GLAST Large Area Telescope: Release System for GLAST Offline Software

Karl Young, Richard Dubois, Alex Schlessinger
Stanford Linear Accelerator Center
kyoung@slac.stanford.edu

http://www-glast.slac.stanford.edu/software
Outline

- Introduction to GLAST and GLAST Offline Software

- Description of Offline Software Release Management System:
  - Nightly Build System
  - System Tests

- Summary
GLAST measures the direction, energy and arrival time of celestial gamma rays

-LAT measures gamma-rays in the energy range ~20 MeV - >300 GeV
  - There is no telescope now covering this range!!

- GBM provides correlative observations of transient events in the energy range ~20 keV – 20 MeV

Launch: September 2006
Florida

Orbit: 550 km,
28.5° inclination

Lifetime: 5 years
(minimum)

K. Young
GLAST Instrument: Large Area Telescope (LAT)

- Array of 16 identical “Tower” Modules, each with a tracker (Si strips) and a calorimeter (CsI with PIN diode readout) and DAQ module.
- Surrounded by finely segmented ACD (plastic scintillator with PMT readout).
GLAST Offline Software

Source Fluxes

Geometry

Particle Transport

“Raw” Data

Recon

Background Rejection - Particle ID
Simulation software in C++; uses standard HEP software tools:
CMT, Geant4, Gaudi, Root, CLHEP

Uses xml for representation of sources and geometry

Linux and Windows are supported operating systems
Motivation for Nightly Builds

Want to encourage developers to commit changes early and often while avoiding a train wreck at release time.

Need to keep track of which versions of a large number of interdependent packages work (i.e. build and pass unit tests) and work together.
Nightly Build Strategy

• Specify all packages associated with a software release via a “container” package – GlastRelease (cmt package with list of packages in the cmt requirements file) –
  - a tag of GlastRelease is a release
  - the HEAD of GlastRelease is a release in progress
  - LATEST (latest version of packages contained in GlastRelease) is a potential future release

• Require package maintainers to provide a unit test with their package (so far about 60 % compliance)

• Package maintainers specify which version of their package should be specified in current version of GlastRelease (not necessarily latest version of package)
Nightly Build Strategy (continued)

• Nightly build scripts try to build and run tests for:
  - packages in current version of GlastRelease (if it has changed since previous night)
  - HEAD, i.e. head version of GlastRelease (which contains tagged versions of contained packages)
  - LATEST, i.e. the latest versions of all packages specified in GlastRelease (to try and anticipate looming build, run, and compatibility problems)

• Before a release is officially declared the system tests are run on the current version of GlastRelease, provided it successfully built and unit tests ran successfully, and the results of the system tests are evaluated.
Nightly Build Strategy (continued)

Web page showing list of packages specified by GlastRelease, versions of those packages specified in the current version of GlastRelease, versions specified in the head of GlastRelease, and the latest tags for the packages in the cvs repository.
Web page showing status of builds and tests for different versions of GlastRelease as well as status for builds of the head of GlastRelease, and the “latest” build (builds and tests using latest versions of packages specified in GlastRelease)
System Tests

- Provide end to end test of system under different conditions (after verifying that the last nightly build was successful)

- Track performance of offline software, release by release, via a broad range of diagnostics

- Allow for use of diagnostics, via comparison to a standard set of diagnostic results, to determine when GlastRelease is actually ready for release
Architecture of System Tests

Executive Script (Perl)

Oracle DB

Web Interface (Carrot, JAS, ..., Scripts)

Suite of Simulations Run on Supported Platforms

Root Files, Stdio Dumps

Root Scripts

Root Histogram Files
System Test Database

System test relational tables (as a part of larger data tracking and release system database) designed to maximize flexibility and extensibility – allows for tracking, across software releases, of both:

• instrumental quantities (track multiplicity, energy deposited in subsystems, …)

• operational quantities (cpu and memory usage, …)
Entity diagram for system test tables in DB
Web Interface

• Intended to provide easy access to and analysis of system test results

• Based on scripts (currently experimenting with both Carrot and JAS) that display histograms and metadata form system test database

• Prior to programs like Carrot and JAS, displaying such extensive information via the web was difficult

• Histogram display allows for overlaying of test and standard histograms for quick visual impression (scripts also calculate the Kolmogorov-Smirnov statistic as a diagnostic measure of the difference between test and standard histograms)
Comparison of Carrot and JAS

• **Carrot**
  - Web development reasonably easy modulo subtleties re. Carrot “write” statements
  - Carrot threads are fragile - require bullet proof scripts
  - Oracle connections require initial Root configuration setup
  - Root functionality native
  - Plotting faster than JAS (but on faster dedicated machine and with no optimization of JAS)
  - Last release winter ’02
  - Broke with Root 3.04.02
  - Remote support
  - Possible security problems

• **JAS**
  - Web development easy via, e.g. FrontPage
  - Oracle connections, i.e. ODBC connections via IIS, straightforward
  - Fetching DB results with vbscript straightforward
  - Lacks some of Root’s functionality
  - Possibility of using netbeans (gui tool) for building Root access pages
  - Local support available
Web Interface (continued)

Introduction

Welcome! This system allows us to track the performance of our workspace Gleam applications as it evolved through its early releases. It allows us to run several configurations of Gleam (e.g., different energies, different geometries and modes, etc.) and compare the output histograms to previous standard versions.

The portal to the system is at http://www.glast.slac.stanford.edu/software/systems/TestsInfoRD.asp

JAS is used as the web plot display tool, with apg and visual basic providing the connection to the underlying database.

Usage

Having selected a version of GleamTest.exe (which sets the Gleam version), you will be presented with a table of tests for that version. It has the following features:

- time the CPU time and memory used in the process generating the test output
- a link to view the histograms
- a button showing all the metadata associated with the test (currently mean, sigma, # entries and KS test value for each histogram)
- buttons showing the KS test results above and below threshold, and a summary count of those two conditions.

If you are not familiar with KS testing for comparisons of distributions, see this link.

Once you select a particular test’s histograms to browse, you’ll see a pull-down menu of histogram titles available in the Root file and a button to enable overlays.

Caveats

March 18, 2003

Web portal (JAS) to system test information
Web page containing diagnostic info. and access to histograms for specific system tests
Web Interface (continued)

Web page containing metadata for specific system tests
Web page containing overlaid Root histograms from system tests and standard
Summary

• A reasonably robust Release Management System for GLAST Offline Software has been designed and mostly implemented

• Nightly build system allows for rapid development and continuous testing of complex, multi developer, multi institution software, and thus greatly reduces problems encountered at release time

• System tests allow for easy viewing and analysis of extensive diagnostic information prior to release

• Focus now shifting to content of system tests rather than infrastructure design