Suez and the Event Display

Jim Pivarski

Cornell University

8 June, 2006
Outline for this talk

- Overview of Suez
- Demonstration of Suez with the Event Display
- Raw data versus Pass2
- Walk-through of creating a processor for analysis
- Filling histograms
- Homework: repeat this presentation!
Suez (where CLEOpatra lives)

Software framework for accessing CLEO data

Generalized ‘for’ loop:
- you select data for processing
- select processors to perform operations on events
- type ‘go’
- suez loops over events, applying requested operations

You can write your own processors (we will today)
This means a text editor will be your dearest friend though some processors have GUI interfaces

- Event Display
- HistogramViewer
Suez data-access model

Data which is valid for a given event is accessible through the Frame (like a movie frame), and obsolete data is inaccessible.
CMS, an LHC Detector Experiment

Follows the same model (thanks to Chris Jones)
Suez data processing model

**processors** are called in the event loop, extract data from the Frame, fill histograms, and filter events.

**producers** are called when data is requested, extract what they need, and insert the desired results into the Frame.

**modules** are the most general; we usually use them to read data from disk (EventStoreModule) and manage histogram output (RootHistogramModule).

![Diagram of Suez data processing model](image-url)
A typical tcl (suez control file)

- module GoGetDataModule
- prod sel CalibrateDataProd  
- prod sel IdentifyTracksProd
- proc sel FindGlueballsProc
- proc sel MakePlotsOfGlueballsProc
- go

Later, you can run MakePlotsOfGlueballsProc on your skim file.
A typical tcl (suez control file)

- module GoGetDataModule
- prod sel CalibrateDataProd
- prod sel IdentifyTracksProd
- proc sel FindGlueballsProc
- file out ToSkimFile
- go

Later, you can run MakePlotsOfGlueballsProc on your skim file
Demonstration: The Event Display

The Event Display is a set of processors that draw CLEO data on the screen, event by event. It’s useful for

- Data sanity check (e.g. during data-taking)
- Identifying backgrounds
- **Introduction to CLEO/data analysis**
Setup (review of yesterday)

Connect to a Linux computer

For **bash/sh** users (type "echo $SHELL" to identify),

```
▶ . /nfs/cleo3/Offline/scripts/cleo3logins
▶ . /nfs/cleo3/Offline/scripts/cleo3defs
▶ c3rel 20060224_FULL_2
▶ export USER_SRC=$HOME/my_src
▶ export USER_BUILD=/cdat/tem/mccann/build/$C3LIB
▶ export USER_SHLIB=$USER_BUILD/Linux/shlib
▶ c3rel $C3LIB  
```

(Yes, again!)

Be sure to make the following directories (all shells):

```
▶ mkdir -p $HOME/my_src
▶ mkdir -p $HOME/my_tcl
```
Setup (review of yesterday)

Connect to a Linux computer

For `tcsh/csh` users (type “echo $SHELL” to identify),

- source /nfs/cleo3/Offline/scripts/cleo3def
- c3rel 20060224_FULL_2
- setenv USER_SRC $HOME/my_src
- setenv USER_BUILD /cdat/tem/mccann/build/$C3LIB
- setenv USER_SHLIB $USER_BUILD/Linux/shlib
- c3rel $C3LIB

(Yes, again!)

Be sure to make the following directories (all shells):

- mkdir -p $HOME/my_src
- mkdir -p $HOME/my_tcl
Create a tcl file

▶ cd $HOME/my_tcl
▶ favorite_text_editor hitsandeverything.tcl &

Fill it with the following:

```tcl
# Get raw events so we can look at hits
module sel EventStoreModule
eventstore in 20060601 daq all runs 205507 217380

# All the things you need to process raw data
run_file $env(C3_SCRIPTS)/getNewestConstants.tcl
run_file $env(C3_SCRIPTS)/trackingDataFull.tcl
run_file $env(C3_SCRIPTS)/CcP2.tcl

# Run the event display
run_file $env(C3_SCRIPTS)/view_command.tcl
view -display_only ShowerAttributes DRHits ZDHits SeedTrack
    KinePionFit DBEventHeader StandAloneGeom

go
```
Run Suez

▶ suez -f hitsandeverything.tcl
Things to do with the Event Display

1. Move windows within windows, zoom around, look at side view, select hits

2. Step through events ("Continue" button); zip through them ("Auto Advance" button)

3. SeedTracks (green) versus fitted tracks (KinePion, pink)

4. Calorimeter shower representation: label with energy

5. Make DR hits circles

6. Make ZD hits lines
The Drift Chamber

Wires strung between two endplates report time of hit which yields distance of closest approach of charged particle
The Drift Chamber

field wires (in perspective)

circle of closest approach
electrons drift and multiply
sense wire

Particle trajectory

1.4 cm

y
x closest
approach
circle of
particle trajectory
sense wire
field wires

16/29
The ZD

A small drift chamber with an extreme angle between the wire end positions (stereo angle)
Using tilted wires to obtain \( z \) information in the outer drift chamber/ZD. Tilted wires extrude lines in the \( x-y \) projection; position along a tilted wire indicates the \( z \) of the track helix near that wire. The closest wires to the track (in three dimensions) are highlighted.
Cheat Sheet for 4, 5, and 6

View menu → Set 2D Representation...
Pre-processed data: pass2

> cd $HOME/my_tcl
> favorite_text_editor afterpass2.tcl &

# Get pass2'ed events (no hits, but much faster!)
module sel EventStoreModule
eventstore in 20050316 physics all

# Setup standard analysis and event display
setup_analysis

# Run the event display
run_file $env(C3_SCRIPTS)/view_command.tcl
view -display_only Pass2

gob
> suez -f afterpass2.tcl

1. Note the missing hits
2. See what happens when you press “Auto Advance”
First Data Analysis

Many events look like this:

- 2 tracks
- minimal calorimeter energy \((E/p \lesssim 10\%)\)
- misses the beamspot
Select Cosmic Rays

We will make a processor that identifies cosmic rays

- cd $HOME/my_src
- mkproc -track -shower -histogram MySecondProcessor
- favorite_text_editor MySecondProcessor/Class/MySecondProcessor.cc

The processor is already filled with a lot of example code. We'll ignore that and add the following block at the beginning of the MySecondProcessor::event method (line 153). Before editing, the code looks like this:

```cpp
ActionBase::ActionResult
MySecondProcessor::event( Frame& iFrame ) // anal3 equiv.
{
    report( DEBUG, kFacilityString ) << ¨here in event()¨ << endl;

    // insert new code here

    // Create a table of tracks and fill it.
    FATable< NavTrack > trackTable;
    extract( iFrame.record( Stream::kEvent ), trackTable );
```
int number_of_tracks = 0;
double closest_to_beamline = 1000.; // meters

FATable<NavTrack> tracks;
extract(iFrame.record(Stream::kEvent), tracks);

for (FATable<NavTrack>::const_iterator track = tracks.begin();
    track != tracks.end(); ++track) {

    double distance_from_beamline = fabs(track->pionHelix()->d0());
    if (distance_from_beamline < closest_to_beamline) {
        closest_to_beamline = distance_from_beamline;
    }

    double track_momentum = track->pionFit()->momentum().mag();
    if (track_momentum > 1.) {
        number_of_tracks++;
    }
}
FATable<NavShower> showers;
extract(iFrame.record(Stream::kEvent), showers);

double biggest_shower_energy = 0.; // GeV

if (showers.size() > 0)
{
    // the showers table is sorted by energy
    biggest_shower_energy = showers.begin()->attributes().energy();
}

// Now filter the events
if (number_of_tracks == 2 &&
    closest_to_beamline > 0.05 &&
    biggest_shower_energy < 0.3) // less than 300 MeV
{
    return ActionBase::kPassed;
}
else
{
    return ActionBase::kFailed;
}
Compile and Run

- c3make
- cd $HOME/my.tcl
- favorite_text_editor afterpass2.tcl &

In `afterpass2.tcl`, add the processor after setting up the view command but before calling it:

```tcl
run_file $env(C3_SCRIPTS)/view_command.tcl
proc sel MySecondProcessor
view -display_only Pass2
```

Run `suez`

- `suez -f afterpass2.tcl`

and press the “Auto Advance” button in the Event Display. See how you have biased the events!
Histogram track $\phi$

1. In $\$HOME/my\_src/MySecondProcessor/MySecondProcessor/MySecondProcessor.h$, add
   
   ```c++
   HIHist1D* m_histphi; after HIHist1D* m_histo1;
   ```

2. In $\$HOME/my\_src/MySecondProcessor/Class/MySecondProcessor.cc$, add
   ```c++
   m_histphi = iHistoManager.histogram(¨phi¨, 100, 0., 2.*M_PI); at the end of MySecondProcessor::hist_book(){ } (line 144)
   ```

3. Also add the following just before your return ActionBase::kPassed; (line 181)
   ```c++
   for (FATable<NavTrack>::const_iterator track =
            tracks.begin(); track != tracks.end(); ++track)
   {
      m_histphi->fill(track->pionHelix()->phi0());
   }
   ```

4. Recompile (described on previous page)
5. At the beginning of $HOME/my_tcl/afterpass2.tcl, add

```tcl
# Load a histogram manager.
module sel HbookHistogramModule
hbook file myhistograms.rzn
hbook init
```

6. Also add `proc sel HistogramViewerProc after` 
```tcl
proc sel MySecondProcessor.
```

7. Run `suez -f afterpass2.tcl` in your $HOME/my_tcl directory.

8. Select Root → MySecondProcessor → phi from the heirarchy on the left of the HistogramViewer window

9. Click “Continue” HistogramViewer and “Auto Advance” in the Event Display
Homework

1. Walk through the demonstrations on your own.
2. Study $e^+e^- \rightarrow \mu^+\mu^-$: plot the cos $\theta$ distribution.

Hints:
- Collision-borne muons deposit the same energy in calorimeter showers as cosmic ray muons.
- But $e^+e^- \rightarrow \mu^+\mu^-$ events must come from the beam.
- Given a FATable<NavTrack>::const_iterator track, the momentum vector components are obtained by track->pionFit()->momentum().x(), .y(), and .z().
- $\theta$ is the polar angle between $\sqrt{x^2 + y^2}$ and z.
- p. 137 Peskin & Schroeder: $P(\cos \theta) \propto 1 + \cos^2 \theta$
Shopping for data

Very important webpages/directories

▶ Member functions:
  ▶ http://www.lns.cornell.edu/restricted/webtools/doxygen/
    Offline/html/hierarchy.html

▶ Looking at the code:
  ▶ http://www.lns.cornell.edu/restricted/webtools/cleo3/
  ▶ ls $C3_CVSSRC/
  ▶ ls $C3_OTHER/

▶ Producers to add to your .tcl:
  ▶ http://www.lns.cornell.edu/~cleo3/current/data/
    proxiesOfProducers.txt

▶ Event classification:
  ▶ http://www.lns.cornell.edu/restricted/CLEO/CLEO3/soft/hints/
    CLEOIIIEventClassificationDescription.html