

Setting Up a C++ Analyzer Replay

Ole Hansen

Jefferson Lab

Joint Hall A & Hall C Data Analysis Workshop
January 14, 2015

Brief Introduction

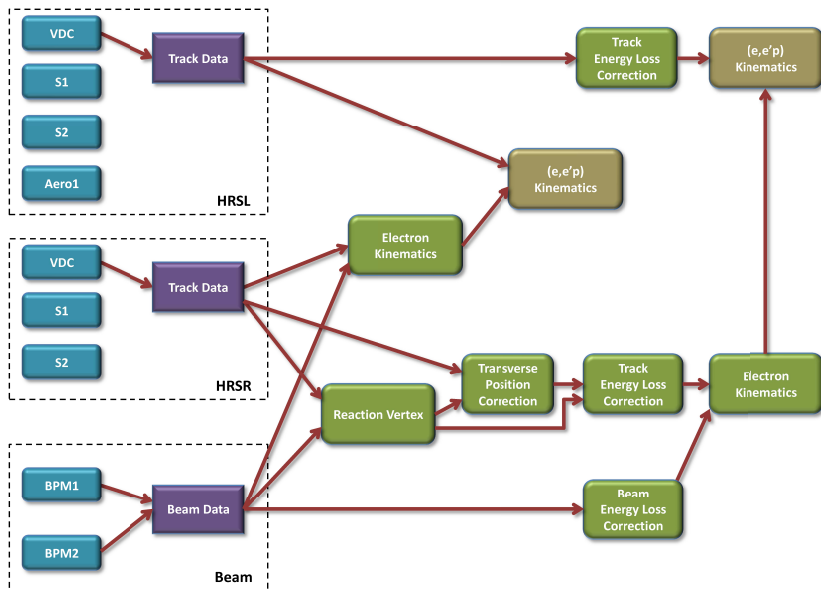
Analyzer Concepts: Analysis Objects

- Any module that produces “results”
- Every analysis object has **unique name**, e.g. **R.s1**
- Results stored in **“global variables”**, prefixed with the respective module’s name, e.g. **R.s1.nhits**

Types of Analysis Objects

- “Detector”
 - ▶ Code/data for analyzing a **type** of detector.
Examples: Scintillator, Cherenkov, VDC, BPM
 - ▶ Typically embedded in an Apparatus
- “Apparatus” / “Spectrometer”
 - ▶ Collection of Detectors
 - ▶ Combines data from detectors
 - ▶ “**Spectrometer**”: Apparatus with support for **tracks**
- “Physics Module”
 - ▶ Combines data from several apparatuses
 - ▶ Typical applications: **kinematics calculations, vertex finding, coincidence time extraction**
 - ▶ Toolbox design: Modules can be chained, combined, used as needed

A (complex) Module Configuration Example



Tutorial (Hall A-Specific)

Getting The Software

- 1 Set up **ROOT**
- 2 Download the **analyzer source code**
- 3 Unpack and **build**
- 4 To install, simply **set environment variables**

Setting Up The Software on JLab CUE

```
ifarm1102> source /apps/root/PRO/setroot_CUE
ifarm1102> wget http://hallaweb.jlab.org/podd/download/analyzer-1.5.28.tar.gz
ifarm1102> tar xzf analyzer-1.5.28.tar.gz
ifarm1102> cd analyzer-1.5.28
ifarm1102> make -j
ifarm1102> setenv PATH ${PWD}:${PATH}
ifarm1102> setenv LD_LIBRARY_PATH ${PWD}:${LD_LIBRARY_PATH}
ifarm1102> analyzer
analyzer [0]
```

More details on the web [► docs](#)

Things You'll Need

① Replay script

- ▶ Defines detectors/apparatuses to be analyzed, kinematics, calculations to be done, file locations, tree variable names etc.
- ▶ Many examples available from previous experiments
- ▶ Simple or fancy `▶ fancy example`. Try to start out simple
- ▶ May be compiled

② Set of database files

- ▶ Usually one file per detector, `db_<name>.dat`
- ▶ Run database, `db_run.dat`, defines beam energy, spectrometer angles
- ▶ `db_cratemap.dat` and `scaler.map`, define decoder parameters
→ get these files from DAQ expert

③ Output definition file

- ▶ Defines which variables to write to the tree in the output ROOT file

④ Raw data (CODA file)

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."));
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."));
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all events in the input
```

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all events in the input
```

Note: Modules are set up by including them in analysis lists

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all events in the input
```

Note: module names → prefix for database keys & global variables

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all events in the input
```

Note: HRS already contains "vdc", "s1" and "s2" detectors

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run);              // Process all events in the input
```

Note: Choosing output definitions

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all events in the input
```

Note: Module chaining

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all invents in the input
```

Note: Setting target mass parameter in script, overrides run database

Example Replay Script

```
// Set up left arm HRS with the detectors we're interested in
THaHRS* HRSL = new THaHRS("L", "Left HRS"); // NB: VDC/s1/s2 already included
HRSL->AddDetector( new THaCherenkov("cer", "Gas Cherenkov counter" ));
HRSL->AddDetector( new THaShower("ps", "Pre-shower pion rej."););
HRSL->AddDetector( new THaShower("sh", "Shower pion rej."););
gHaApps->Add(HRSL);

// Ideal beam (perfect normal incidence and centering)
THaIdealBeam* ib = new THaIdealBeam("IB", "Ideal beam");
gHaApps->Add(ib);

// Simple kinematics and vertex calculations
Double_t mass_tg = 12*931.494e-3; // C12 target
THaPhysicsModule *ekine, *rpl, *Lgold;
ekine = new THaElectronKine( "L.ekine", "Electron kinematics L", "L", "IB", mass_tg );
rpl   = new THaReactionPoint( "rpl", "Reaction vertex L", "L", "IB" );
Lgold = new THaGoldenTrack( "L.gold", "LHRS golden track", "L" );
gHaPhysics->Add(ekine);
gHaPhysics->Add(rpl);
gHaPhysics->Add(Lgold);

// The CODA data file we want to replay
THaRun* run = new THaRun("/rawdata/run_12345.dat");

// Set up and run standard analyzer (event loop)
THaAnalyzer* analyzer = new THaAnalyzer;
analyzer->SetOutFile( "/bigdisk/run_12345.root" );
analyzer->SetOdefFile("HRSL.odef"); // Define output
analyzer->Process(run); // Process all events in the input
```

More complex code usually needed to deal with file locations & split runs

Example Database

```
[ tutorial ]$ ls -FR DB
DB:
20120306/  db_cratemap.dat  db_run.dat  scaler.map

DB/20120306:
db_L.s1.dat  db_L.s2.dat  db_L.vdc.dat
```

- Recommended to set **DB_DIR** to point to top-level database directory
- Database files in **current directory** always take precedence
- Usually only detectors require databases (for the detector map).
Physics modules sometimes read the run database (for masses, angles)
- Contents of files depends on corresponding module. (Will switch to consistent **key/value format** with version 1.6)
- Documentation on the web [▶ docs](#)
- Hall C modules use Hall C-style parameter files

Example Output Definition File

```
# All variables from the GoldenTrack module
block L.gold.*

# Calculated quantities for inclusive electron scattering measured
# by the LHRS, from the ElectronKine physics module
block L.ekine.*

# All LHRS track data (focal plane as well as reconstructed to target)
block L.tr.*
```

- Much more possible
 - ▶ Arithmetic expressions
 - ▶ Using/defining cuts
 - ▶ 1D and 2D histograms
 - ▶ EPICS variables
 - ▶ Scalers
- Full documentation on the web [▶ docs](#) (Bob Michaels)

FAQ: Help! Where Do I Find Those Variable Names?

Options:

- Inspect each module's **DefineVariables** function
- **Init()** analyzer (instead of **Process()**), then print variable list.
May use **wildcards** to select subsets.

Variable List Printout (from tutorial, see later)

```
analyzer [0] .x init.C
analyzer [1] gHavars->Print("", "L.ekine.*")
Collection name='THaVarList', class='THaVarList', size=203
OBJ: THaVar    L.ekine.Q2      4-momentum transfer squared (GeV^2)
OBJ: THaVar    L.ekine.omega   Energy transfer (GeV)
OBJ: THaVar    L.ekine.W2      Invariant mass of recoil system (GeV^2)
OBJ: THaVar    L.ekine.x_bj     Bjorken x
OBJ: THaVar    L.ekine.angle   Scattering angle (rad)
OBJ: THaVar    L.ekine.epsilon Virtual photon polarization factor
OBJ: THaVar    L.ekine.q3m     Magnitude of 3-momentum transfer
OBJ: THaVar    L.ekine.th_q    Theta of 3-momentum vector (rad)
OBJ: THaVar    L.ekine.ph_q    Phi of 3-momentum vector (rad)
OBJ: THaVar    L.ekine.nu      Energy transfer (GeV)
OBJ: THaVar    L.ekine.q_x     x-cmp of Photon vector in the lab
OBJ: THaVar    L.ekine.q_y     y-cmp of Photon vector in the lab
OBJ: THaVar    L.ekine.q_z     z-cmp of Photon vector in the lab
...
```

Other Useful Things You Might Need

- ❶ Cut/test definition file
 - ▶ Defines logical tests to be evaluated at various analysis stages
 - ▶ Special “master” tests at each stage can be used as cuts to reject certain events
- ❷ Environment setup script
 - ▶ Shell script to set up environment variables for replay
- ❸ Calibration scripts
 - ▶ Special replay scripts
 - ▶ Some standardized (VDC time offsets), mostly experiment-specific
 - ▶ This is where the real work begins
- ❹ Disk space!
 - ▶ Output ROOT files tend to be big, and numerous
 - ▶ Do not write production output to a home directory!
- ❺ Software Development Kit
 - ▶ Skeleton classes for rapid development of your own code
 - ▶ Largely self-documenting [▶ download](#)

Using The Tutorial Files

Example g2p optics replay setup [▶ download](#)

Setting Up And Running the Tutorial on JLab CUE

```
ifarm> source /apps/root/PRO/setroot_CUE
ifarm> wget http://hallaweb.jlab.org/data_reduc/AnaWork2015/tutorial-jan15.tar.gz
ifarm> tar xzf tutorial-jan15.tar.gz
ifarm> cd tutorial
ifarm> tar xzf analyzer-1.5.28.tar.gz
ifarm> cd analyzer-1.5.28
ifarm> make -j
ifarm> cd ..
ifarm> source setup.csh
ifarm> ln -s /work/halla/g2p/disk1/ole/g2p_3132.dat.0 data/
ifarm> cd replay
ifarm> analyzer
analyzer [0] .x replay.C
analyzer [1] .x plot.C
```

See the included README file for more info