable number of early adopters to stress test the new framework.
- Re-use existing efforts from Hall A to decode CODA-formatted data in ROOT.
- If resources are limited, the Fortran-based SHMS reconstruction should be a low priority.
- While we encourage the move to git as a code management system, be sure not to underestimate the extent of the paradigm shift. Identify a workflow model for your use of git. Communicate clearly the new paradigm (easy branching, no central repository, etc) Set up (or link to) tutorials for users with a mapping of routine CVS tasks to their git equivalents (such as cvs diff, etc). Document or link to documentation for standard git tasks without obvious equivalent in CVS or SVN, such as git rebase, or bisect.

Actions:

- Biweekly software meetings
- Definitely using the existing Hall A software to decode CODA
- To promote Git usage:
  - Arranged a seminar on Git
  - Encourage use in managing other projects

Review General Recommendations

- Nightly builds are performed by some; we recommend them for all.
- Evaluate standard code evaluation tools, such as valgrind, clang's scan-build, cppcheck, Gooda, ... for inclusion in the software development cycle. We suggest looking at an Insure++ license as well.
- Run a code validation suite such as valgrind as part of the routine software release procedure.
- Give full and early consideration to file management, cataloging and data discovery by physicists doing analysis. Report on this area in future reviews.

Actions:

- Started to investigate code evaluation tools

Progress on Milestones

- Selected HMS run for analysis
- HMS documentation
  - Calorimeter done.
  - Scintillator planes has started.
  - Aerogel , Cerenkov and Drift chambers not started.
- HMS coding in “hcana”
  - Can analyze data at the raw CODA level. Read-in Hall C detector maps and parameter files.
  - Calorimeter and scintillator has made comparison to Fortran analyzer with “slightly processed” data.
  - Aerogel detector has started

Management Structure

Software Manager
Mark Jones
Jefferson Lab

C++/ROOT Analyzer
Gabriel Niculescu,
James Madison University

Fortran Analyzer
Ed Brash
CNU

Simulation (SIMC)
David Gaskell
Jefferson Lab

Calibrations
John Arrington,
Argonne National Lab

Online histogramming
Pete Markowitz,
Florida International University