GLAST Large Area Telescope:

Use of Carrot and Root

Richard Dubois
Stanford Linear Accelerator Center
richard@slac.stanford.edu

Heather Kelly
Goddard Space Flight Center
heather@lheapop.gsfc.nasa.gov

Outline

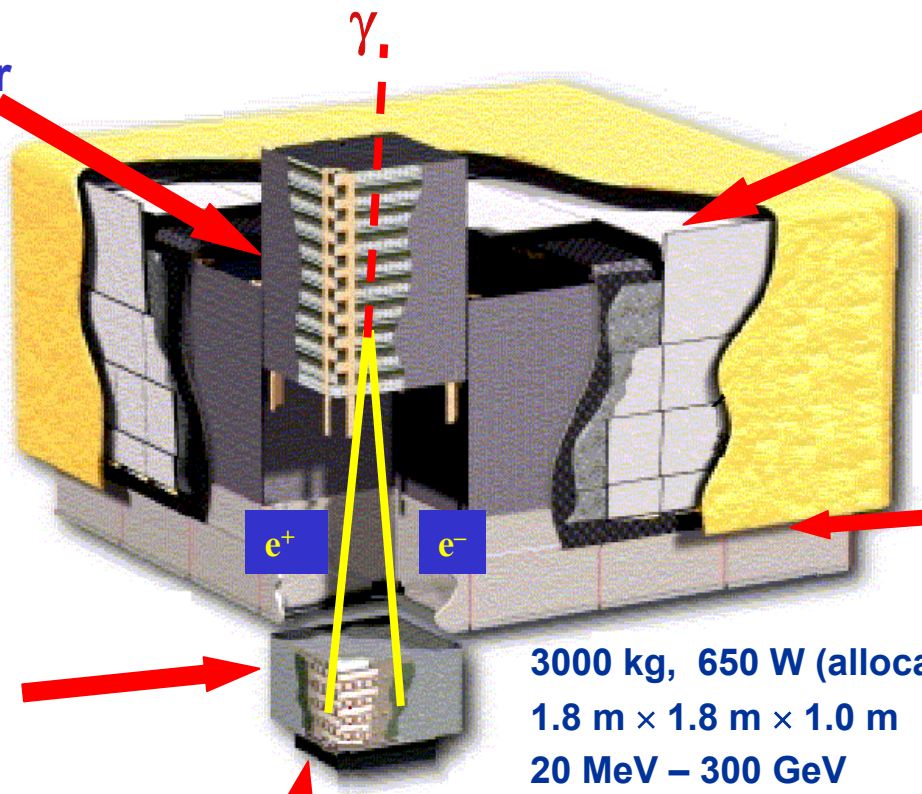
- Brief Introduction to GLAST and the Offline Software
- Creating a Systems Test facility for Code Release Management and the Data Pipeline
- Using Carrot to display System Tests output
- GLAST and Gaudi and Root

GLAST Large Area Telescope (LAT)

~200 collaborators

Si Tracker Tower

pitch = 228 μm
5.52 10^4 channels
12 layers × 3% X_0
+ 4 layers × 18% X_0
+ 2 layers



ACD

Segmented scintillator tiles
0.9997 efficiency
⇒ minimize self-veto

Grid (& Thermal Radiators)

3000 kg, 650 W (allocation)
1.8 m × 1.8 m × 1.0 m
20 MeV – 300 GeV

CsI Calorimeter

Hodoscopic array
8.4 X_0 8 × 12 bars
2.0 × 2.7 × 33.6 cm
⇒ cosmic-ray rejection
⇒ shower leakage correction

Data acquisition

Launch Sept 2006

Sim/Recon Toolset

applications

Root, IDL – analysis

Reconstruction
TkrRecon, CalRecon, AcdRecon

simulation package
GEANT4

xml – geometry, parameters

Root – object I/O

Gaudi – code framework

VC++ – Windows IDE
gnu tools - Linux

vcmt – Windows, linux gui

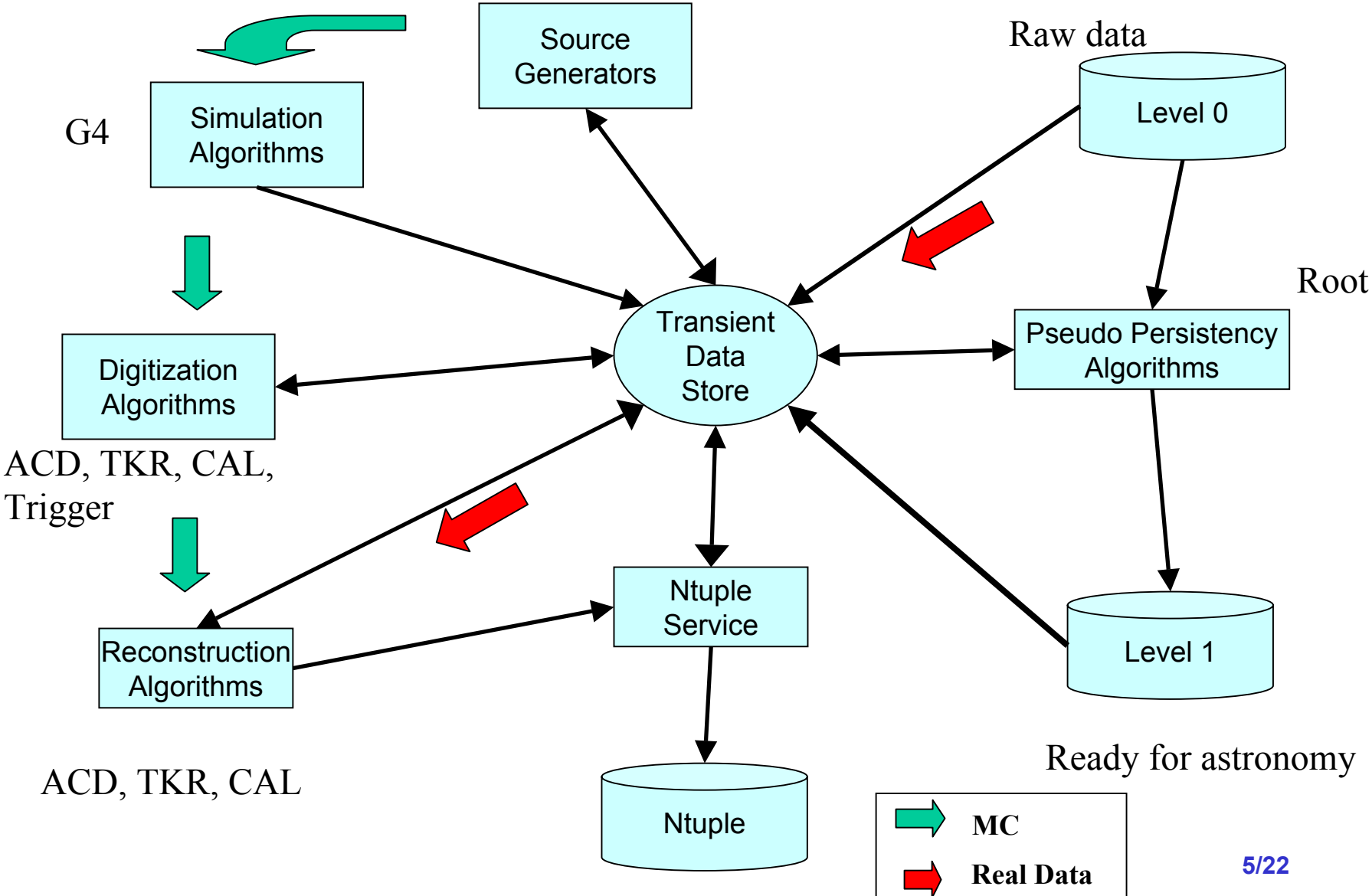
CMT – package version management

ssh – secure cvs access

cvs – file version management

utilities

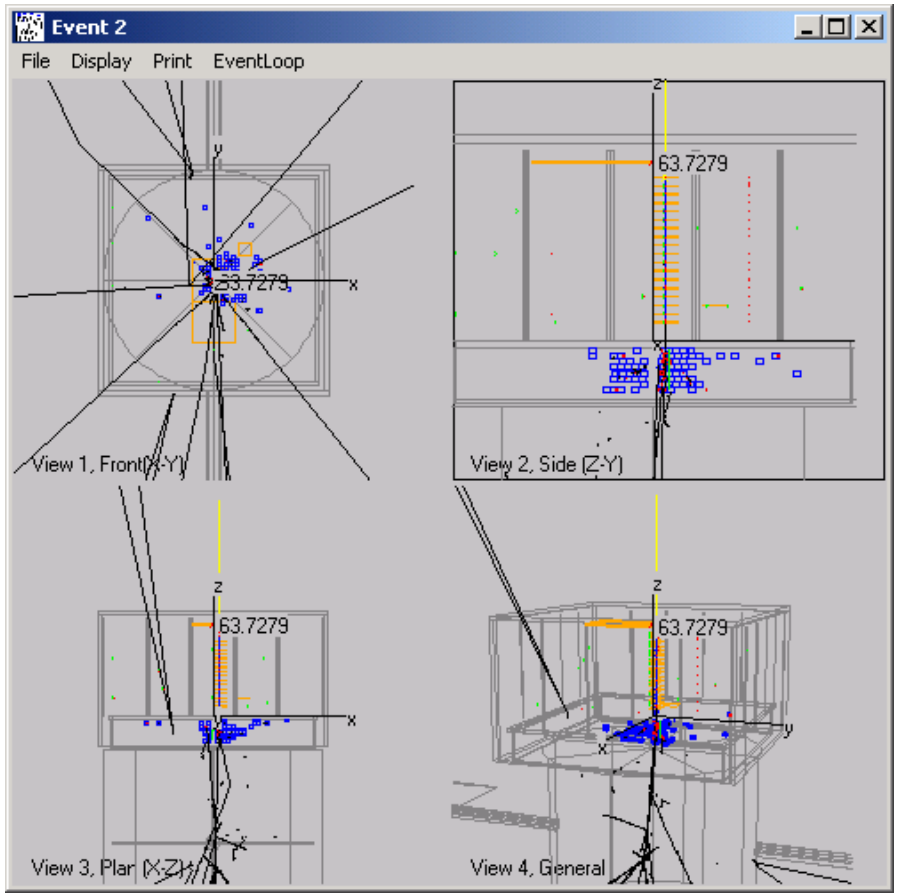
Data flow in the Gaudi framework



System Tests Facility

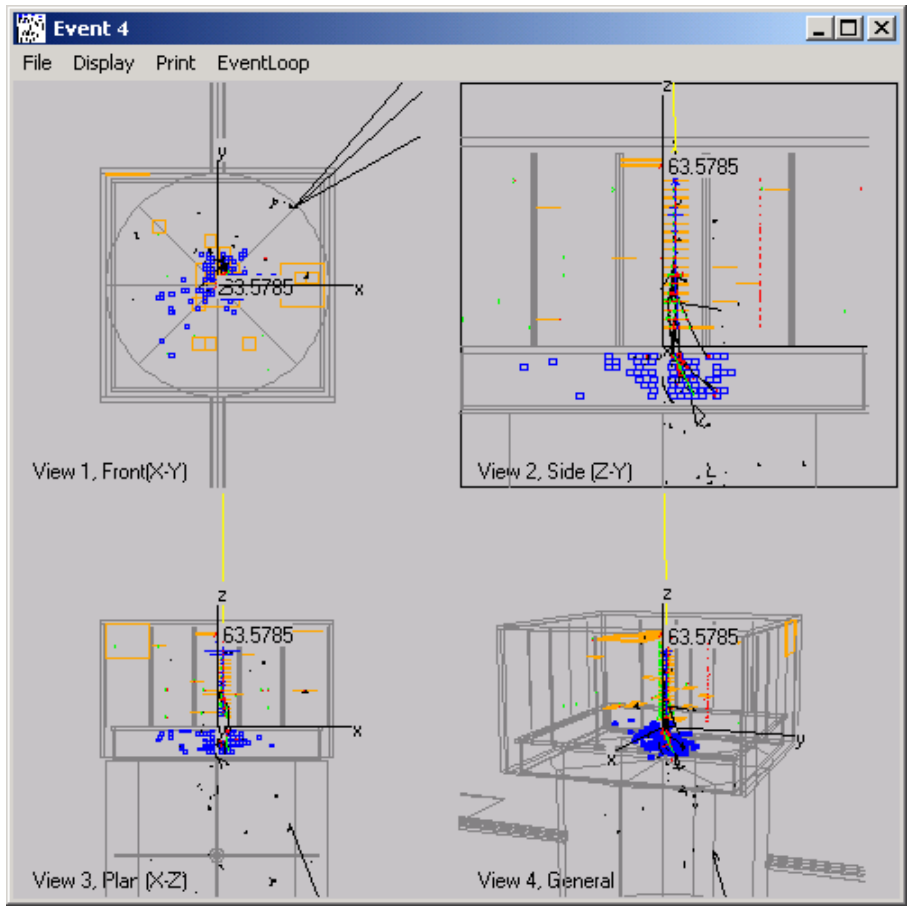
- **Motivation**
  - Comprehensive tests producing histograms & diagnostic statistics
  - A subset of the suite a subsystem would have for its own performance reviews
  - Multiple test configurations possible
  - Run on tags and releases
  - Tracked in database with web plotting display capabilities
  - Comparison to standards (deemed 'correct' by package owners)
  - Tests to be discovered and run automatically by the code Release Manager
    - will capture results for the db

Tests, Tests, Tests



10 GeV proton

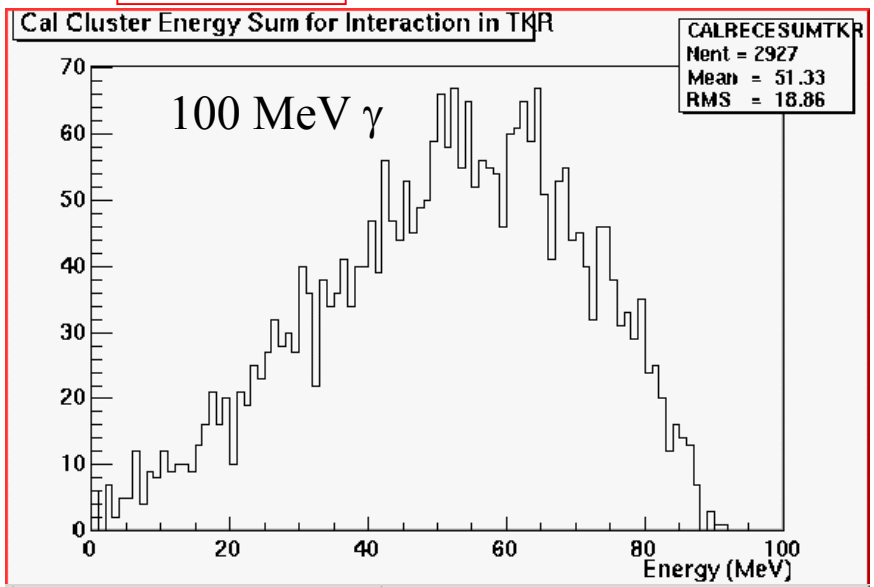
1 GeV gamma

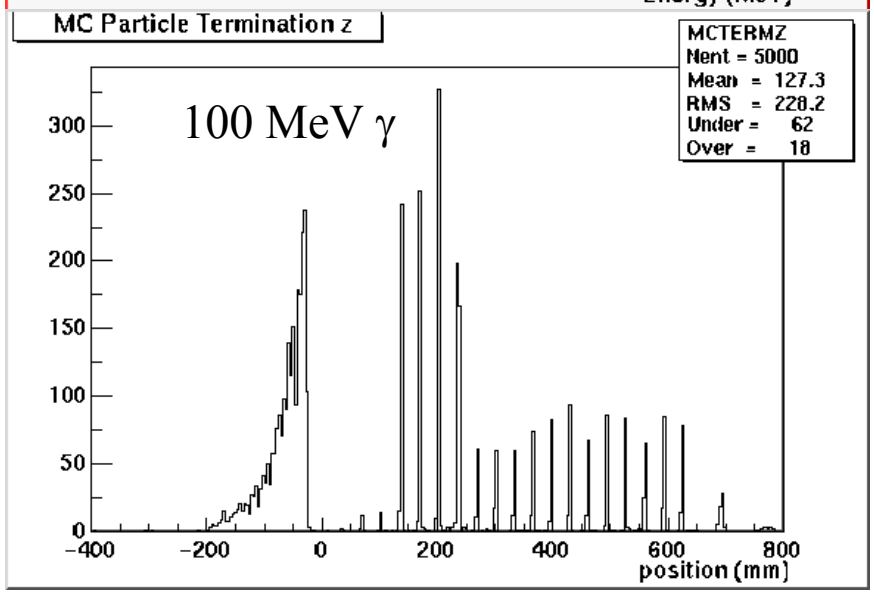


Use muons, gammas and protons at different energies and angles as test configurations: GLAST covers 20 MeV to 300 GeV!

More Tests, Tests, Tests

Detailed plots





Time, memory usage

CPU time : 17056.34 sec.
Max Memory : 97 MB
Max Swap : 120 MB

TkrReconAlg:exe... INFO Time User : Tot= 47[min]
Ave/Min/Max= 1.81(+- 11.8)/ 0/ 268 [s] #=1557

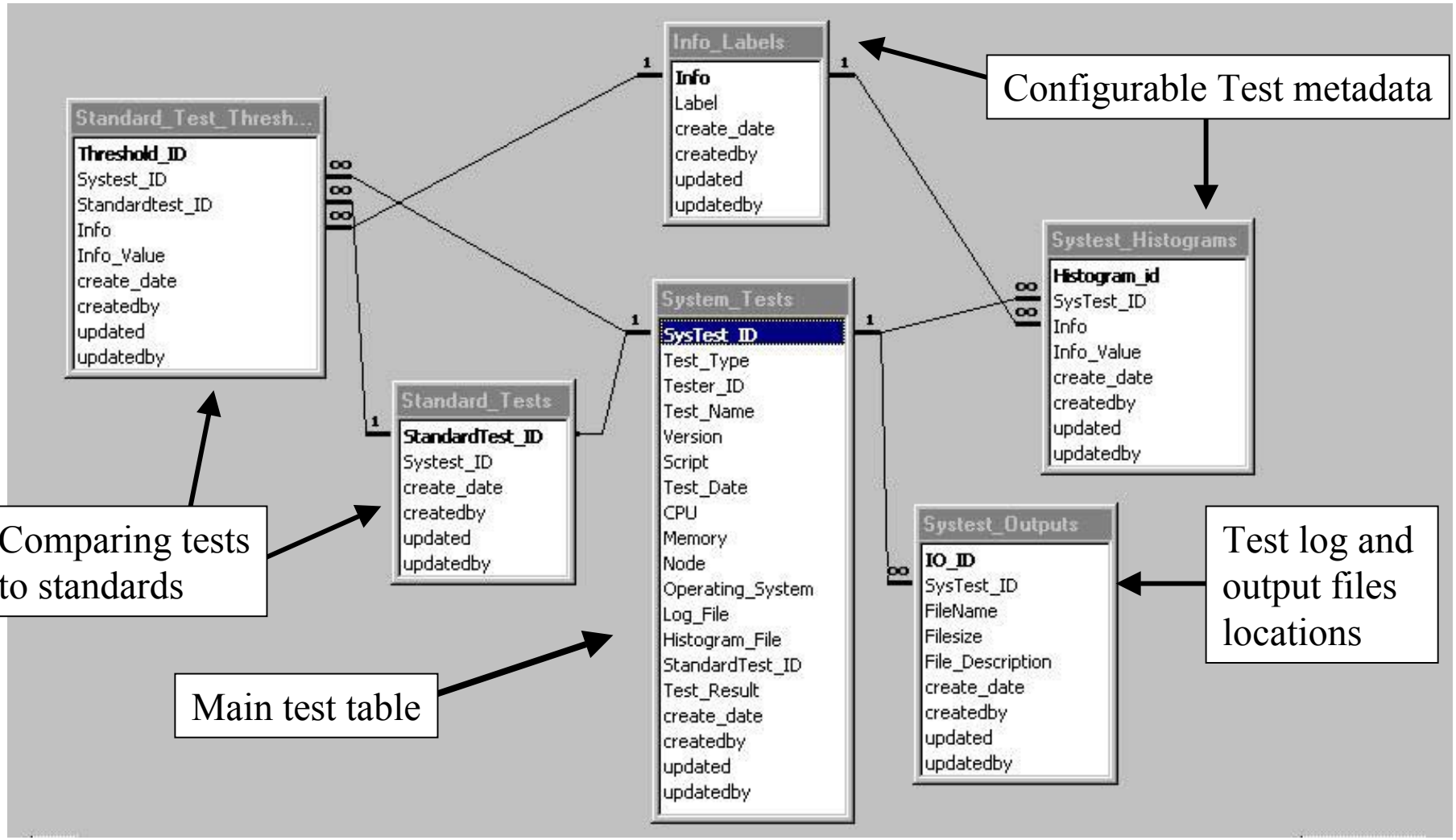
G4Generator:exe... INFO Time User : Tot= 110[min]
Ave/Min/Max= 1.31(+- 8.14)/ 0/ 126 [s] #=5000

ChronoStatSvc INFO Time User : **Tot= 165[min]** #= 1
*****Chrono***** INFO

Reconstruction performance metrics: resolution and efficiency

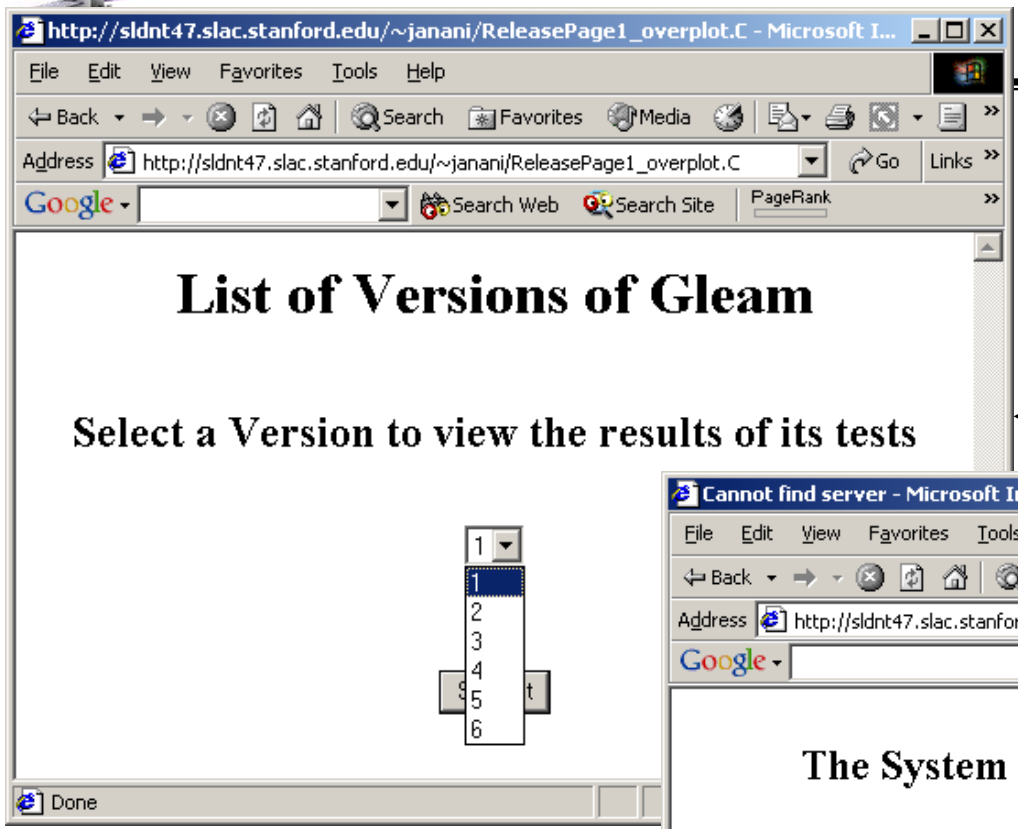
---------- Layers 12-15 Events used : 973
eff. proj. sigma : 4.08 deg = 245 arc-min
68% contained : 6.85 deg = 1.11*(1.51*sigma)
95% contained : 18.1 deg = 1.81*(2.45*sigma)
Energy: meas/gen :0.527
std : 0.196
events w/ no data : 20
effective area : 5838 cm^2
Figure of merit : 1074 cm
---
total effective area : 12066 cm^2
Combined FOM : 2409 cm

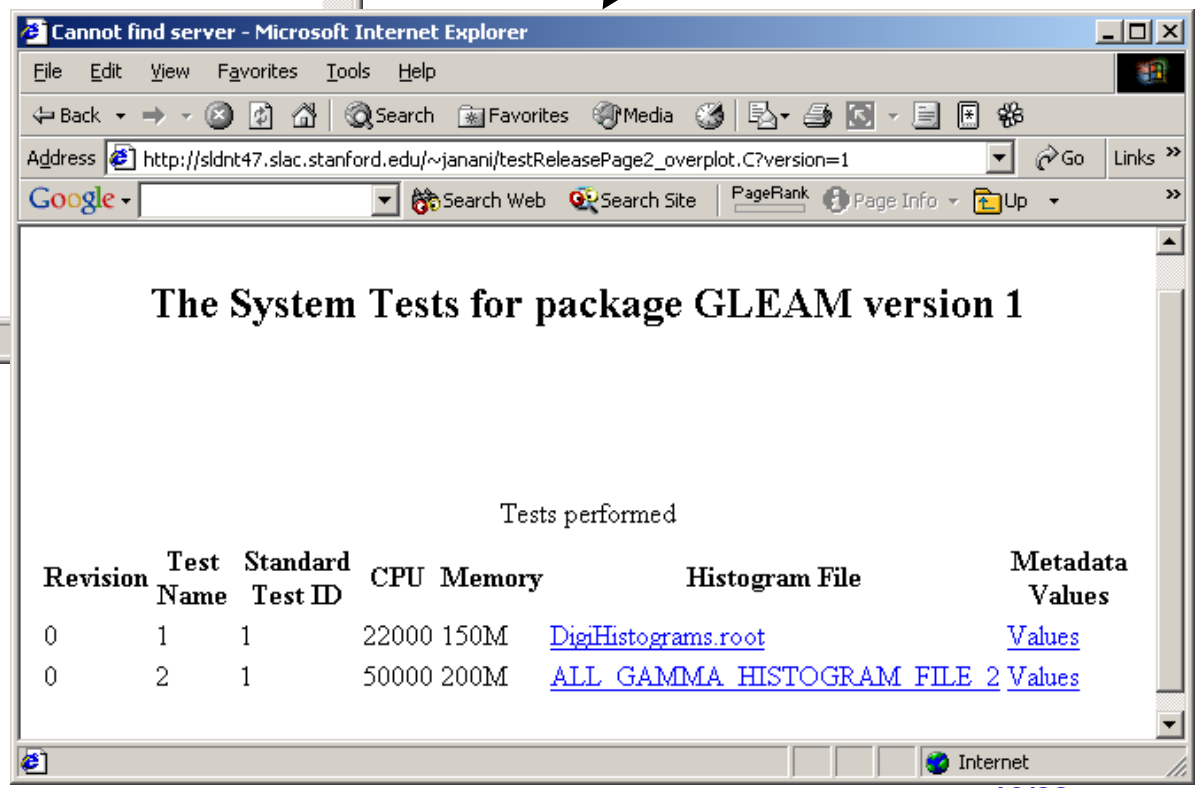
System Tests Database Schema



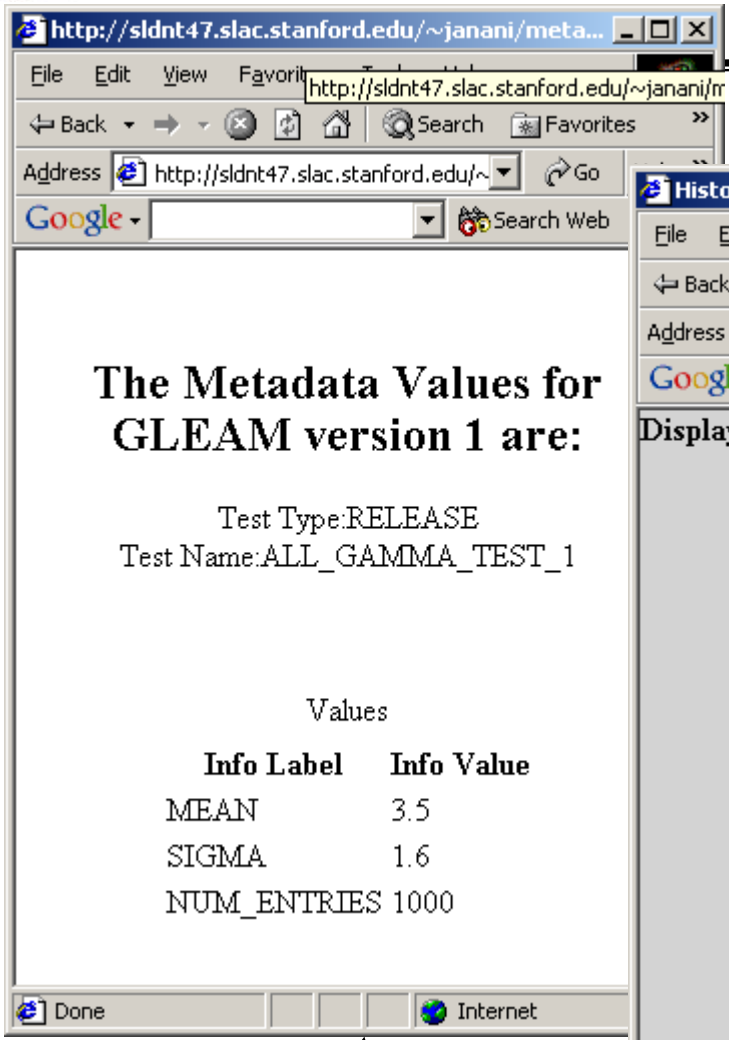
Using Carrot

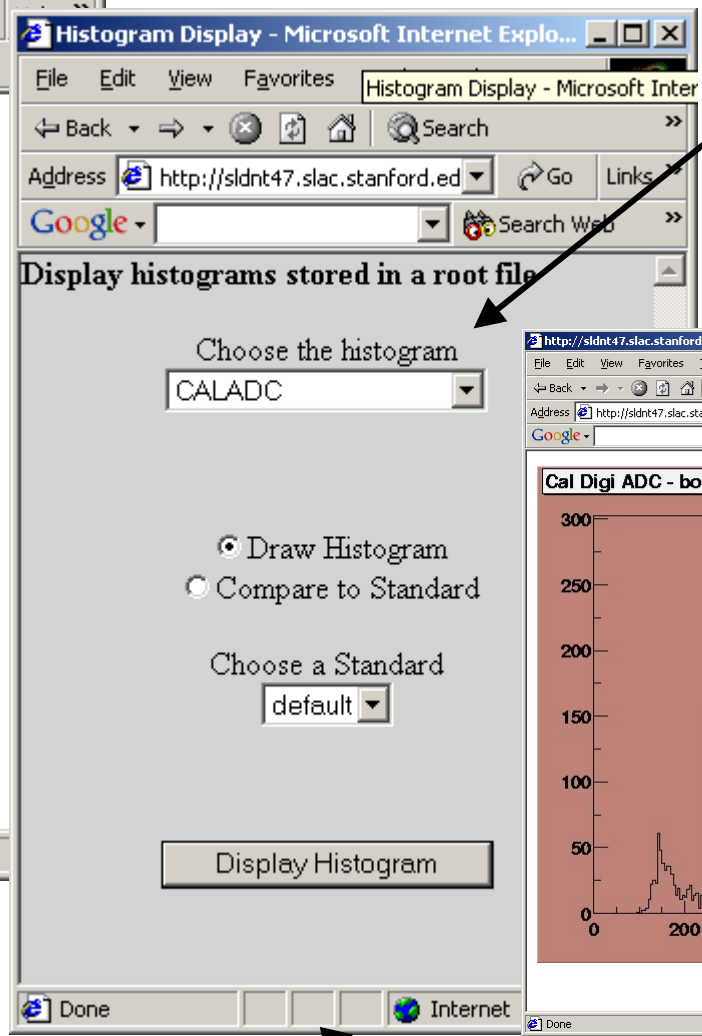
Built using Oracle interface

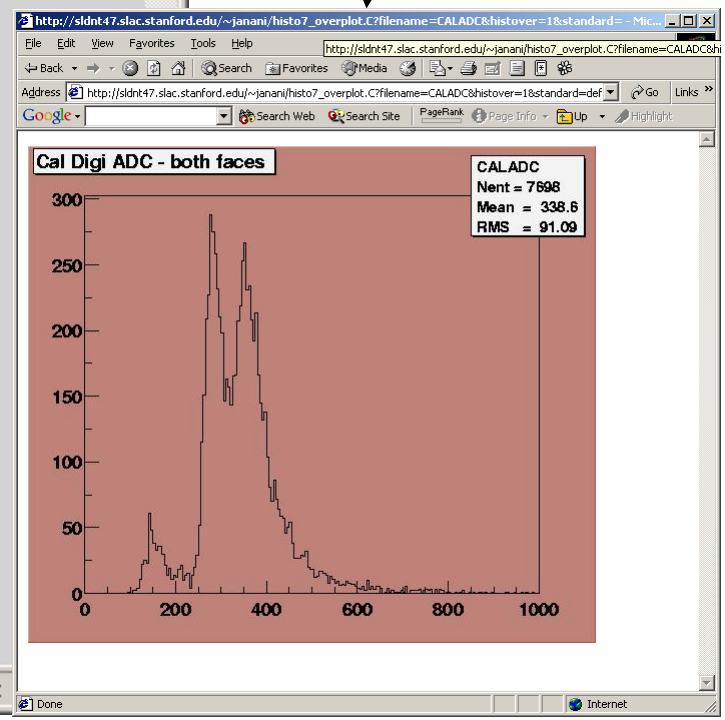




More Carrot



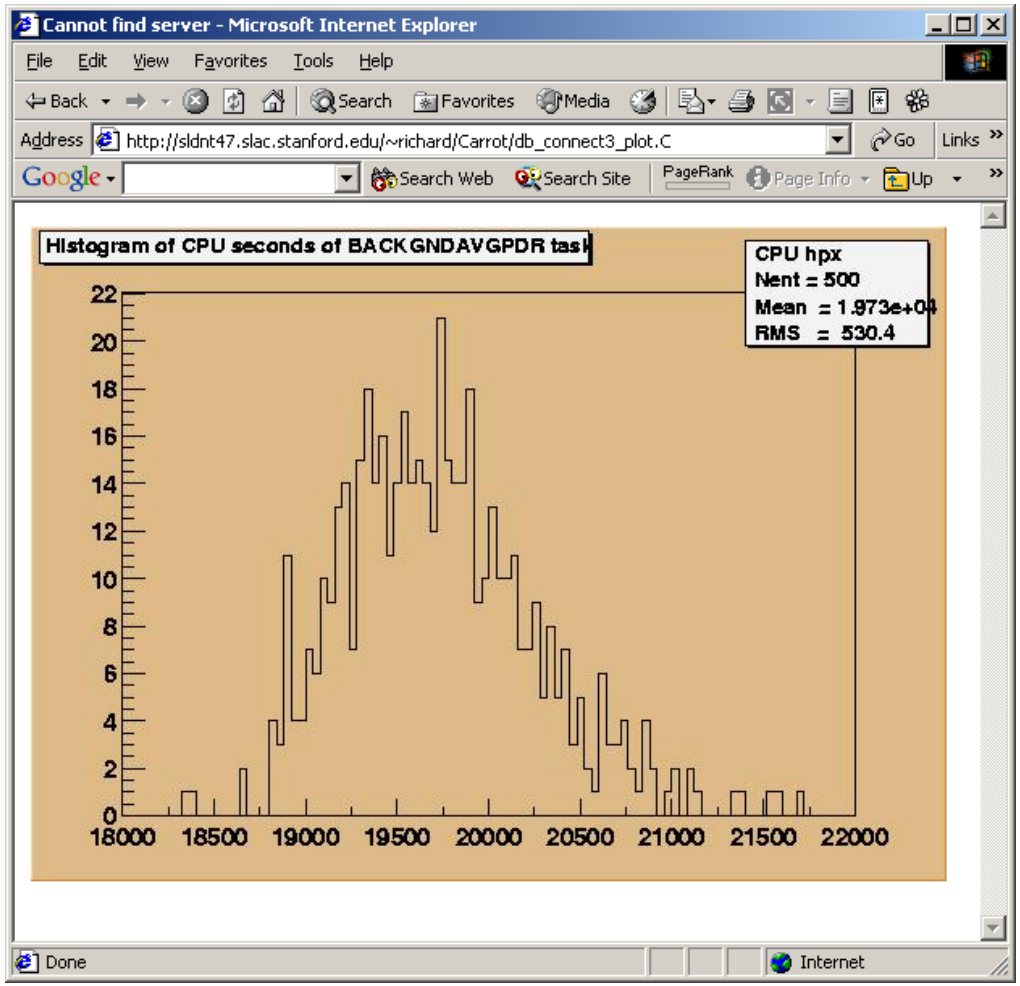




Scan histogram file and plot on demand

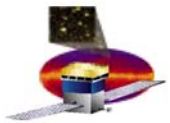
Built using Oracle interface

Even More Carrot



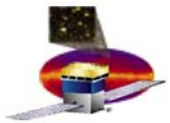
Built using Oracle interface and create plot on the fly

We have demonstrated every use we wanted to have. Now to build the system!



Experiences Developing with Carrot

- It works and the examples are mostly really good!
- BUT
  - Behaviour not always reproducible
    - Page will not initially display, but will upon refresh
  - Have to be careful about server/browser caching
    - Help and examples could be better here
  - Have to think some about the development model
    - Errors are often lost in blank pages
      - Need to use line mode (CINT) to see what's going on
      - A pain to keep toggling the TApache boolean in all macros
      - Somewhat tricky to debug pages that are the result of forms requests – need to get the arg list for the macro as it would appear in the browser
  - Anecdotal evidence of rogue processes
    - Have had to restart the server from time to time



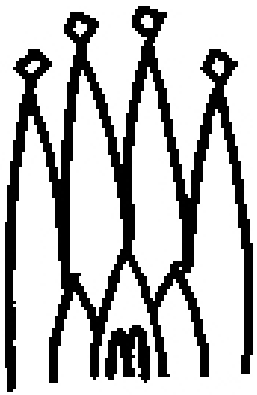
Carrot and Security

- **Met with SLAC Security last week to discuss allowing Carrot through the firewall**
  - **they agreed!**
  - **CERN Security had no opinion**

- **Our agreement**
  - **Inquire with Carrot-central about a possible "CGI-wrap" mechanism to scrub inputs to Carrot**
    - **Plus check inputs to Root macros**
  - **Run on a separate machine with well maintained Apache, sshd etc versions**
  - **Well-defined ownership and oversight of the Root scripts**
  - **Run it as 'nobody' with no write access to the main file systems**

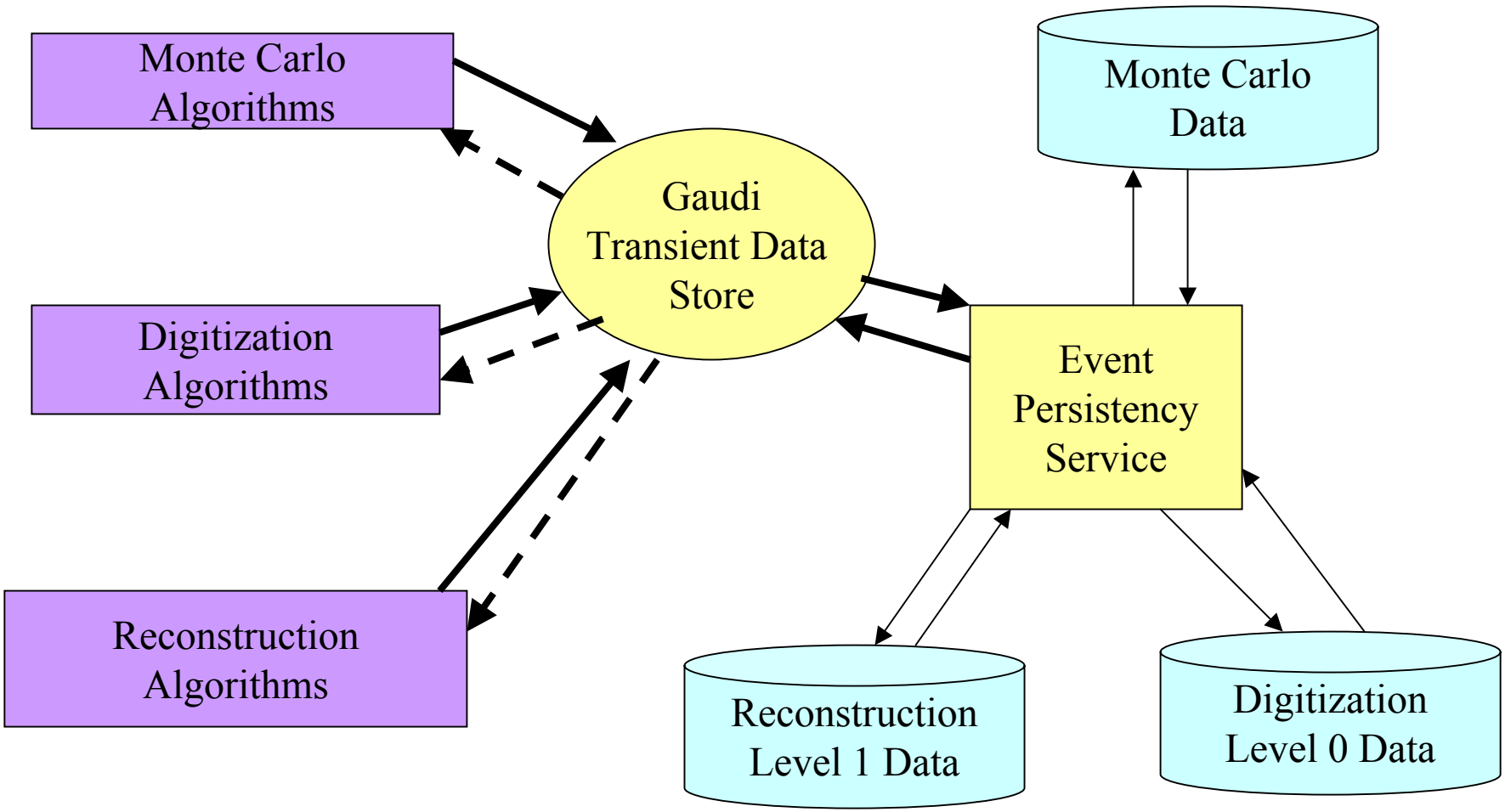
- **They're a bit nervous! "Running C++ scripts in a web server??"**

Gaudi: our framework choice

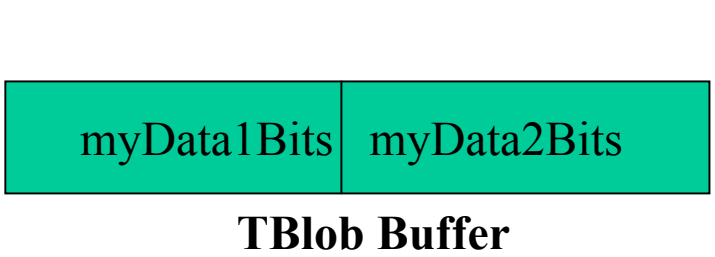


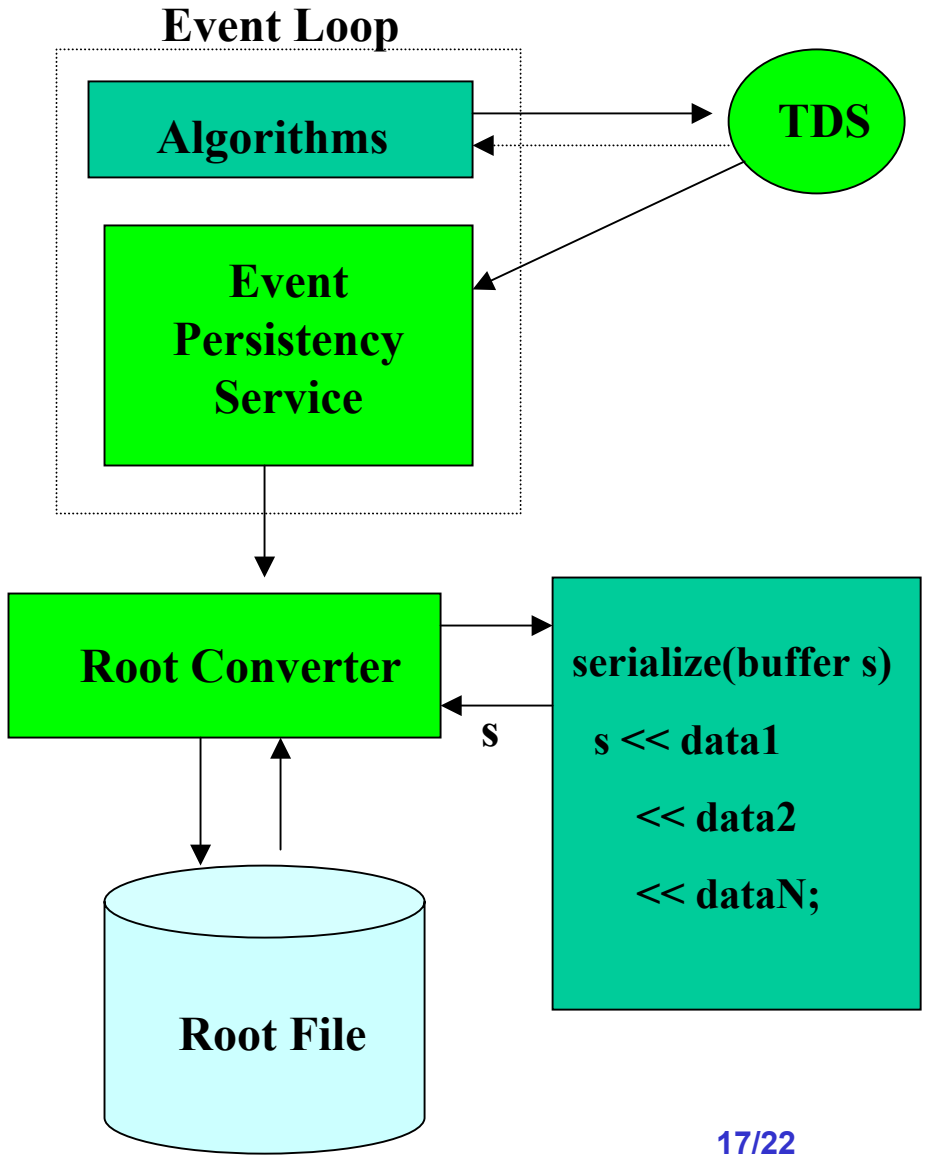
- Open source
- Stable, but active developers, in use by ATLAS, LHCb
- Very good documentation
- All code called via component interfaces:
  - Algorithm
  - Service
- Converter
  - DataObject
- Support for shareables: all code is loaded dynamically
- Job control parameters set in job options file.

Gaudi Persistency Service



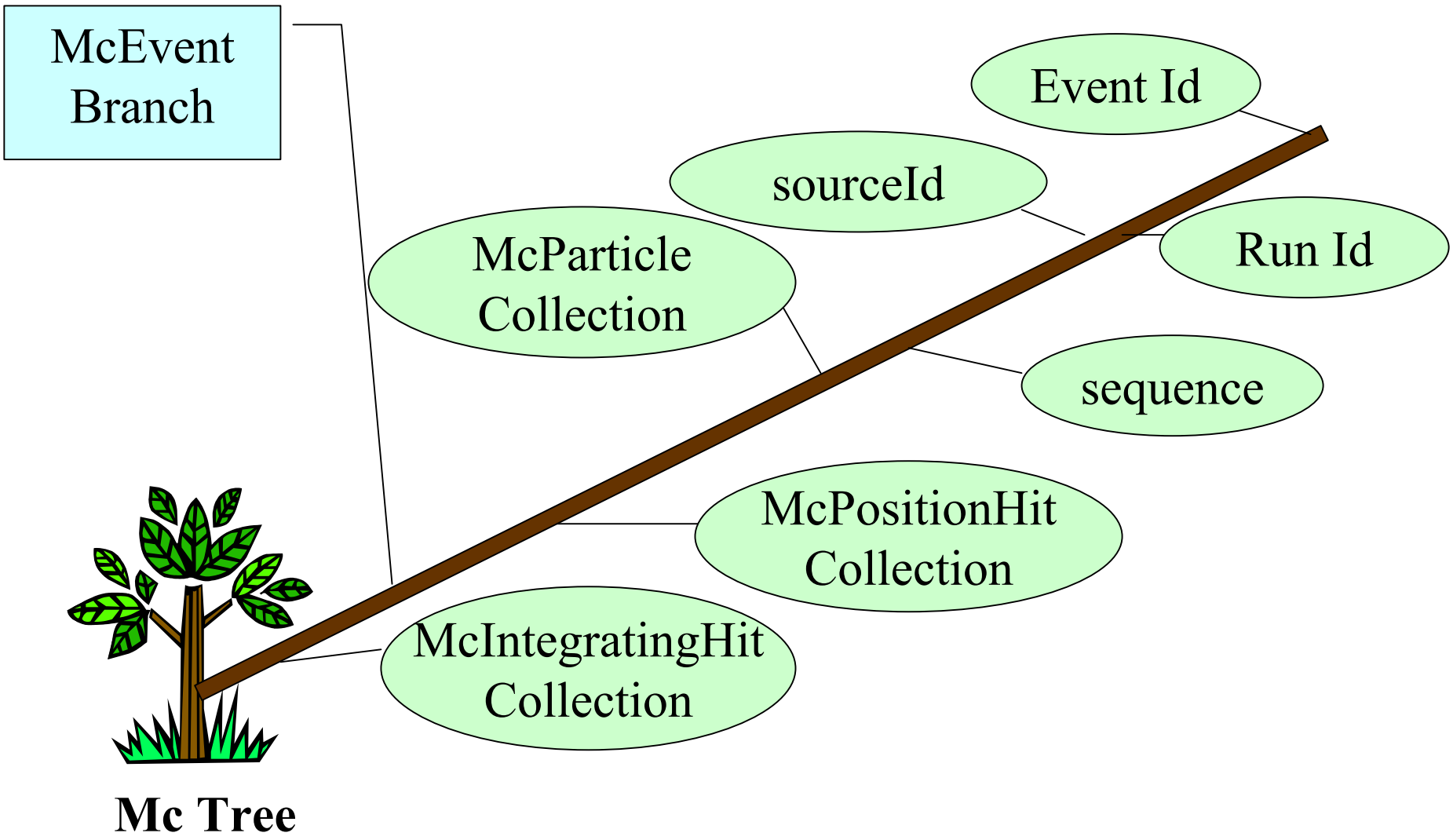
Gaudi ROOT Persistency



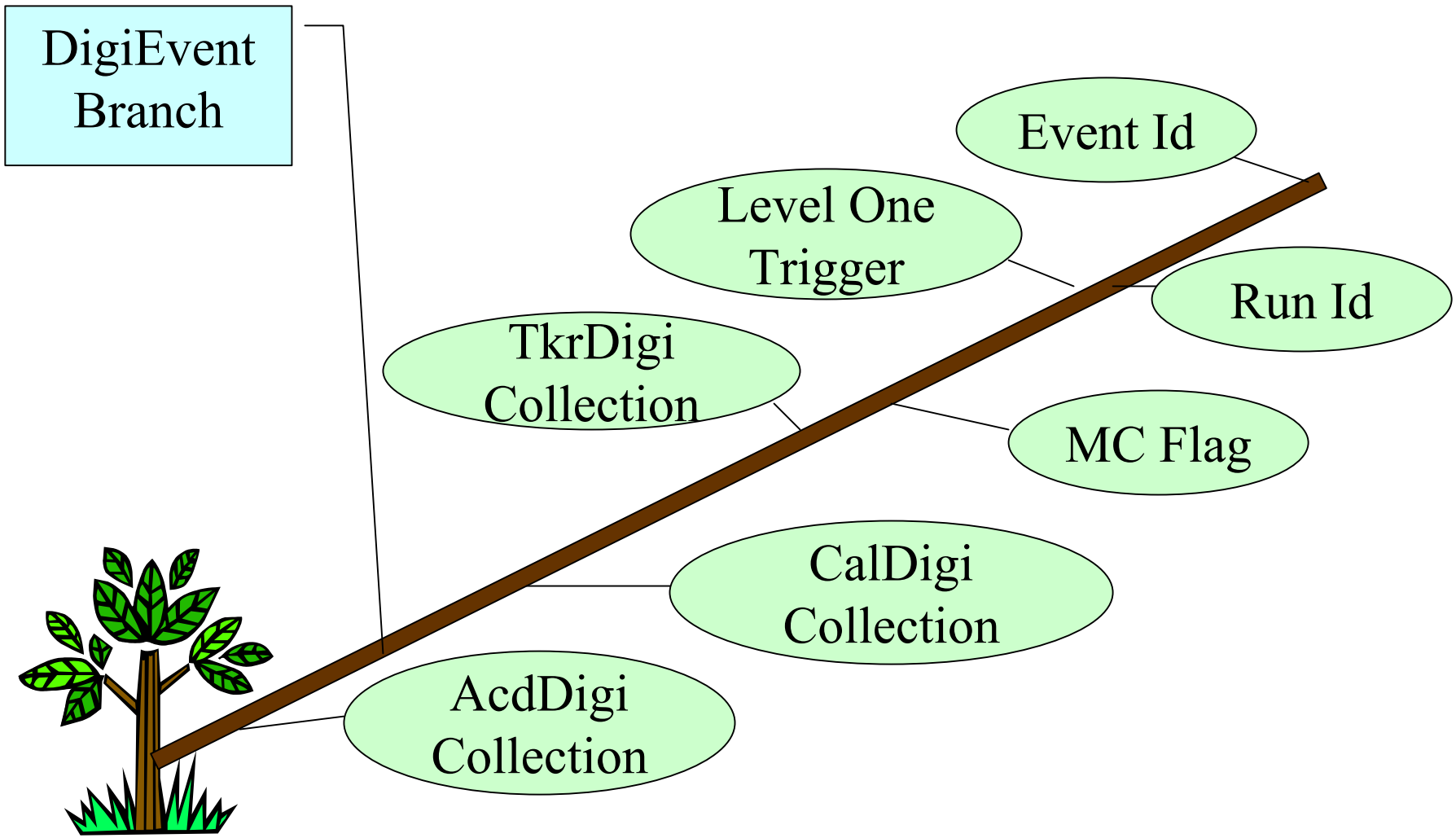


- In order to generalize the writing and reading of data into / out of Root files, a generic data object named TBlob is utilized.
- The TBlob is just a buffer, where the bits of the various data objects is streamed in / out.
- The actual structure of the data objects as they exist in Gaudi is lost.

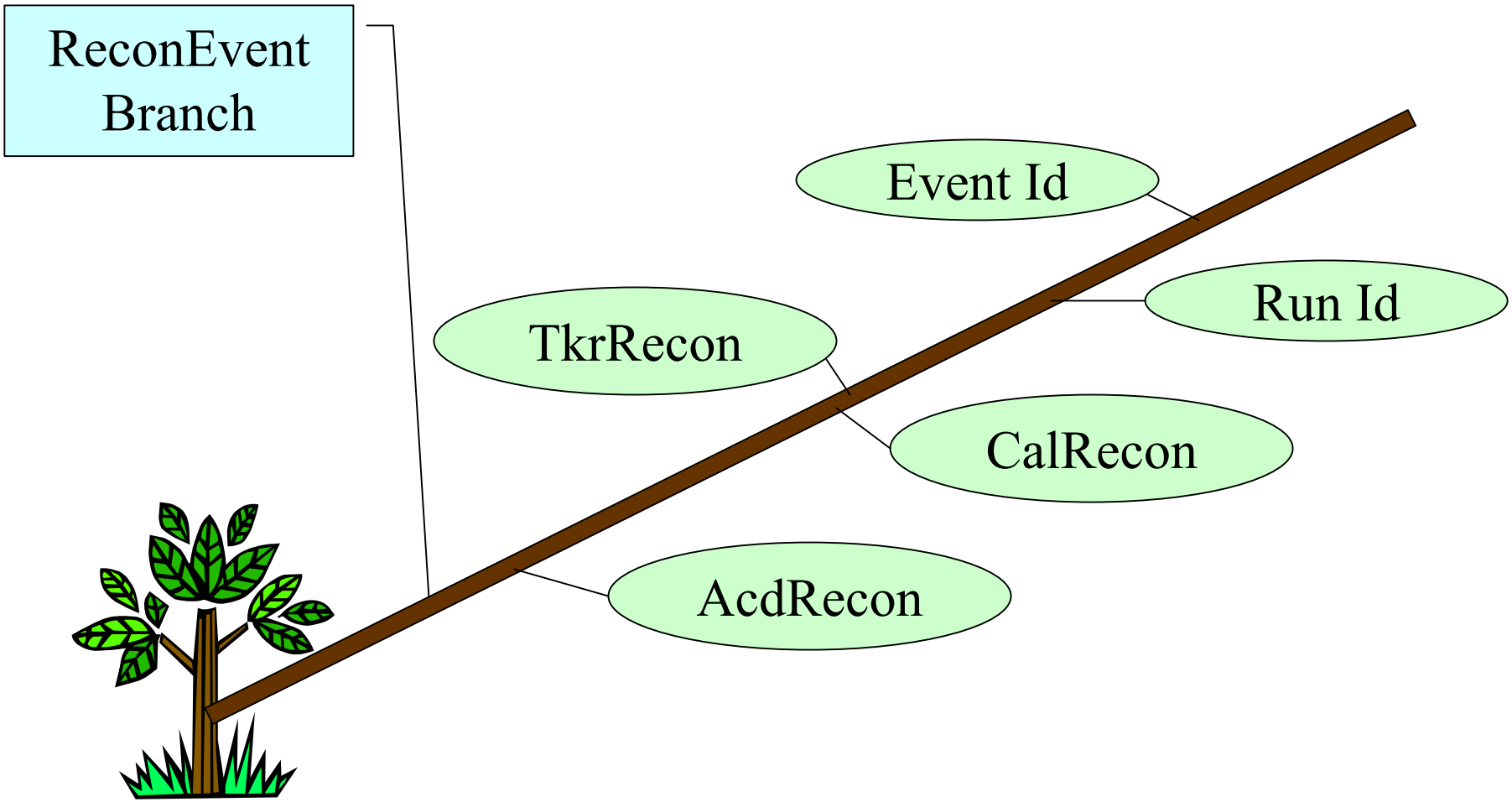
Monte Carlo Data



Digitization Data

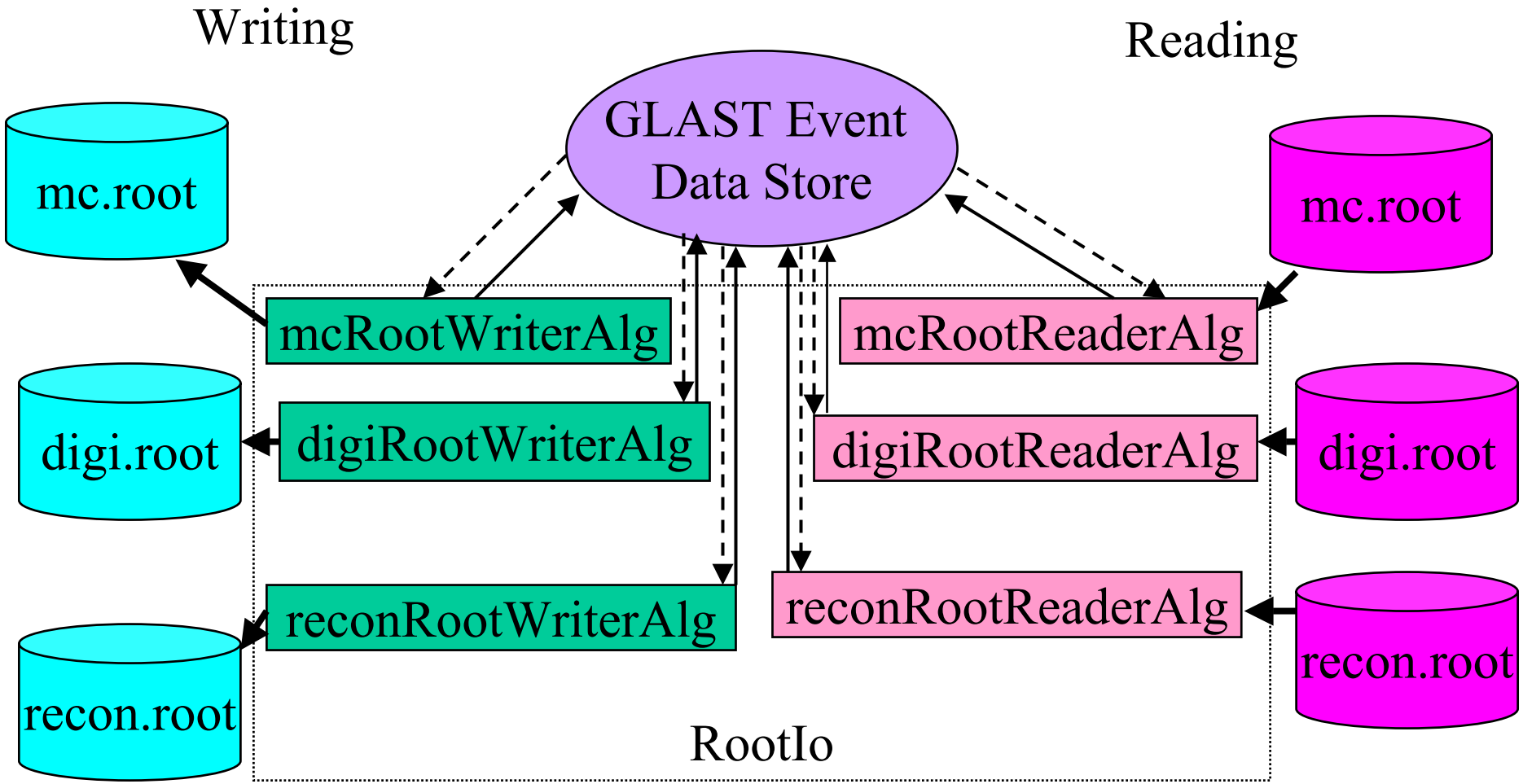


Reconstruction Data



Recon Tree

RootIo – The Rest of the Story



Problems and a Possible Solution

- Use of algorithms is inconsistent with the spirit of Gaudi's Persistency Service.
- Does not provide fine control over what is read/written – it's all or nothing as currently implemented.
- Monolithic algorithms are more difficult to maintain versus light weight converters.

• There is a "real" ROOT service under development
http://www.usatlas.bnl.gov/computing/software/db/rootio.html

-ROOT I/O

-ROOT interactive session by demand

-ROOT share library dynamic loading by demand

-ROOT control over the Gaudi algorithms

• We hope to use this code directly, or modify it for our needs.

Supplemental Slides

Operating Conditions

- Linux Redhat 7.2
- Root 3.02.07
- Carrot 1.0.7
- Oracle
  - interface from GSI
    - http://www.gsi.de/computing/root/OracleAccess.htm
  - Oracle 8

Sample Carrot Script - I

```
Bool_t gApacheExec=1;

#include <TApache.h>
#include <TSQLServer.h>
#include <TSQLRow.h>
#include <TSQLResult.h>
#include <TCanvas.h>
#include <TFrame.h>
#include <TH1.h>

void ReleasePage1_overplot(){

  TApache ap;

  // First we select the package name to display
  char *sql="select version from release_tests where pkg_name = \'GLEAM\'";
  char *connection="xxxxxxxxx";
  char *user="yyyyyyyyy";
  char *pass="zzzzzzzzzz";

  // connect to oracle server
  TSQLServer *db=TSQLServer::Connect(connection, user, pass);

  // Submit the query to the database
  TSQLResult *res=db->Query(sql);
  TSQLRow *row1=res->Next();

  // The HTML form
  gApache->Puts("<HTML><BODY><CENTER>");
  gApache->Puts("<h1>List of Versions of Gleam</h1>");
  gApache->Puts("<h2><BR>Select a Version to view the results of its
tests</h2>");
  gApache->Puts("<FORM METHOD = \"GET\" ACTION =
                \"testReleasePage2_overplot.C\">");
  gApache->Puts("<BR><P><SELECT NAME=\"version\">");
```

```
do{

   // Drop down menu for versions for package GLEAM
   gApache->Puts("<OPTION>");
   package_name = row1->GetField(0);
   ap.Printf(row1->GetField(0));

  }// end while
  while(res->Next());

  gApache->Puts("</SELECT></P>");
  gApache->Puts("<BR><BR><input type=\"submit\" value=\"Submit\">");
  gApache->Puts("<BR></form></center></body></html>");



}// end of ReleasePage1_overplot
```

Creates front page with pulldown menu of code versions

Sample Carrot Script - II

```
Bool_t gApacheExec=1; Bool_t gApacheNoCache=kTrue;

#include <TApache.h>
#include <TCanvas.h>
#include <TH1.h>
#include <THashList.h>
#include <TIterator.h>
#include <TList.h>
#include <TObject.h>
#include <TFile.h>
#include <TRandom.h>
#include <TSystem.h>
#include <TFrame.h>
#include <TString.h>
#include <TKey.h>
#include <string.h>

void histo6_overplot()
{

 TApache ap;

 // Get the systest_id of the test
 TString histfile_T = gSystem->Getenv("hf");
 TString systest_id_T = gSystem->Getenv("systestid");
 TString stdtest_id_T = gSystem->Getenv("stdtestid");
```

```
// Setting up the HTML form
 gApache->Puts("
<html>
 <head>
  <meta HTTP-EQUIV=\"Content-Type\" CONTENT=\"text/html;
                charset=iso8859-1\">
  <meta NAME=\"hsimple7.C\"
                Content=\"Display histograms on a web page\">

  <title>Histogram Display</title>
 </head>

 <body bgcolor=\"LightGrey\" topmargin=\"0\" leftmargin=\"0\"
                marginwidth=\"0\" marginheight=\"0\">
");

 char *hist_file = histfile_T.Data();
 char *systest_id = systest_id_T.Data();
 char *stdtest_id = stdtest_id_T.Data();

 // This is the histogram file of the test
 TFile *histfile = new TFile(hist_file);

 TList *pKeyList = gDirectory->GetListOfKeys();

 TObject *pObj;
 TKey *pKey ;
 TIter it(pKeyList);
```

Creates pulldown menu of histograms

Sample Carrot Script – II (cont)

```
gApache->Puts("<h4>Display histograms stored in a root
file</h4>");
 gApache->Puts("<center><form method=\"get\"
                action=\"histo7_overplot.C\"
name=\"histo4form\">");
 gApache->Puts("<P>Choose the histogram <BR>
                <SELECT NAME=\"filename\">");

 while (pKey = (TKey*)it())   {
    TString pClassName(pKey->GetClassName());
    if(pClassName.BeginsWith("TH1") ||    // just get histos
       pClassName.BeginsWith("TH2") ||
       pClassName.BeginsWith("TH3"))
                {
                 // object associated with the key is read into pObj
                 pObj = pKey->ReadObj();
                 // if it inherits from TH1 it is a histogram
                 if (pObj->InheritsFrom("TH1")) {
                    gApache->Puts("<OPTION>");
                    ap.Printf(((TH1*)pObj)->GetName());
                  }
                }
    else if (pClassName == "TDirectory") {
                 TDirectory *pSubDir = (TDirectory*)pKey->
                                ReadObj();
                 fillListFromDir(pList, pSubDir);
                }// end else
  }// end while
```

```
gApache->Puts("</SELECT></P>");
 gApache->Puts("<BR><BR>");

 // These are the radio buttons to determine whether the histogram
 // should be overplotted with the standard or just drawn
 gApache->Puts("<INPUT TYPE=\"radio\" NAME=\"histover\"
                VALUE=\"1\" CHECKED>");
 gApache->Puts("Draw Histogram");
 gApache->Puts("<BR>");
 gApache->Puts("<INPUT TYPE=\"radio\" NAME=\"histover\"
                VALUE=\"2\" UNCHECKED>");
 gApache->Puts("Compare to Standard");
 gApache->Puts("<BR><BR>");

 // prepare a drop down list of the the versions which are present as
 // standard and specify one value as default

 gApache->Puts("<P>Choose a Standard <BR>
                <SELECT NAME=\"standard\">");
 gApache->Puts("<OPTION>");
 ap.Printf("default");

 // Now for the set of standards
 char *connection="xxxxxxxxxxxx";
 char *user="yyyyyyyyy";
 char *pass="zzzzzzzzzzzz";

 char *sql = "select stdt.standardtest_id from system_tests syst,
                standard_tests stdt where stdt.systest_id =
                syst.systest_id";
```

Sample Carrot Script – II (cont)

```
// connect to oracle server
  TSQLServer *db=TSQLServer::Connect(connection, user, pass);
  TSQLResult *res=db->Query(sql);
  TSQLRow *row1=res->Next();

  // Display the list of standard tests available
  do{

    gApache->Puts("<OPTION>");
    ap.Printf(row1->GetField(0));

  }while(res->Next());

  gApache->Puts("</SELECT></P>");
  gApache->Puts("<BR><BR>");


  // This is to pass the name of the histogram file as a hidden value
  char *hiddenvalue1 = "<INPUT TYPE = \"hidden\"
                  NAME = \"histfilename\" VALUE = \"";
  char *hiddenvalue2 ="\">";

  char *histfilename = new char[200];
  std::strcpy(histfilename, hiddenvalue1);
  std::strcat(histfilename, hist_file);
  std::strcat(histfilename, hiddenvalue2);

  // This is to pass the id of the standard test as a hidden value
  gApache->Puts(histfilename);
```

```
  char *hiddenvalue3 = "<INPUT TYPE = \"hidden\" NAME =
                  \"stdtestid\" VALUE = \"";
  char *hiddenvalue2 ="\">";
  char *standard = new char[200];
  std::strcpy(standard, hiddenvalue3);
  std::strcat(standard, stdtest_id);
  std::strcat(standard, hiddenvalue2);

  gApache->Puts(standard);

  gApache->Puts("<input type=\"submit\" value=
                  \"Display Histogram\"><BR></center></form>
                  </body></html>");


  delete[] histfilename;
  delete[] standard;


}// end void histo6_overplot
```