Hall A Software & Analysis Technical Details

Ole Hansen

Jefferson Lab

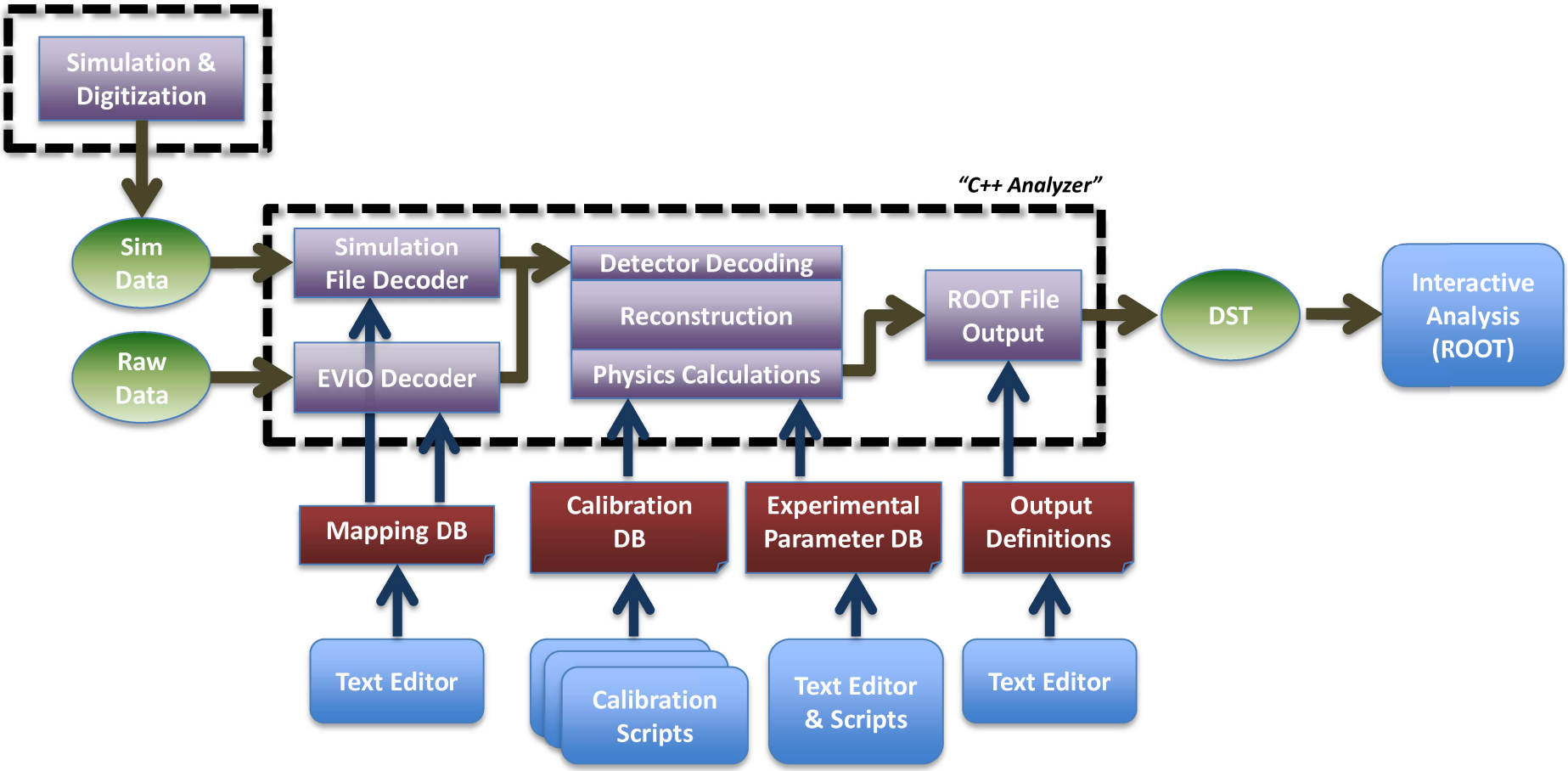
JLab 12 GeV Software Review
Afternoon Session
June 7, 2012

Jefferson Lab
Thomas Jefferson National Accelerator Facility

Contents

1. Main Reconstruction Software
2. Specific Analysis Algorithms
3. Calibrations
4. Simulations
5. Data Quality Monitoring
6. Code Management
7. Potential Concerns

Hall A Data & Analysis Flow

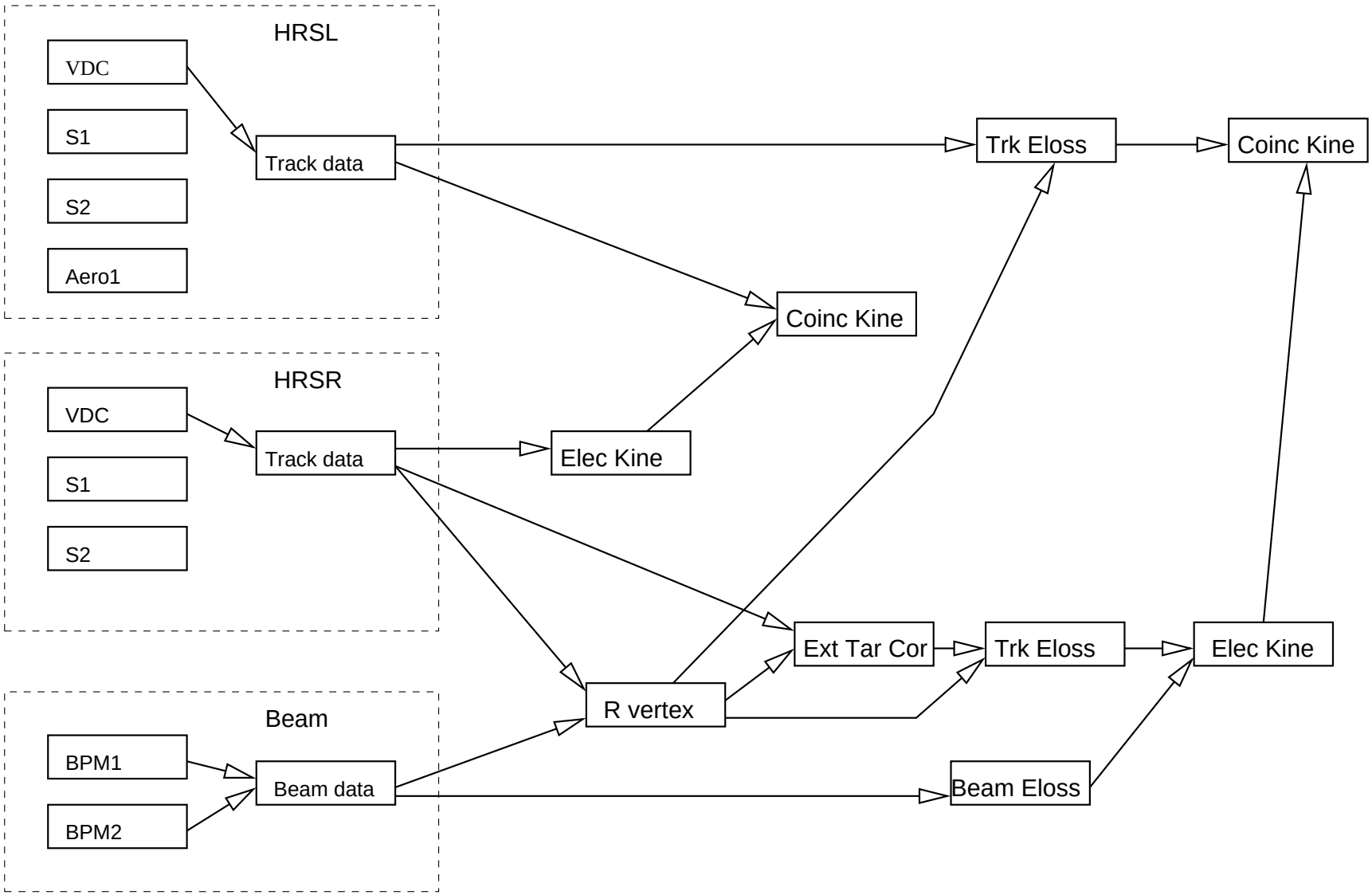


C++ Analyzer Architecture I

- Standard Hall A analysis software since 2003
- Class library on top of ROOT
- Toolbox of analysis modules
- Predefined modules for generic analysis tasks and standard Hall A equipment
- Analysis controlled via interpreted C++ scripts
- Special emphasis on modularity
  - "Everything is a plug-in"
  - User code separate from core code
  - Load external user libraries dynamically at run time
  - Users write no more than the code really needed
  - Core analyzer suitable for fixed installation, like ROOT itself

C++ Analyzer Architecture II

(TODO: ditch crummy 1980s graphics)



C++ Analyzer Output (DST)

- ROOT format
- Contents can be dynamically defined for each replay via input file

Example Output Definition File

```
# ----  Example e12345.odef  -------------

# Variables to appear in the tree.
variable L.s1.lt[4]
variable L.s1.rt

# The 'block' variables:  All data in Right HRS go to tree.
block R.*

# Formulas can be scalers or vectors.
# Lt4a is a scaler.
formula Lt4a 5.*L.s1.lt[4]

# Cuts can be defined globally and used in histograms.
# Cut C1 is a scaler.  Data is 0 or 1.
cut C1 L.s1.lt[4]>1350

# Histograms can involve formulas, variables, and cuts.
# TH1F, TH1D, TH2F, TH2D supported.
TH1F  rv1n  'L-arm vdc hits on V1'  L.vdc.v1.nhit  10 0 10
```

C++ Analyzer Database

- Currently only flat text files supported
- Key/value pairs with support for scalars, arrays, matrices, strings
- Support for time-dependent values (essential!)
- Plan to investigate server-based database system, *e.g.* the one developed by Hall D. Low priority

Example Database File

```
B.mwdc.planeconfig = u1 u1p x1 x1p v1 v1p \
                     u2 x2 v2 \
                     u3 u3p x3 x3p v3 v3p

# "Crate map":  crate slot_lo slot_hi model# resol   nchan
B.mwdc.cratemap =  3     6       21      1877   500     96  \
                   4     4       11      1877   500     96  \
                   4     17      24      1877   500     96

--[ 2008-02-31 23:59:45 ]
B.mwdc.maxslope      = 2.5

B.mwdc.size          = 2.0  0.5   0.0
B.mwdc.x1.size       = 1.4  0.35  0.0
```

C++ Analyzer Performance

(insert suitably impressive performance numbers here)

C++ Analyzer Parallelization

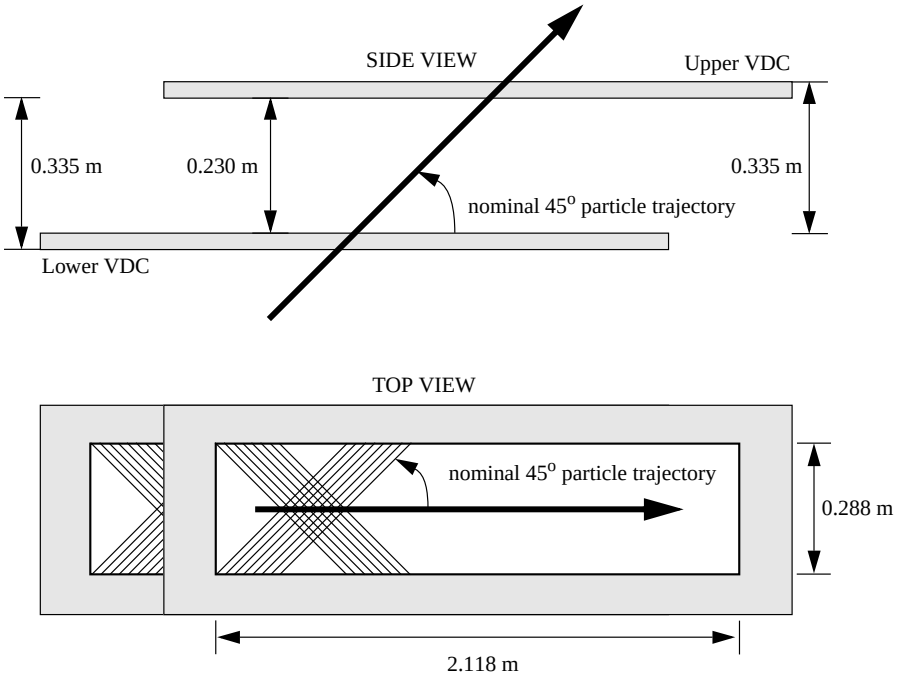
(description of event distribution via `fork`
yes, it's not going to be multi-threaded, but multi-process)

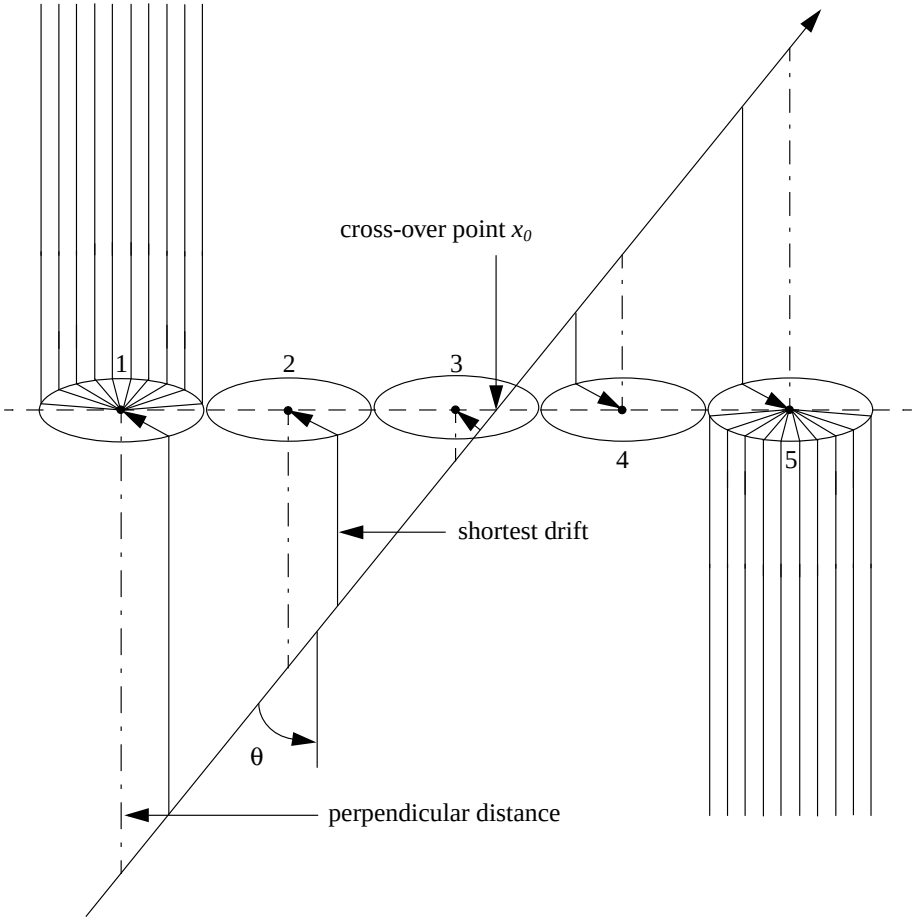
Decoding Event Data from Pipelined Electronics

(should discuss details since it's an action item)

HRS VDC Tracking

- Position x_0 determined based on fit of 5 independent time measurements
- Position resolution $\approx$ 225 μm FWHM





SBS Tracking I

(GEM trackers, DAQ, parameters, rates, occupancies)

SBS Tracking II

(TreeSearch algorithm, performance tests with MC)

SBS Calorimeter Analysis

(problem description, HCal illustration)

HRS Optics Calibrations

("optimization" software, procedure)

General Calibration Tools

(VDC time offsets, VDC time-to-distance; what else?)

Matrix-Based Simulations

(SIMA & MCEEP details)

Particle Transport-Based Simulations

(GEMC illustration)

Online Histogramming

(OnlineGUI screenshot)

Source Code Control & Build System

(`CVS` $\rightarrow$ `git`; GNU toolchain)

Documentation & Developer Support

- Web-based user guide
- Example scripts
- ROOT `THtml` reference documentation
- Software development kit (SDK)

(example/illustration/screenshot(s) here)

Areas of Concern, Weaknesses

("Must", "Should", "Like" prioritized list)