slic & lcsim

Norman Graf (SLAC)
(for the ALCPG Simulation & Reconstruction WG)
HPS Software Meeting
April 7, 2011

LCD Simulation Mission Statement

- Provide full simulation capabilities for Lepton Collider physics program:
  - Physics simulations
  - Detector designs
  - Reconstruction and analysis
- Need flexibility for:
  - New detector geometries/technologies
  - Different reconstruction algorithms
  - Different machine environments
- Limited resources demand efficient solutions, focused effort.

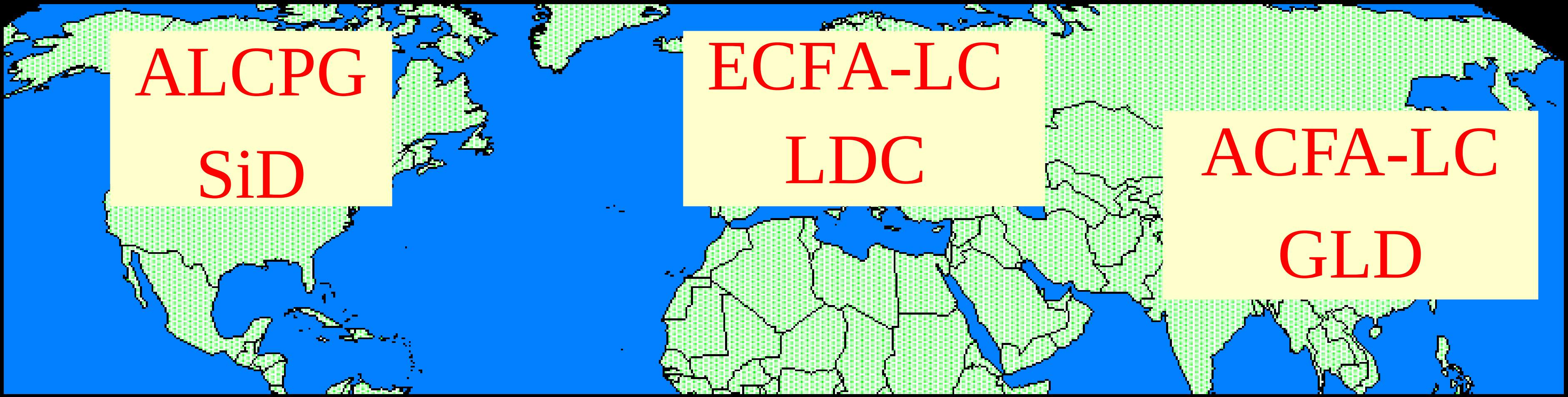
Overview: Goals

- Facilitate contribution from physicists in different locations with various amounts of time available.
- Use standard data formats, when possible.
- Provide a general-purpose framework for physics software development.
- Develop a suite of reconstruction and analysis algorithms and sample codes.
- Simulate benchmark physics processes on different full detector designs.

Detector Performance Studies

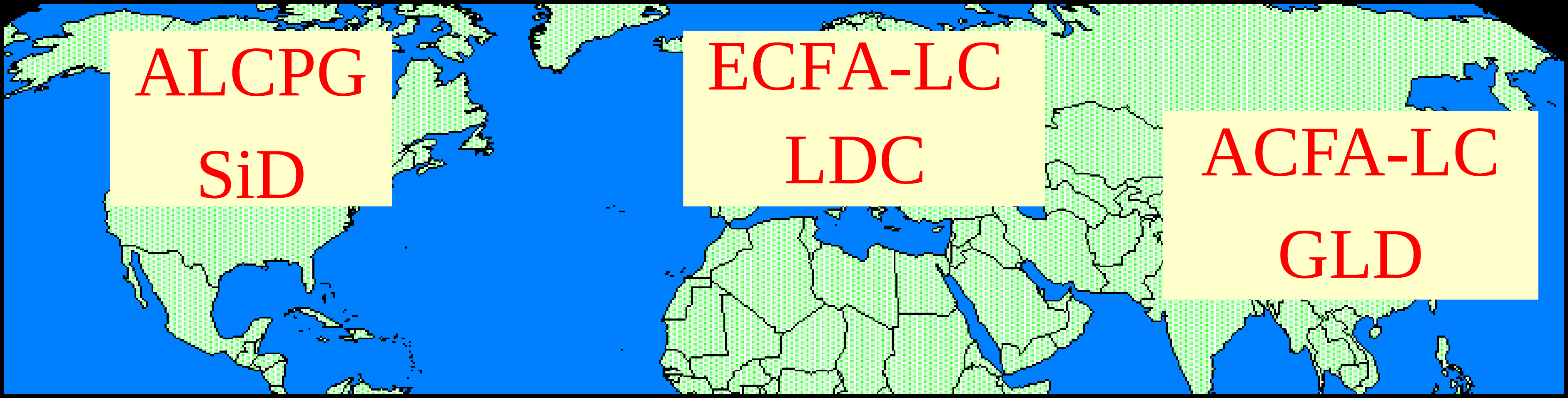
- The ILC community recently finished a very intensive round of detector performance and optimization studies, culminating in the submission of LOI’s and is engaged in preparing more detailed updates for the DBD in 2012.
- The CLIC community is currently engaged in an aggressive effort to provide a CDR later this year.
- The Muon Collider community will be using these tools for physics and detector studies.
- Provides an excellent opportunity for HPS to benefit from the very large investment in software and the lessons learned.
  - What worked
  - What didn’t work

LCIO



slic	MOKKA	JUPITER
org.lcsim	MarlinReco	Satellites
Java	C++	root

LCIO



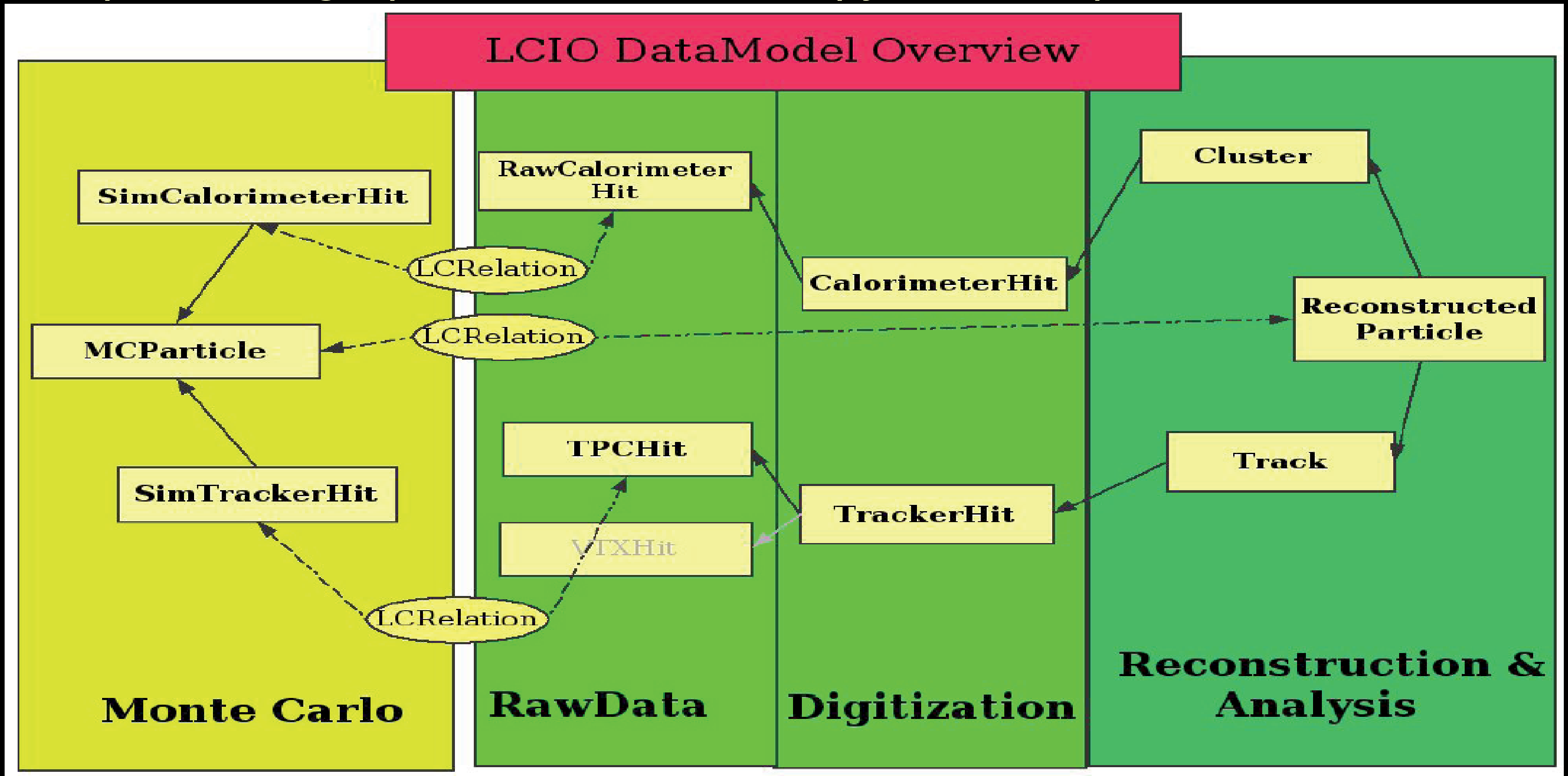
LCIO

Common Data Model

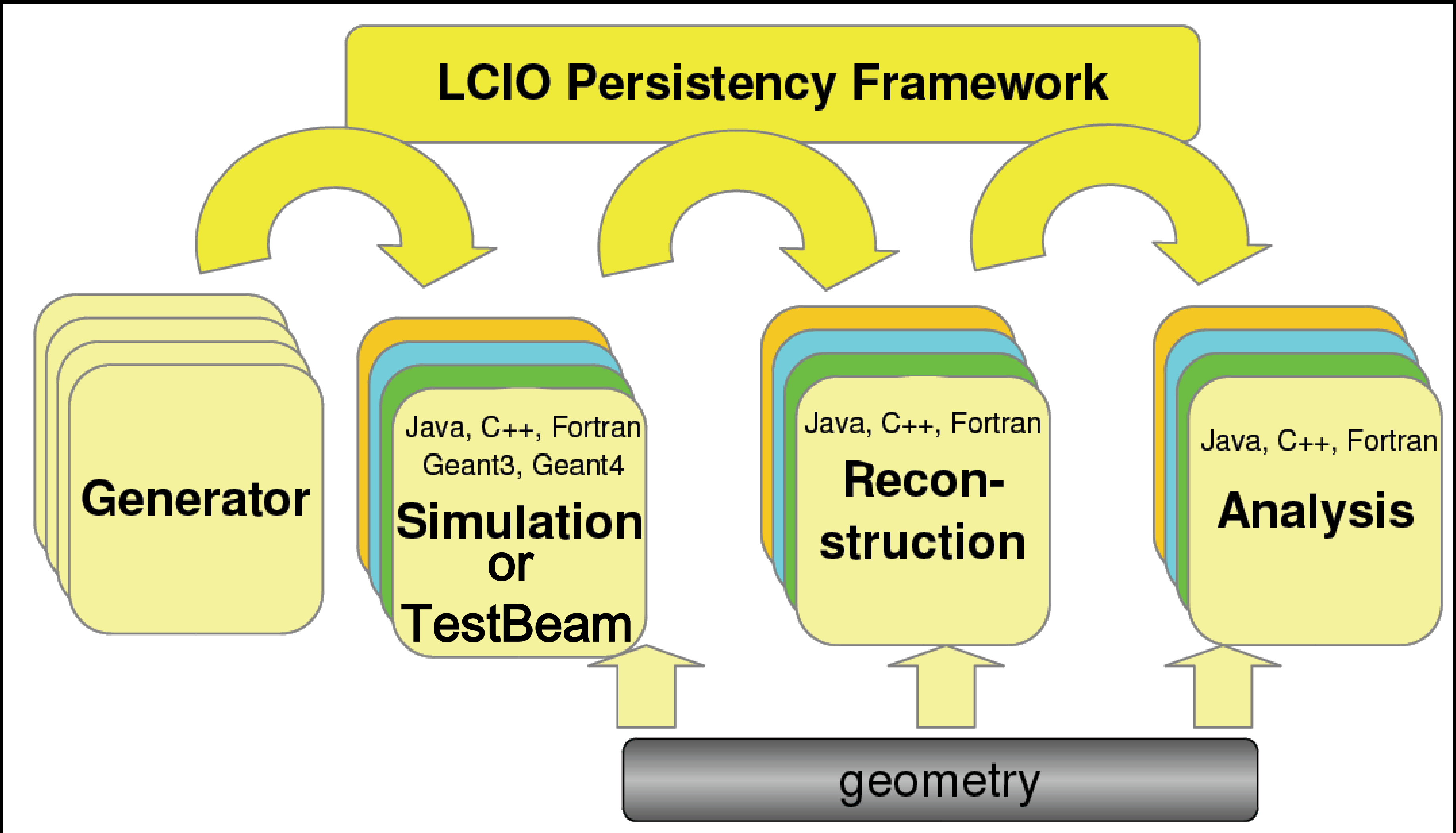
Common IO Format

LCIO Overview

- Object model and persistency format for HEP events
  - MC simulation
  - Test Beam data
  - Reconstructed Objects
- Multiple bindings (C++, Java, Fortran, python, root)



LCIO Overview



LCIO

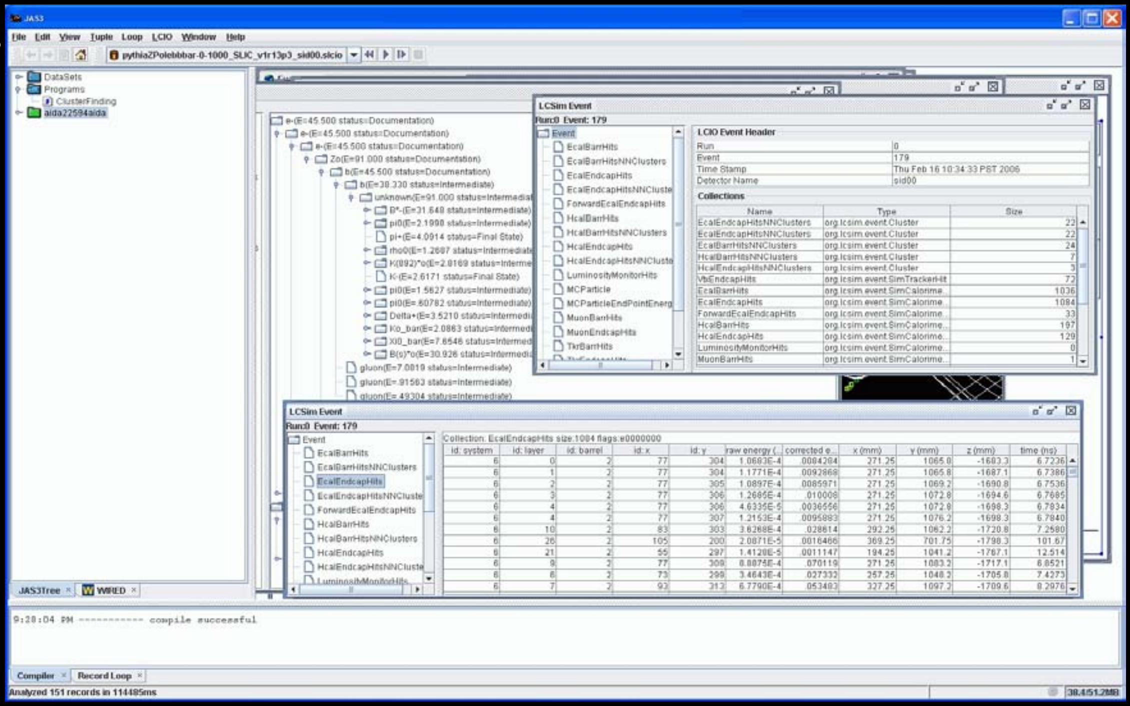
- Direct access to events
  - Overlay of random background events
  - physics analysis using preselection on metadata
- ROOT dictionary
  - use LCIO classes in root macros
  - write simple ROOT trees
  - write complete LCIO events in one ROOT branch
  - comes at a cost, of course, slower than native access
- LCIO 2.0 will add requested user functionality.

LCIO Interoperability

- All three regional LCD simulators write LCIO
  - Cross-checks between data from different simulators
  - Read/write LCIO from
    - Fast MC / Full Simulation / Test Beams
    - Different detectors
    - Different reconstruction tools
    - Different frameworks, languages, operating systems
- Reconstruction also targets LCIO
  - Can run simulation or reconstruction in one framework, analysis in another, benefit from worldwide developer community.
    - Generate events in Jupiter, analyze in MarlinReco.
    - org.lcsim to find tracks in Java, LCFI flavor-tagging to find vertices using MarlinReco (C++) package, slicPandora to run Particle Flow Analysis.

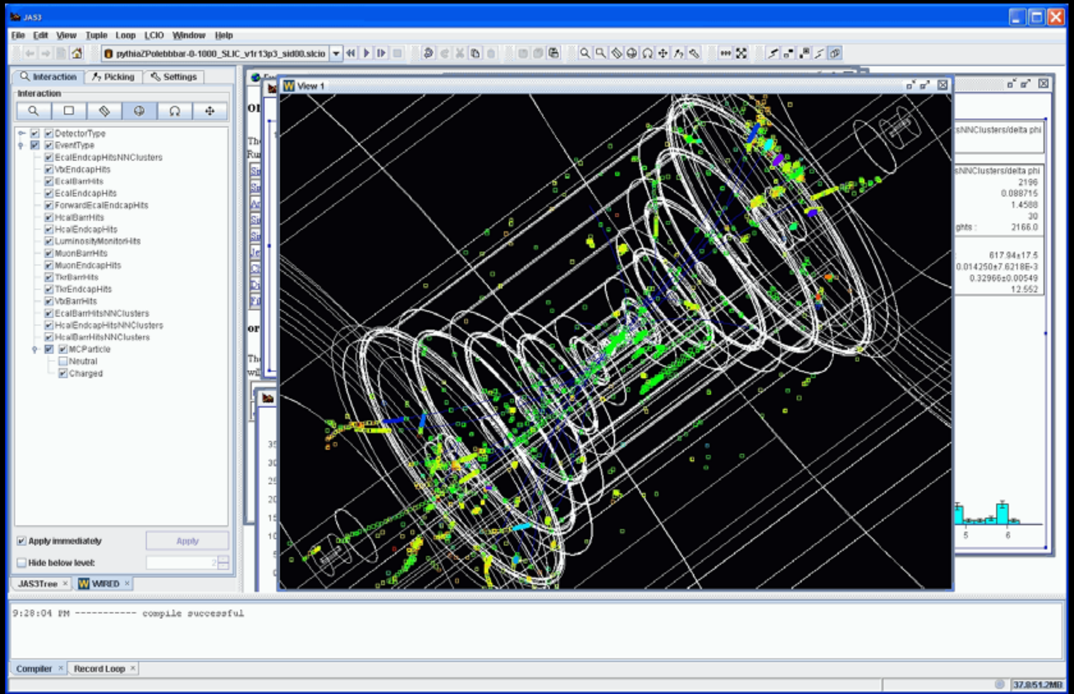
LCIO Event Browser

- Fully supported within JAS.
- Open any LCIO file, browse collections and objects.
- Traverse MC particle hierarchies
- Print, sort, analyze.



LCIO Event Display

- Fully integrated within JAS using Wired.
- Fully interactive event display
- Detector & Event objects selectable, pickable, queryable, can have cuts applied, etc.
  - Not just a static image.



Raw Data

- LCIO has been used for many years by various testbeam experiments, both tracking and calorimetry.
  - EDM supports raw data taking and analysis.
  - Simple, robust & fast
- Many tools exist for data monitoring, QA, analysis, etc.
- Will need to learn what infrastructure and support exists for EVIO before deciding how best to interface to that file format and event data model.

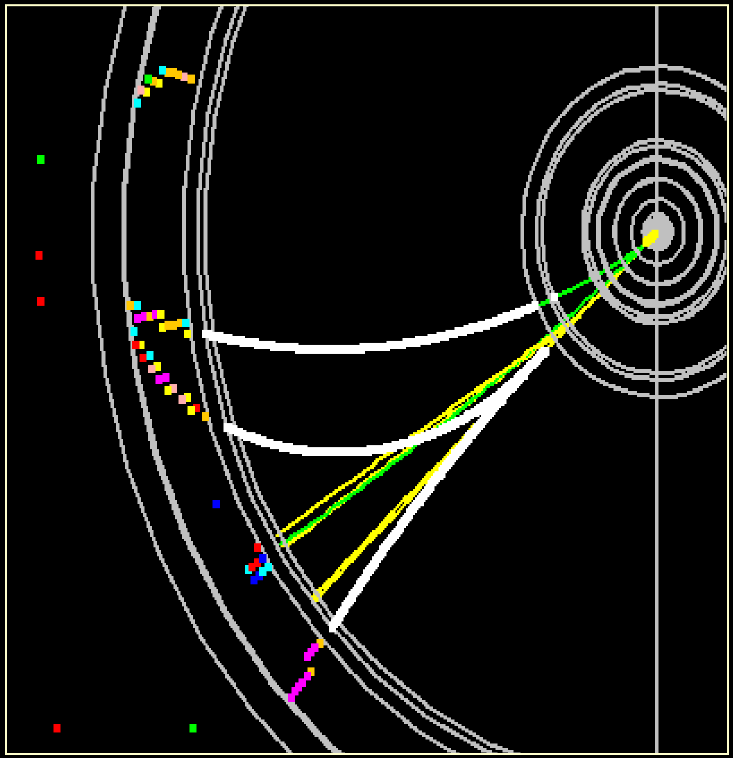
Fast Detector Response Simulation

- Covariantly smear tracks with matrices derived from geometry, materials and point resolution using Billoir's formulation.
  - Analytically from geometry description.
- Smear neutrals according to expected calorimeter resolution (EM for γ, HAD for neutral hadrons)
  - Derived from full Geant4 simulations
- Create reconstructed particles from tracks and clusters (γ, e, μ from MC, $\pi^{+/-}$, K^0_L for others)
- Can also dial in arbitrary effective jet energy resolution.
  - Derived from full simulation, reconstruction & analysis.
- "Supersymmetry, the *ILC*, and the LHC *inverse problem*"

lelaps

- Fast detector response package.
- Handles decays in flight, multiple scattering and energy loss in trackers.
- Parameterizes particle showers in calorimeters using Grindhammer & Peters, and Bock algorithms.
- Produces detector data at the hit level.
  - Feeds directly into full reconstruction.
- Uses runtime geometry (compact.xml $\rightarrow$ godl).

Lelaps: Decays, dE/dx, MCS

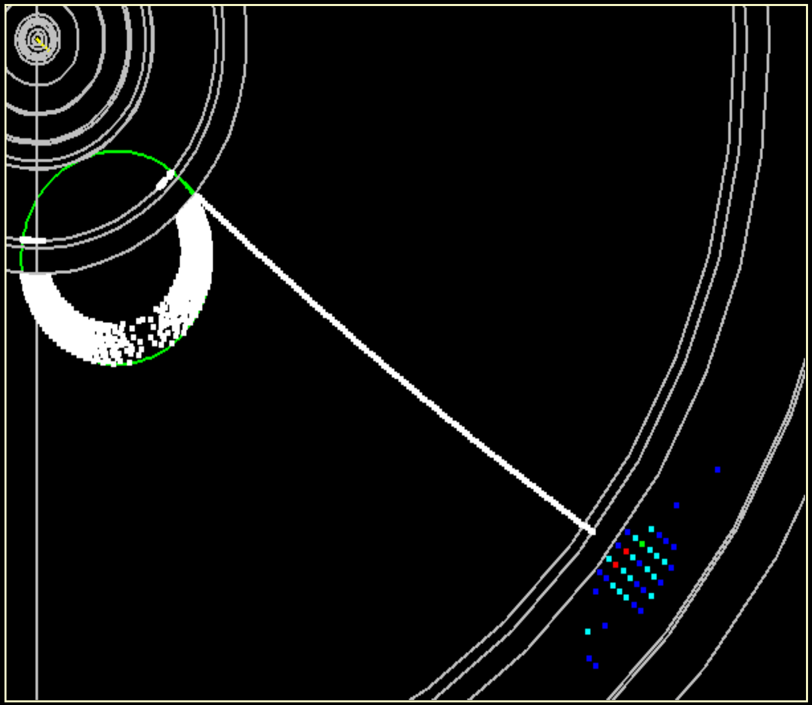


$\Omega^- \rightarrow \Xi^0 \pi^-$

$\Xi^0 \rightarrow \Lambda \pi^0$

$\Lambda \rightarrow p \pi^-$

$\pi^0 \rightarrow \gamma \gamma$ as simulated by Lelaps for the LDC model.



gamma conversion as simulated by Lelaps for the LDC model.

Note energy loss of electron.

Detector Design (GEANT 4)

- Need to be able to flexibly, but believably simulate the detector response for various designs.
- The daunting machine backgrounds expected at the HPS will require detailed full detector simulations.
- GEANT is the de facto standard for HEP physics simulations.
- Use runtime configurable detector geometries
- Write out “generic” hits to digitize later.

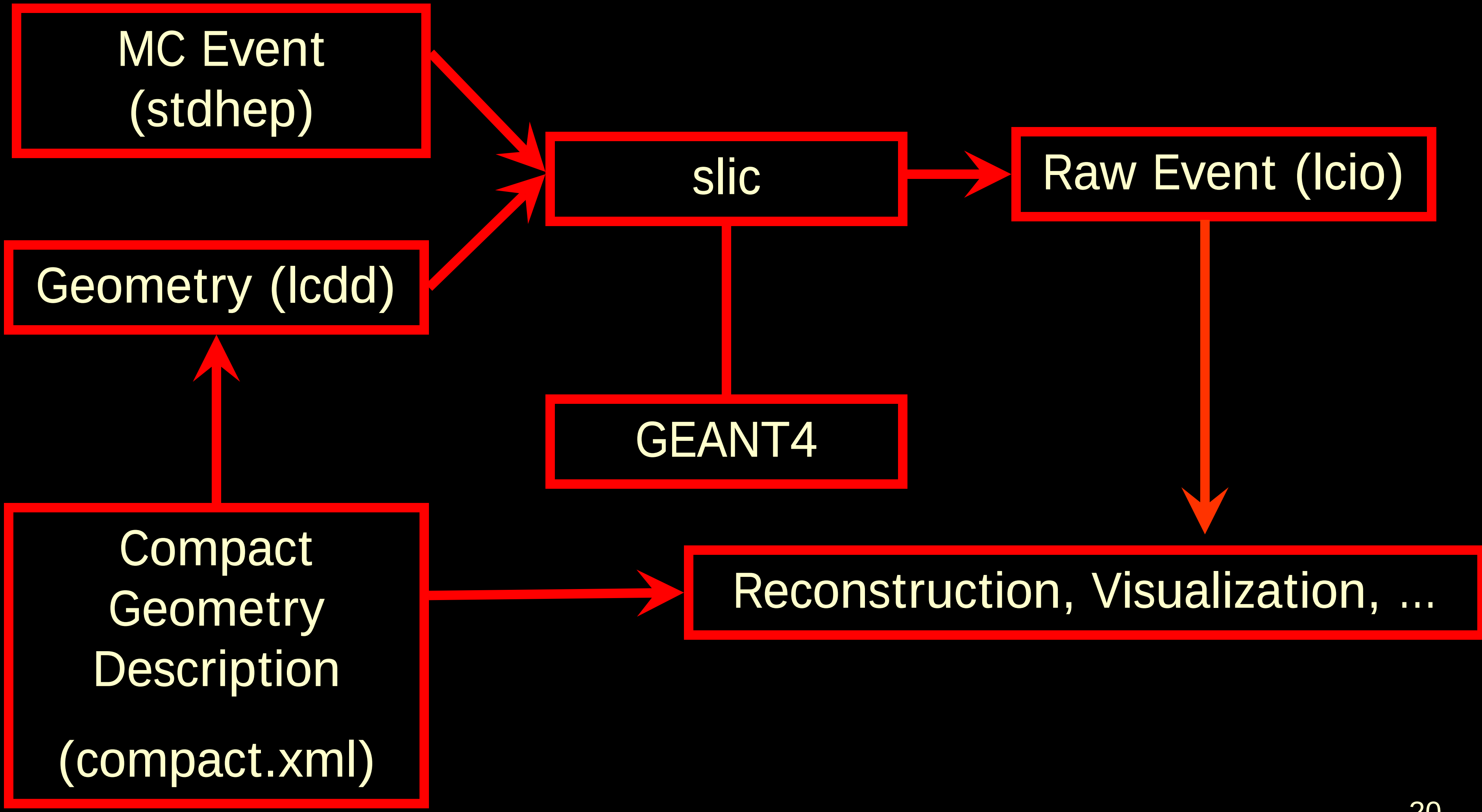
Full Detector Response Simulation

- Use Geant4 toolkit to describe interaction of particles with matter and fields.
- Thin layer of LC-specific C++ provides access to:
  - Event Generator input (binary stdhep format)
  - Detector Geometry description (XML)
  - Detector Hits (LCIO)
- Geometries fully described at run-time!
  - In principle, as fully detailed as desired.

Geometry Definition

- Goal was to free the end user from having to write any C++ code or be expert in Geant4 to define the detector.
- All of the detector properties should be definable at runtime with an easy-to-use format.
- Selected xml, and extended the existing GDML format for pure geometry description.

LC Detector Full Simulation

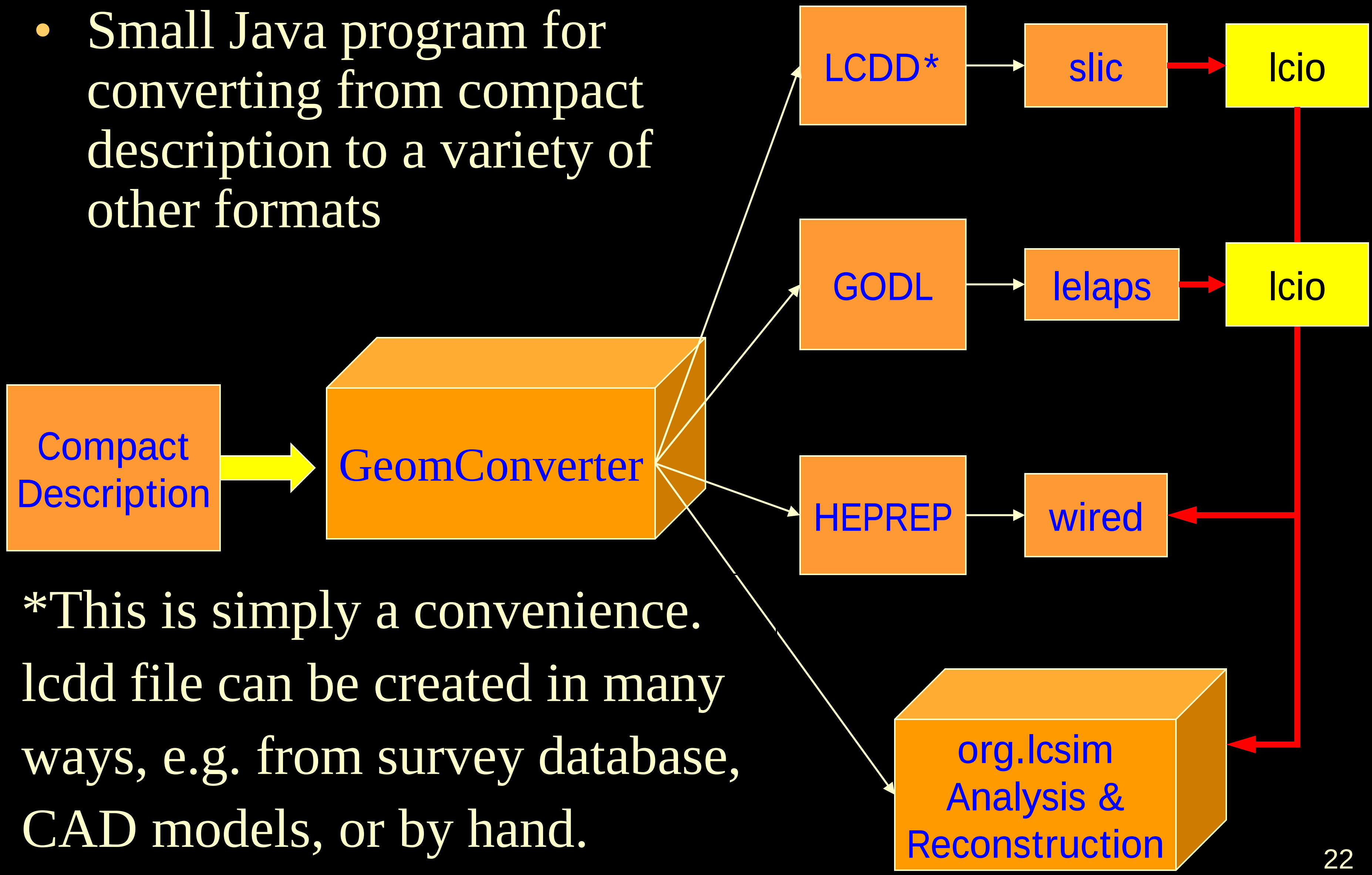


slic: The Executable

- Build static executables on Linux, Windows, Mac.
- Commandline or G4 macro control.
- Only dependence is local detector description file.
  - Trivial Grid usage (no database call-backs, etc.)
  - Grid ready, Condor and lsf scripts available.
- Event input via stdhep, particle gun, …
- Detector input via GDML, lcdd
- Response output via LCIO using generic hits.
- Fast (~30s/event for TeV ILC)

GeomConverter

- Small Java program for converting from compact description to a variety of other formats

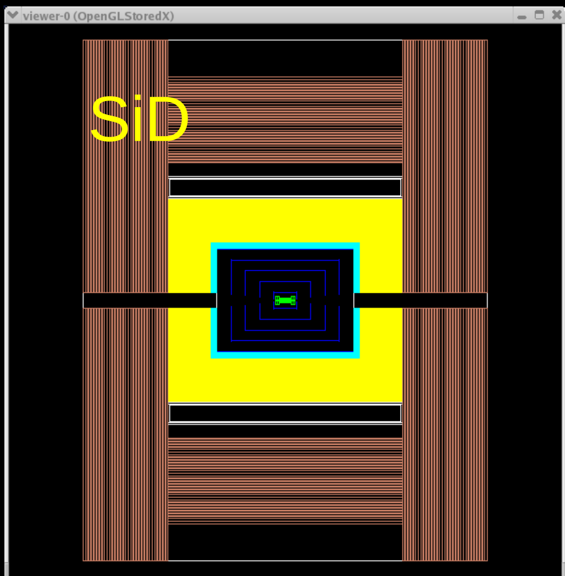


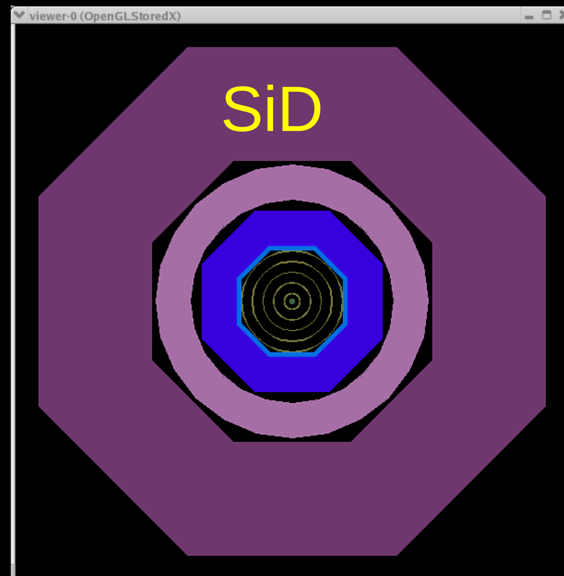
*This is simply a convenience. lcdd file can be created in many ways, e.g. from survey database, CAD models, or by hand.

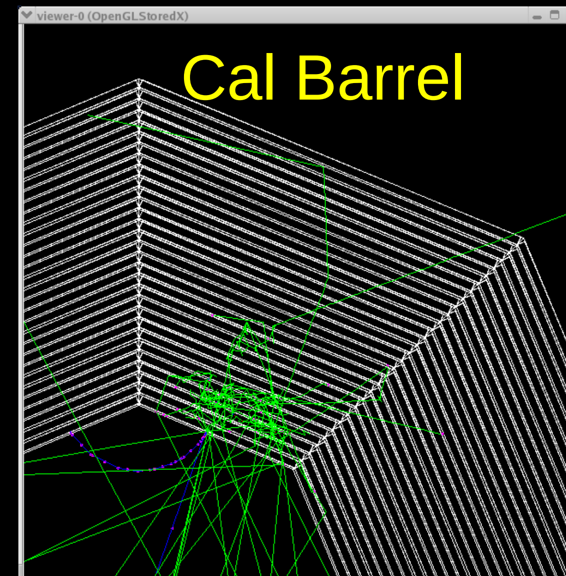
Detector Variants

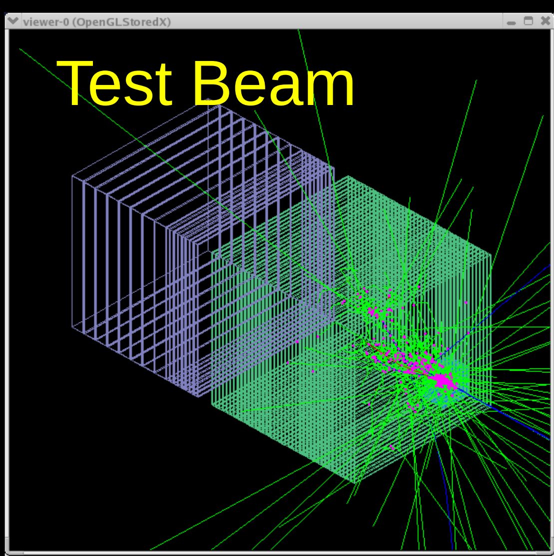
- Runtime XML format allows variations in detector geometries to be easily set up and studied:
  - Absorber materials and readout technologies for sampling calorimeters
    - e.g. Steel, W, Cu, Pb + RPC vs. GEM vs. Scintillator readout
  - Optical processes for dual-readout or crystal calorimeters
  - Layering (radii, number, composition)
  - Readout segmentation (size, projective vs. nonprojective)
  - Tracking detector technologies & topologies
    - TPC, Silicon microstrip, pixels, …
    - “Wedding Cake” Nested Tracker vs. Barrel + Cap
  - Far forward MDI variants, shielding, field strength, etc.

Example Geometries

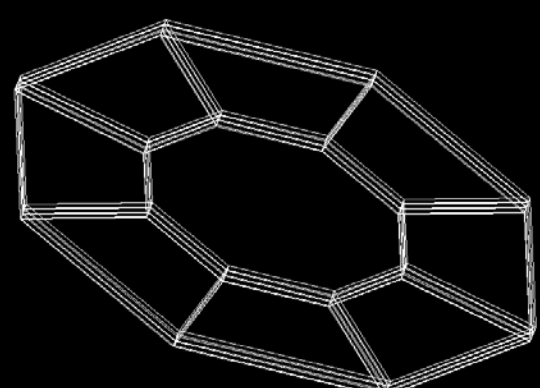




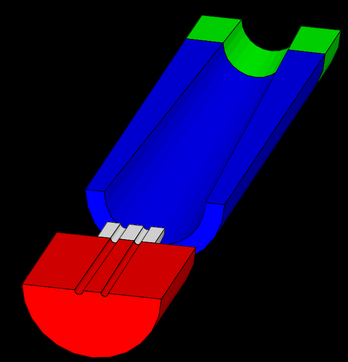




Cal Endcap

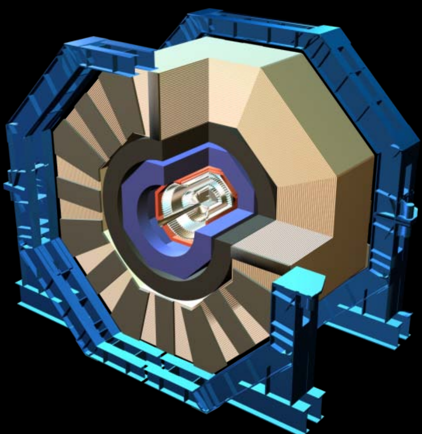


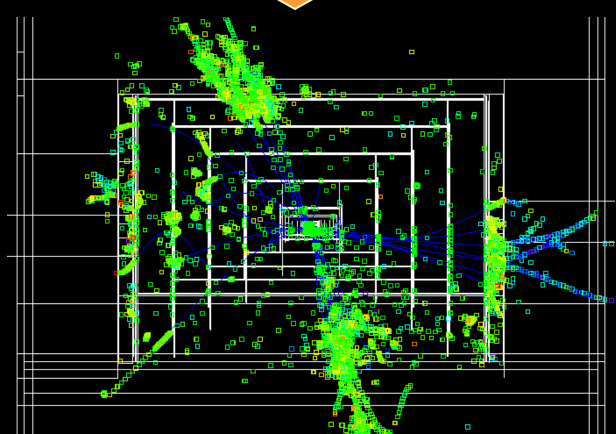
MDI-BDS



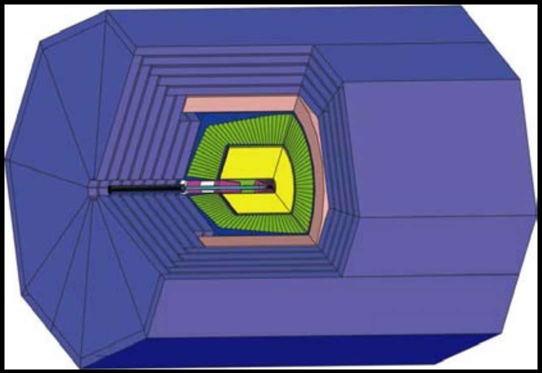
ILC Full Detector Concepts

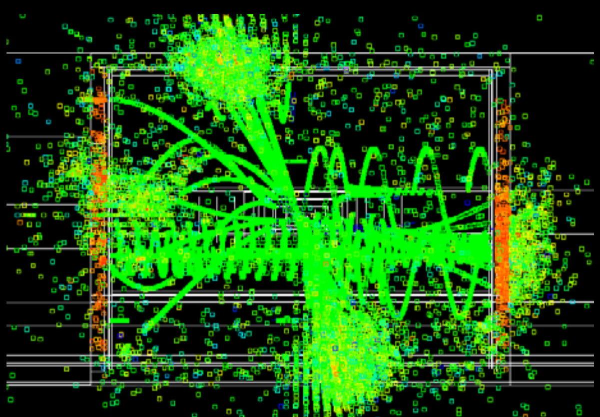
SiD



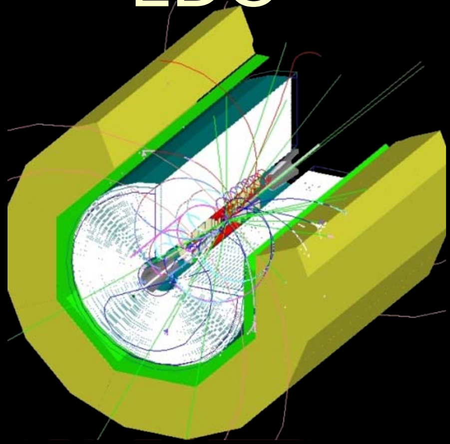


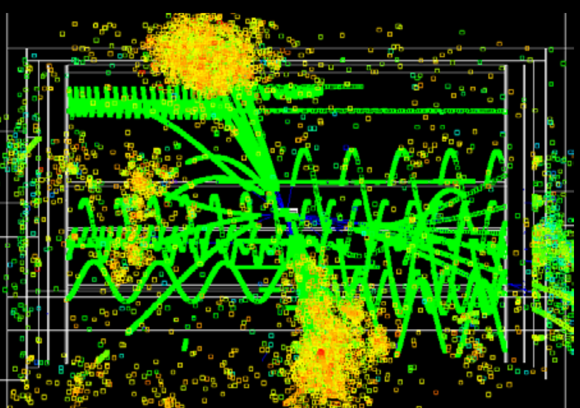
GLD



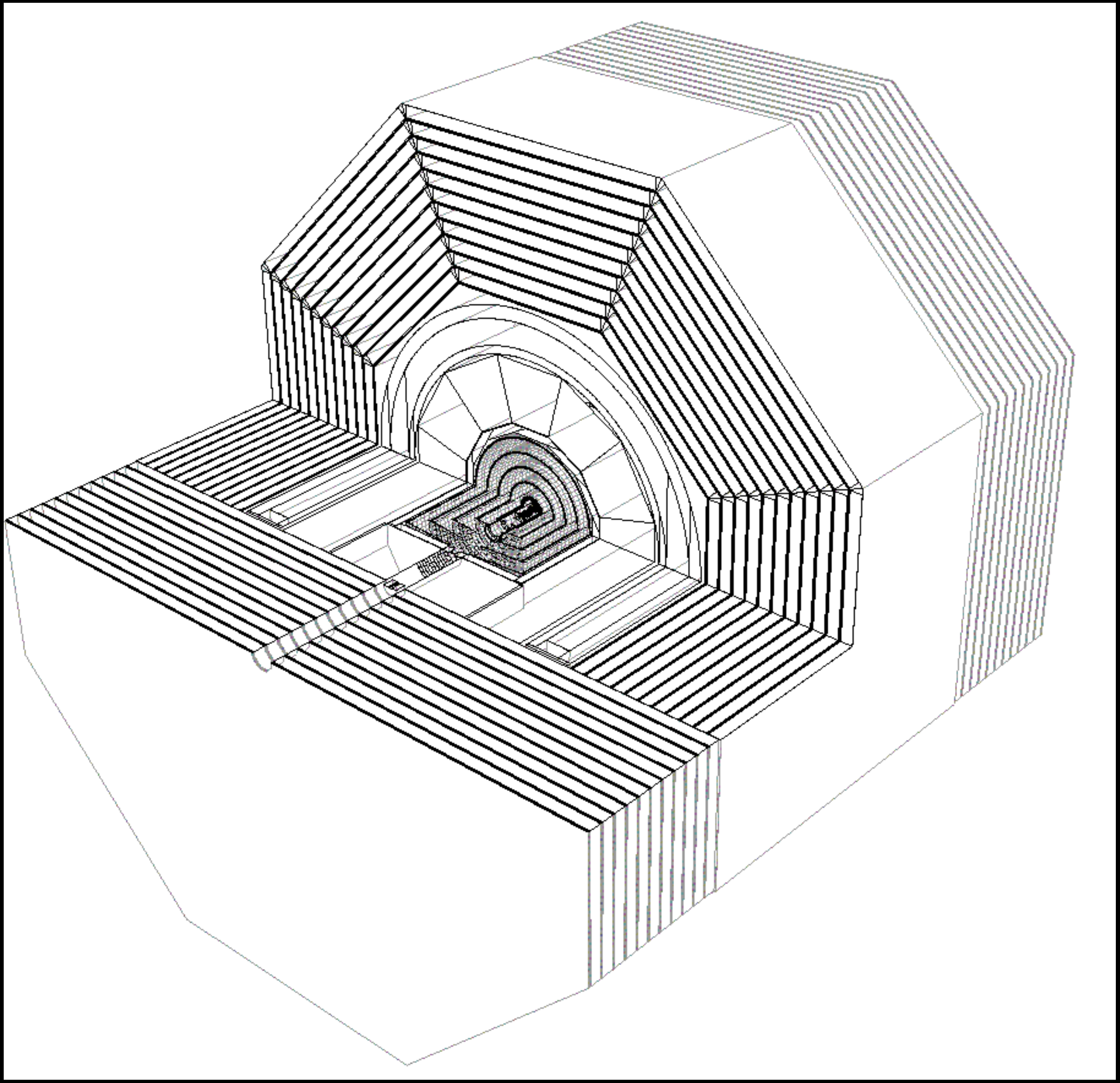


LDC

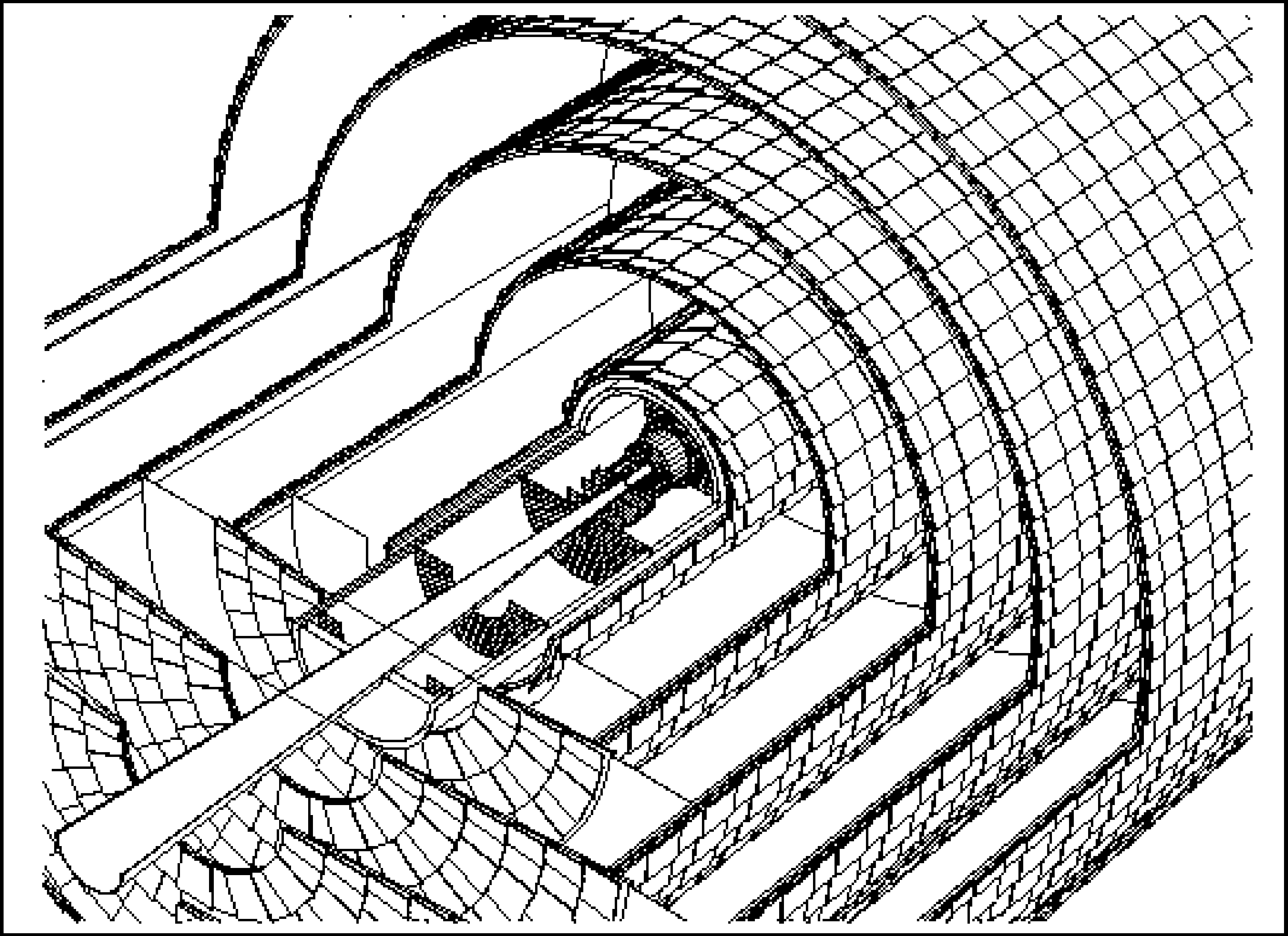




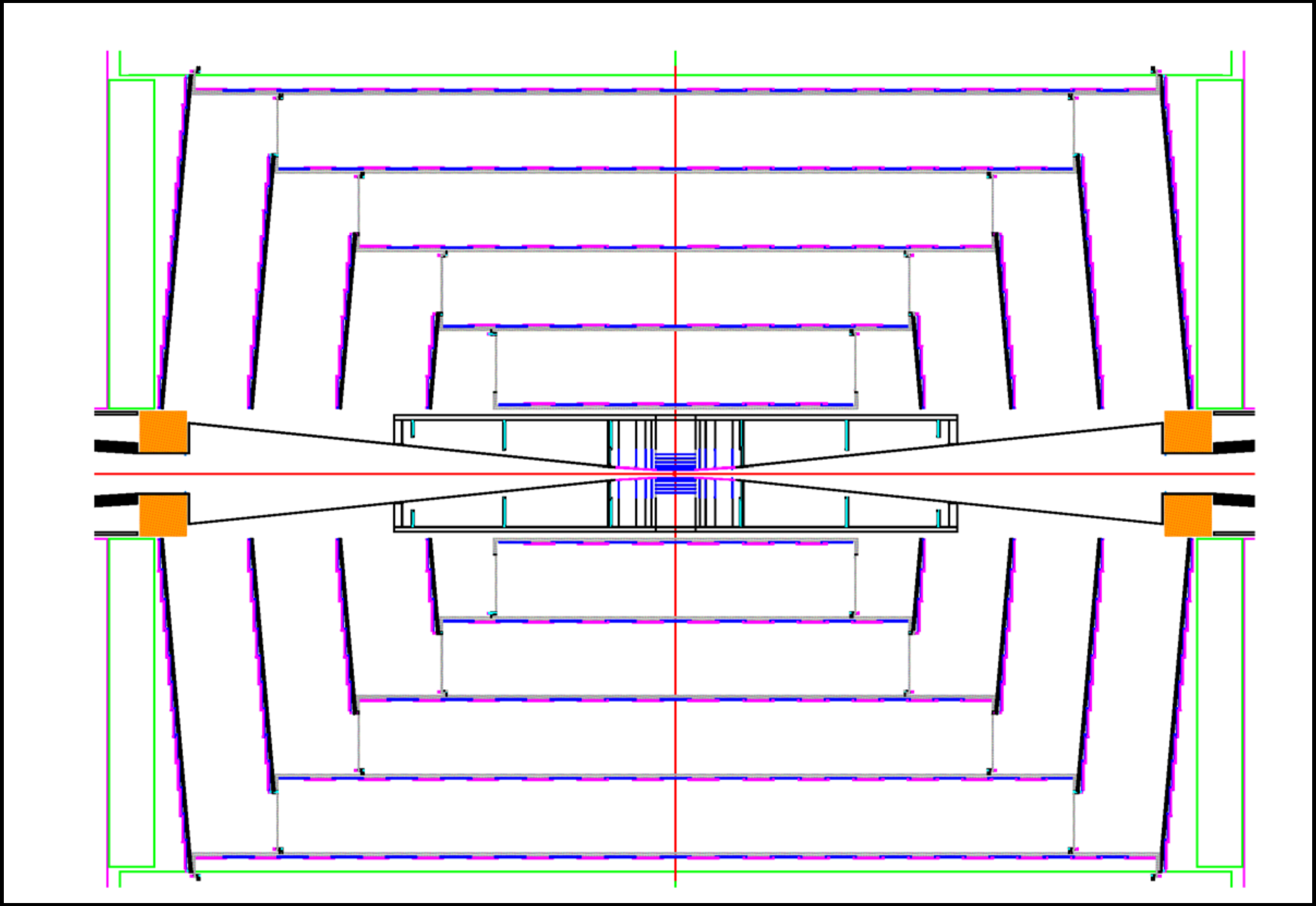
Silicon Detector Tracker



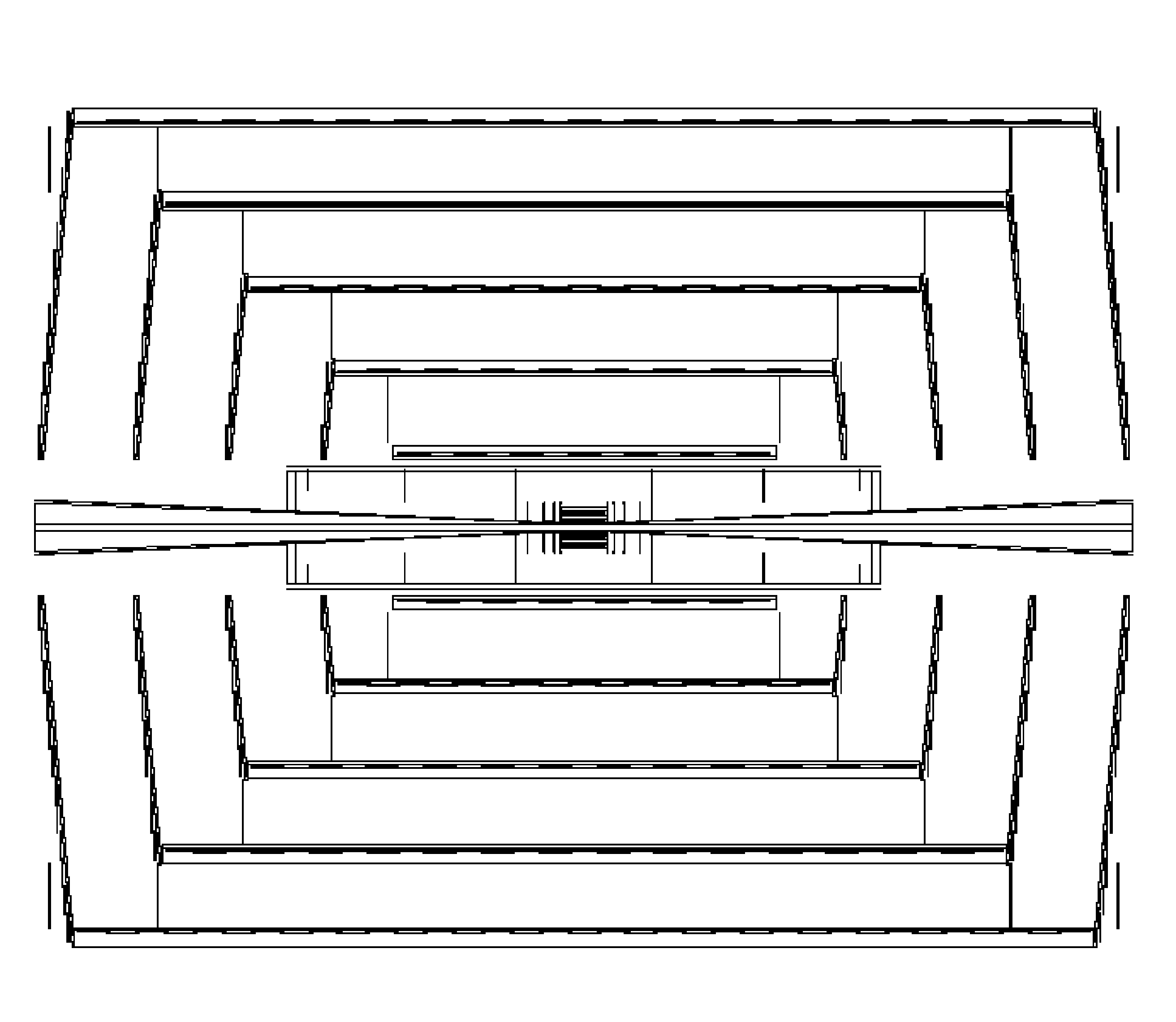
Silicon Detector Tracker



Silicon Detector Tracker



Silicon Detector Tracker



Simulating the HPS Experiment

- Compared to Silicon Detector, HPS tracker is straightforward to implement.
  - Complete control over definitions of tracker sensitive wafers and support structures.
  - Very detailed models for charge deposition, drift and diffusion.
  - Detailed model for the electronics response.

    MC Hits→ Channel ID & Pulse Height → Clusters → Hits (x +/- δx)
  - Matt and Rich have implemented the designs, simulated the response, reconstructed tracks and vertices and presented analysis results many times.
    - will not repeat here.

Simulating the HPS Experiment II

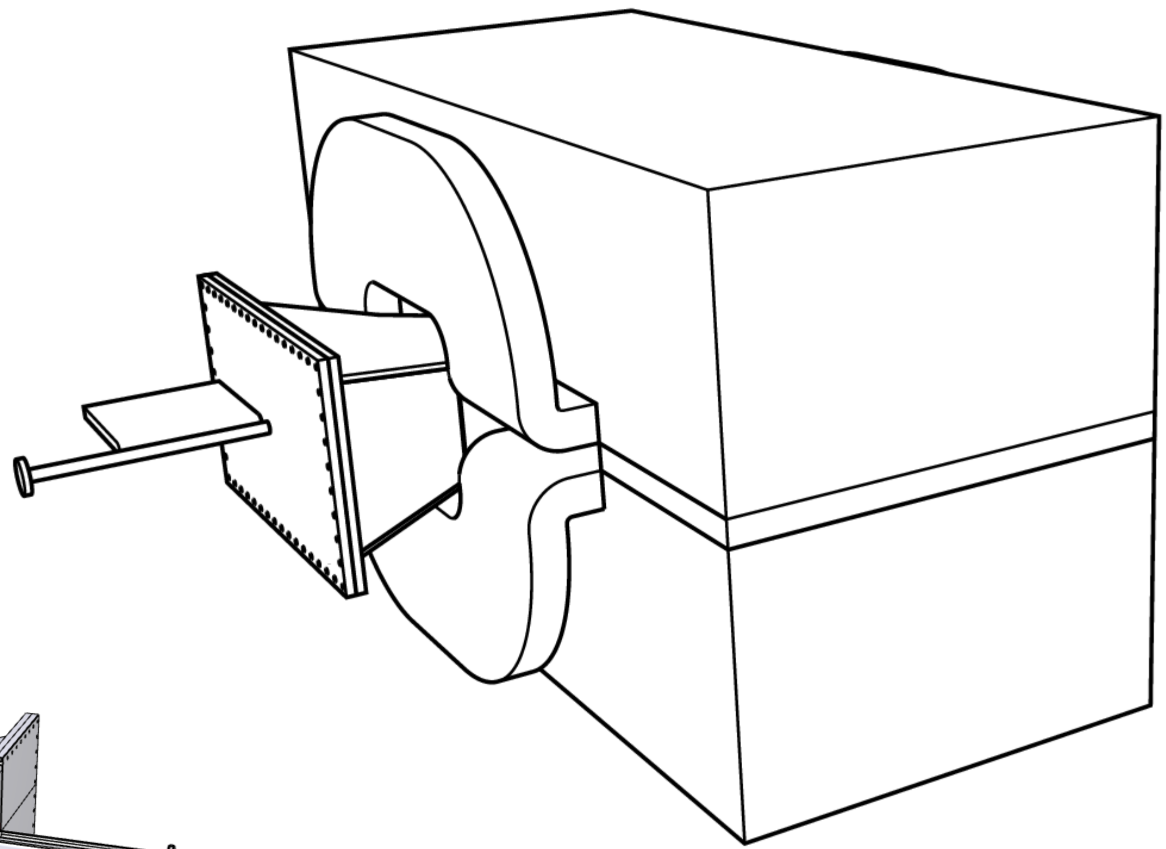
- Other major detector elements are the electromagnetic calorimeter and the muon system.
  - Will need same geometry in reconstruction.
  - Additions to compact.xml & GeomConverter needed to provide same geometry to sim and reco.
    - Straightforward, but still requires expert development.
      - Needs time from Jeremy McCormick
- Need detailed simulation of the beamline and support elements.
  - Complex geometries, but not needed in reconstruction.
  - Can go straight from CAD to slic via GDML.

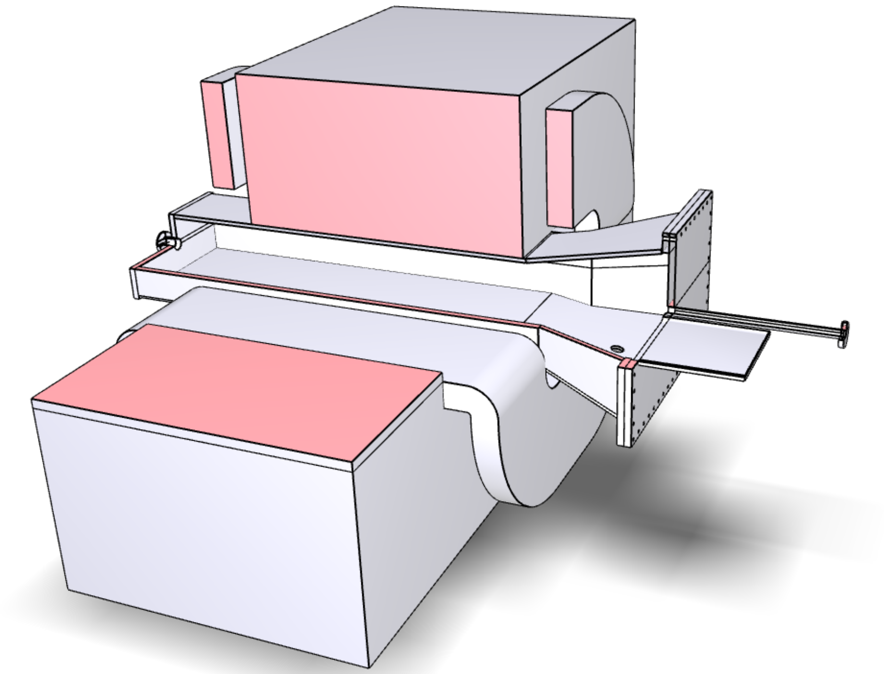
HPS Dipole and Vacuum Vessel

- CAD Model from Marco Oriunno
- Conversion to GDML by N. Graf
  - Resulting geometry is tesselated solid
    - performance not expected to be as good as using Geant4 primitives.
    - but most particles should never interact with these elements
  - included into compact.xml as gdml snippet

HPS Dipole and Vacuum Vessel

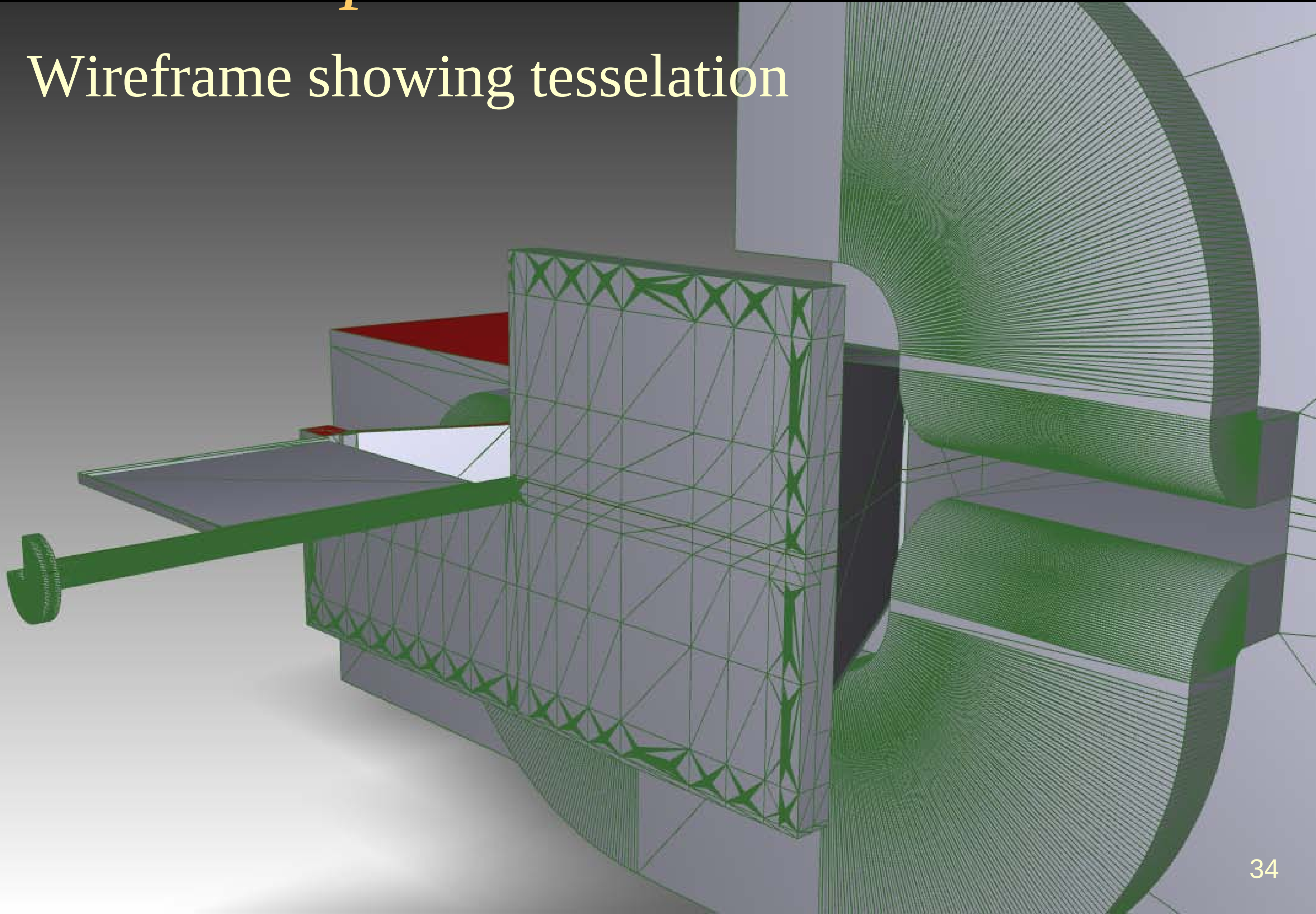
- Images from geometry output from slic.





HPS Dipole and Vacuum Vessel

- Wireframe showing tesselation

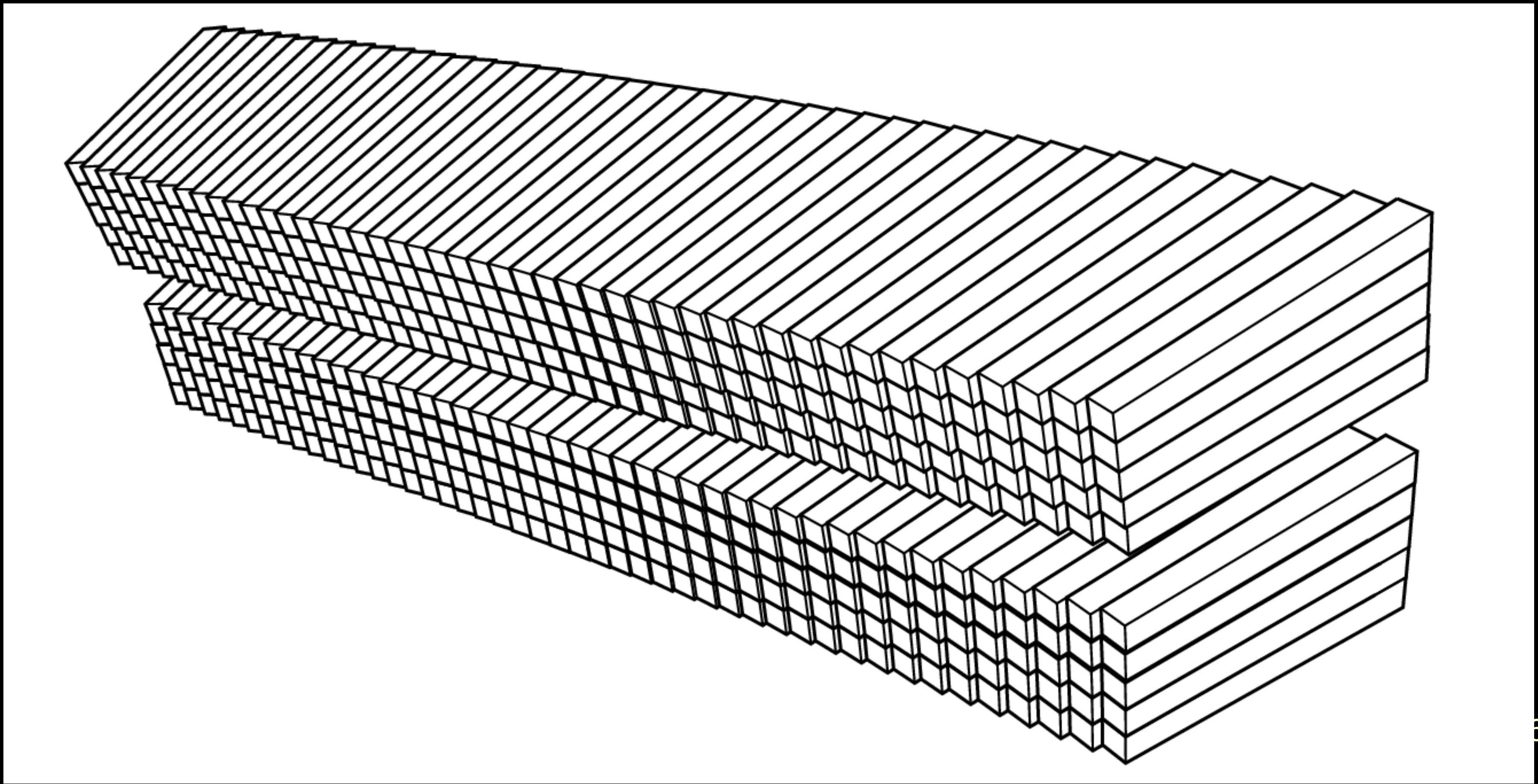


HPS ECal

- Will want to have the geometry definable from the compact.xml in order to have it accessible from the reconstruction.
- However, can also implement simply using the CAD model to get started and as proof-of-principle.

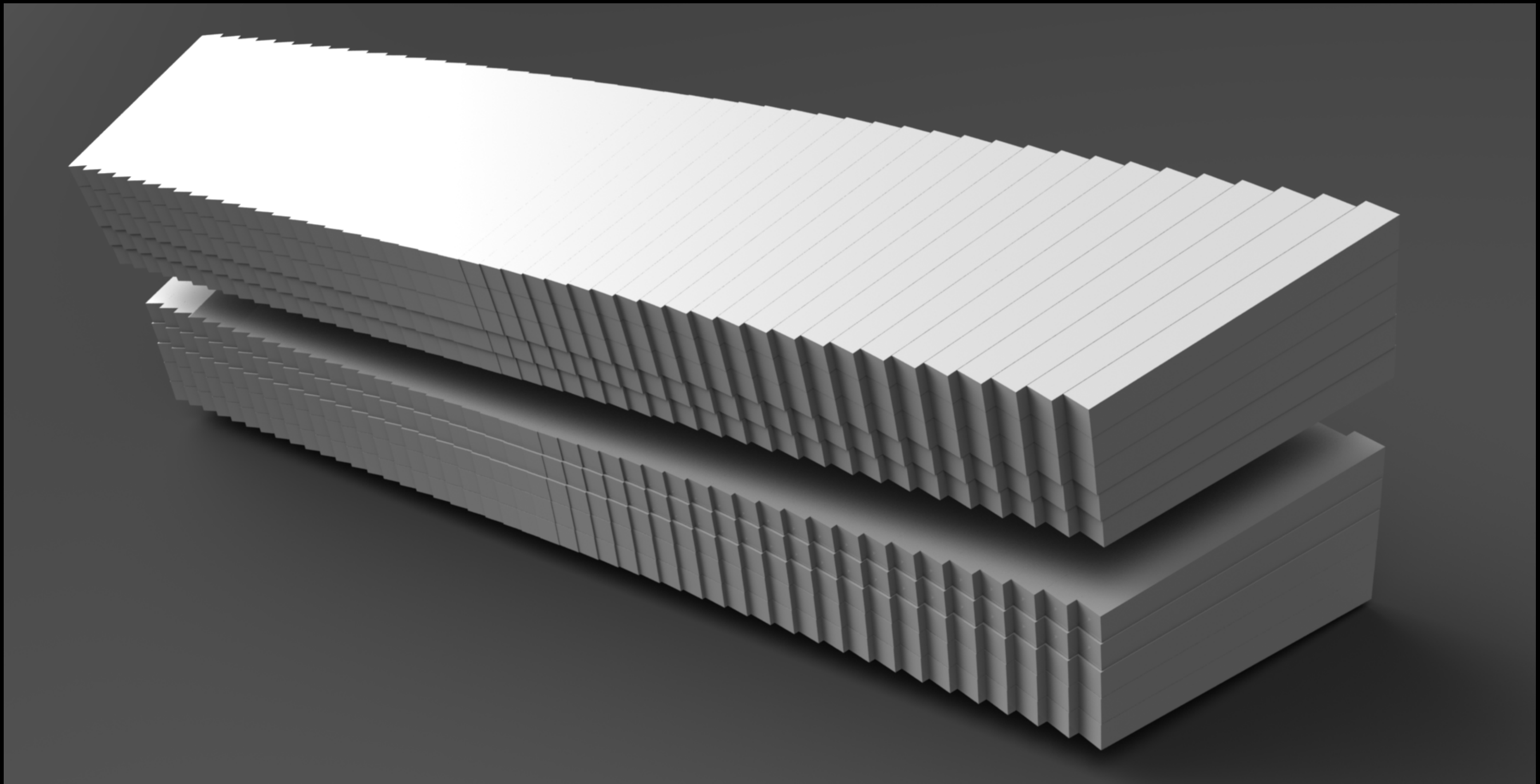
HPS ECal

- CAD Model from Marco Oriunno.
- Translated to GDML by N. Graf.
  - not yet a sensitive detector (but could be made so)



HPS ECal

- CAD Model from Marco Oriunno.
- Translated to GDML by N. Graf.

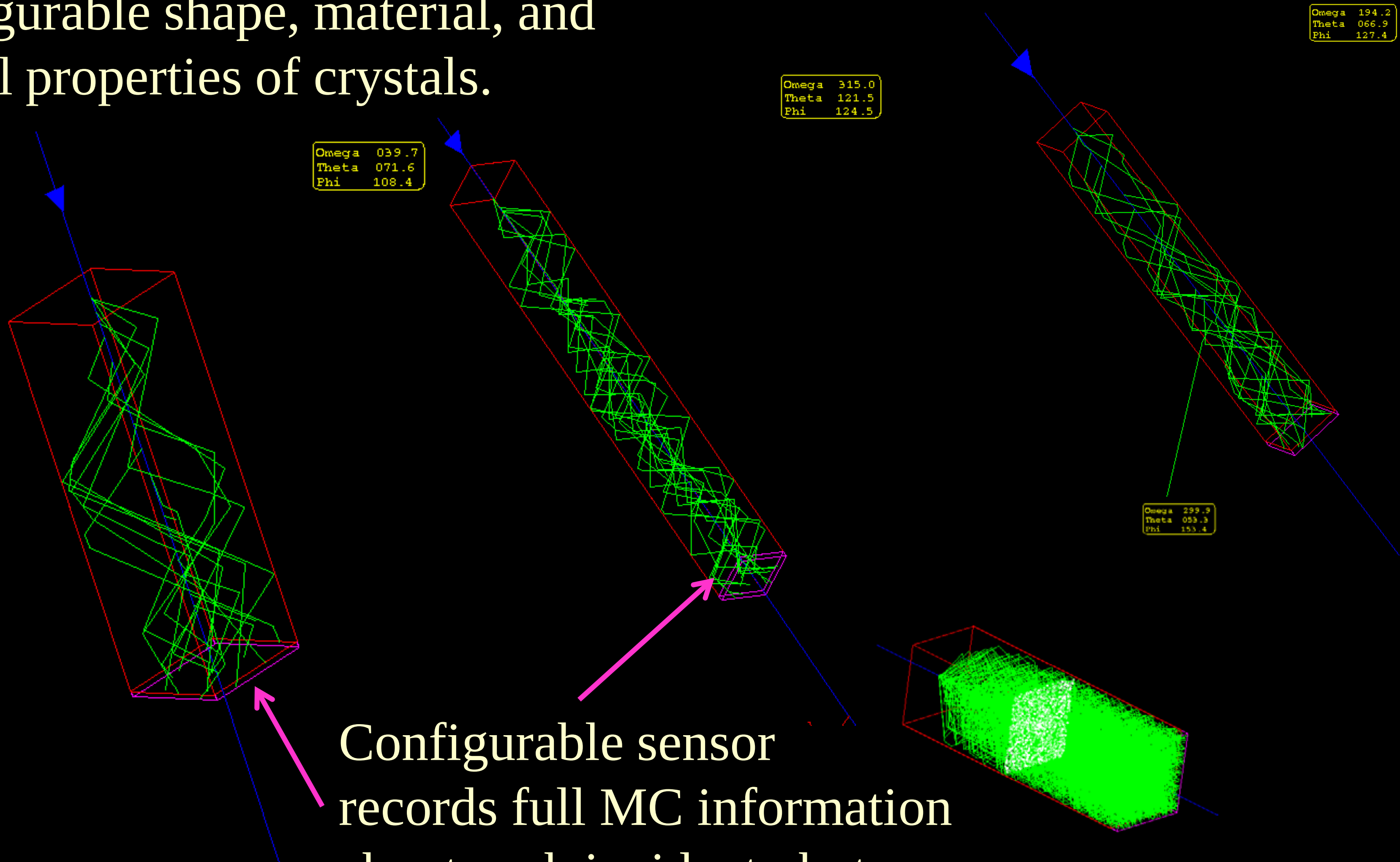


HPS ECal Response

- Default Sensitive Detector response for calorimeters is to simply record energy deposition, time and MC particle information.
- Hans Wenzel also implemented scintillation and Cherenkov light deposition within slic/lcsim for studies of total absorption, dual-readout crystal calorimetry.
  - simple accumulation of energy deposit in crystal
- Currently implementing full optical photon ray tracing within crystal and propagation to sensitive detector.

Optical Ray Tracing in Crystals

Configurable shape, material, and optical properties of crystals.



Configurable sensor records full MC information about each incident photon

Courtesy Hans Wenzel

HPS ECal & Muon System Todo

- Implement geometry description of crystal array, plus wrapping, support, etc. into compact.xml and GeomConverter.
  - Need some input on level of detail of simulation.
    - Is simple Σ dE good enough?
    - Is simple Σ (scint + Cherenkov) good enough?
    - Do we need full optical definition and ray tracing?
    - How detailed do we need to model the readout sensor?
- Ignorant of latest muon system geometry or readout.
  - Is current calorimeter support sufficient?

Beamline, Magnets and Supports Todo

- Is CAD to Geant solution usable?
  - Memory and time requirements on slic.
- If not, what level of simplification can be achieved automatically? What level requires manual intervention?
- Is a magnetic field map available for the experimental region?
  - What level of accuracy is required?

Reconstruction/Analysis Overview

- Java based reconstruction and analysis package
  - Runs standalone or inside Java Analysis Studio (JAS)
  - Fast MC $\rightarrow$ Smeared tracks and calorimetry clusters
  - Full Event Reconstruction
    - Beam background overlays at detector hit level, including time offsets.
    - detector readout digitization (CCD pixels, Si μ-strips, TPC pad hits)
    - *ab initio* track finding and fitting for ~arbitrary geometries
    - multiple calorimeter clustering algorithms
    - Individual Particle reconstruction (cluster-track association)
  - Analysis Tools (including WIRED event display)
  - Physics Tools (Jet Finding, Vertex Finding, Flavor Tagging)
- Write once run, run anywhere
  - Exact same libraries run on all platforms (Windows, Mac, Linux(es), Grid) using the Java Virtual Machine.

Tracking

- Analytic covariance matrices available for fast MC smearing for each detector.
- Track "cheater" available for studies of full detector simulation events. Assigns hits on basis of MC parentage.
- Ab initio track finding packages.
- Is fitting code incorporating multiple scattering and energy loss via weight matrix or Kalman Filter desirable?

Tracking Detector Readout

- Hits in Trackers record full MC information.
- Module tiling and signal digitization is deferred to analysis stage.
  - Used to rapidly study many possible solutions.
- Fully-featured package to convert MC hits in silicon to pixel hits. Fully configurable at runtime.

  MC Hits$\rightarrow$ Pixel ID & ADC$\rightarrow$ Clusters$\rightarrow$ Hits ($x \pm \delta x$)
- Can correctly study occupancies, overlaps, ghost hits, etc.

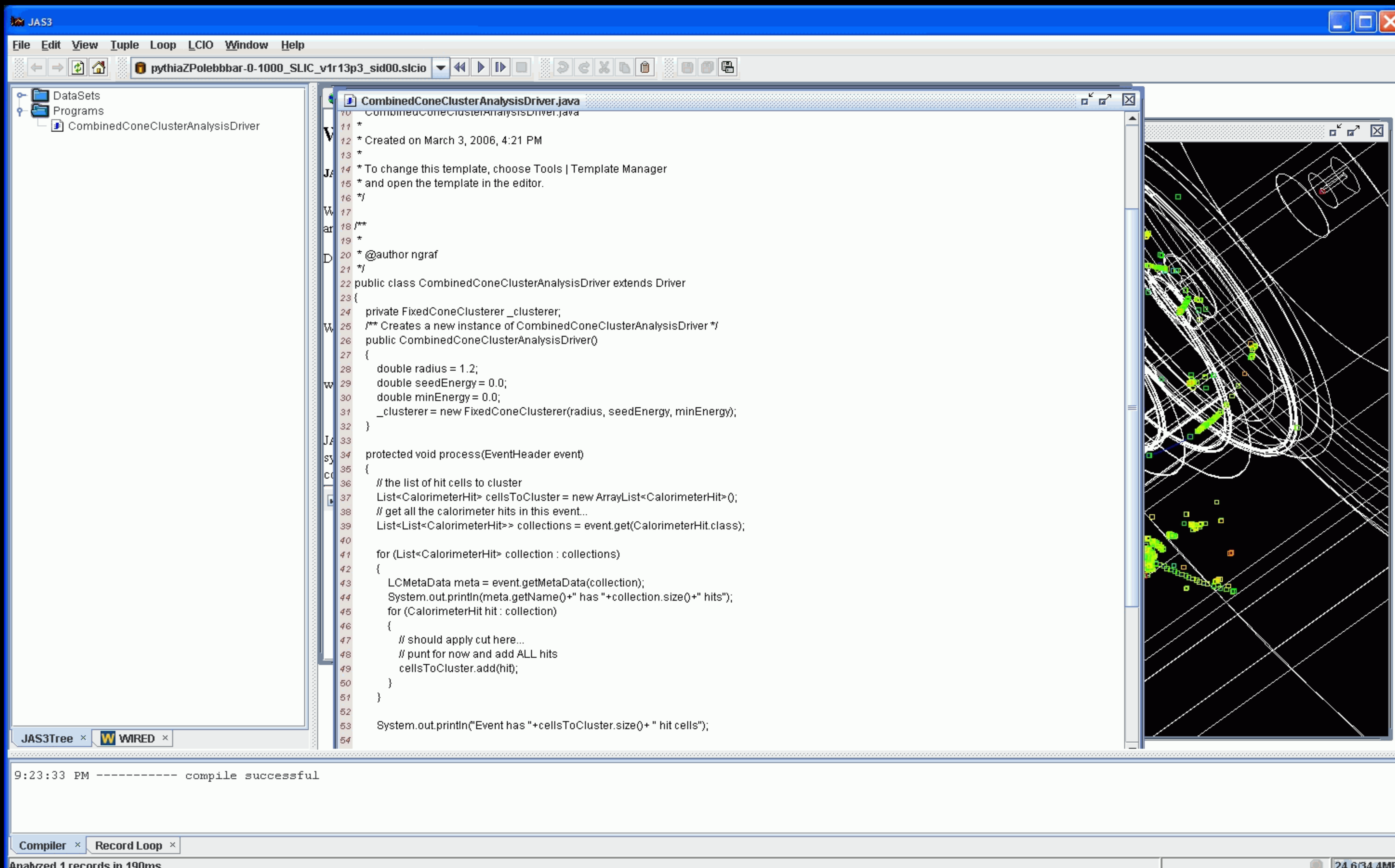
Calorimeter Reconstruction

- MC $\rightarrow$ ADC, time + channel ID $\rightarrow$ energy in cell
  - Depends on how complicated the response simulation needs to be.
  - Can get started with existing simulation change and improve as needed.
- CalorimeterHits $\rightarrow$ Cluster (energy + position)
  - Existing clustering algorithms most likely usable as-is.
  - New algorithms can be implemented as needed
- Track-shower association likely trivial compared to ILC or CLIC PFA.

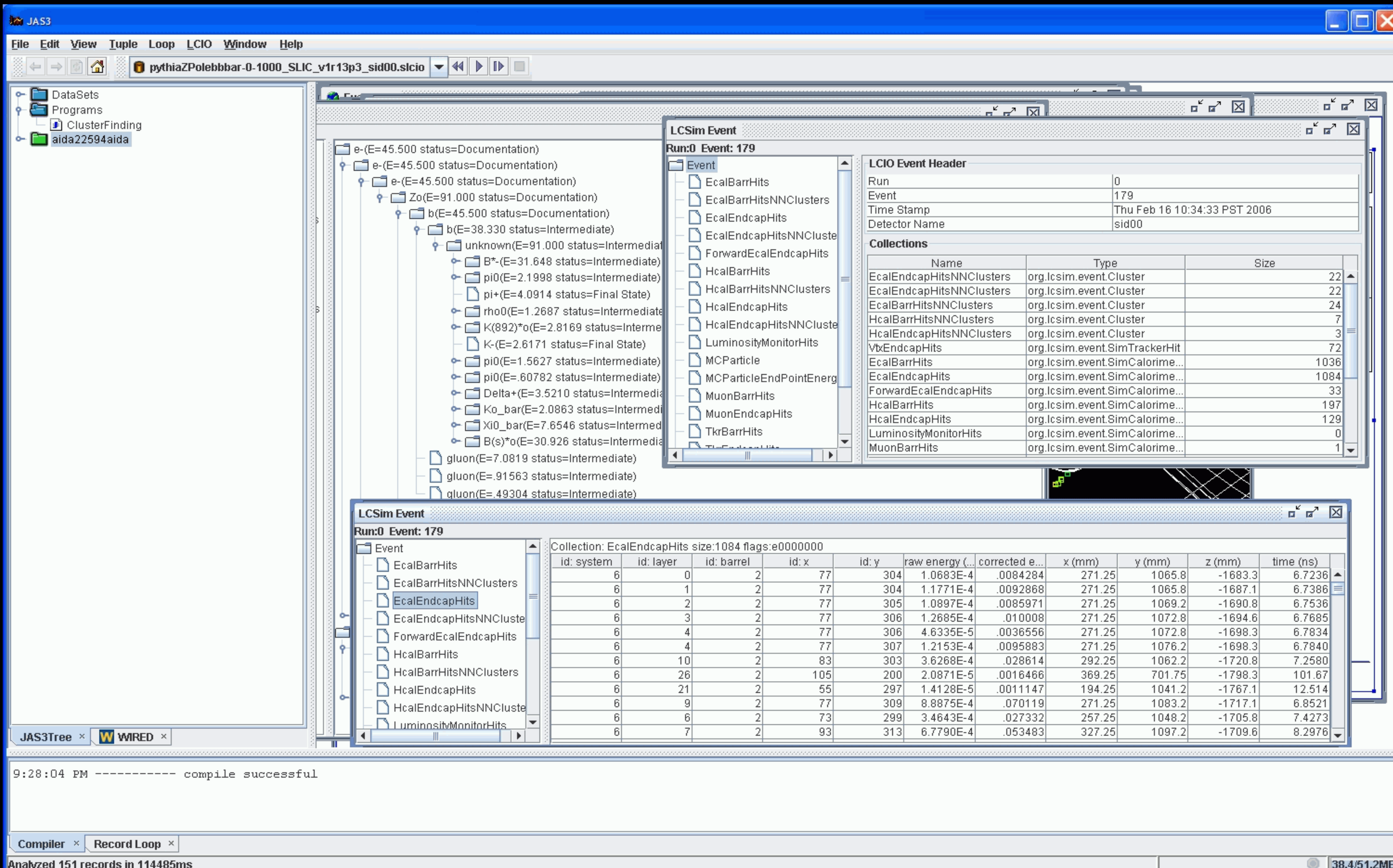
Java Analysis Studio (JAS)

- Integrated Development Environment (editor, compiler)
- Cross-platform physics analysis environment with iterative, event-based analysis model
  - quick development, debugging, ad hoc analysis
  - additional functionality with plugins
- Dynamically load / unload Java analysis drivers
  - Supports distributed computing.
- Plotting and fitting and analysis (cuts, scripting) engine
  - 1D, 2D histograms, clouds, profiles, dynamic scaling, cuts
  - high-quality output to vector or raster formats
- Integrated event browser and event display

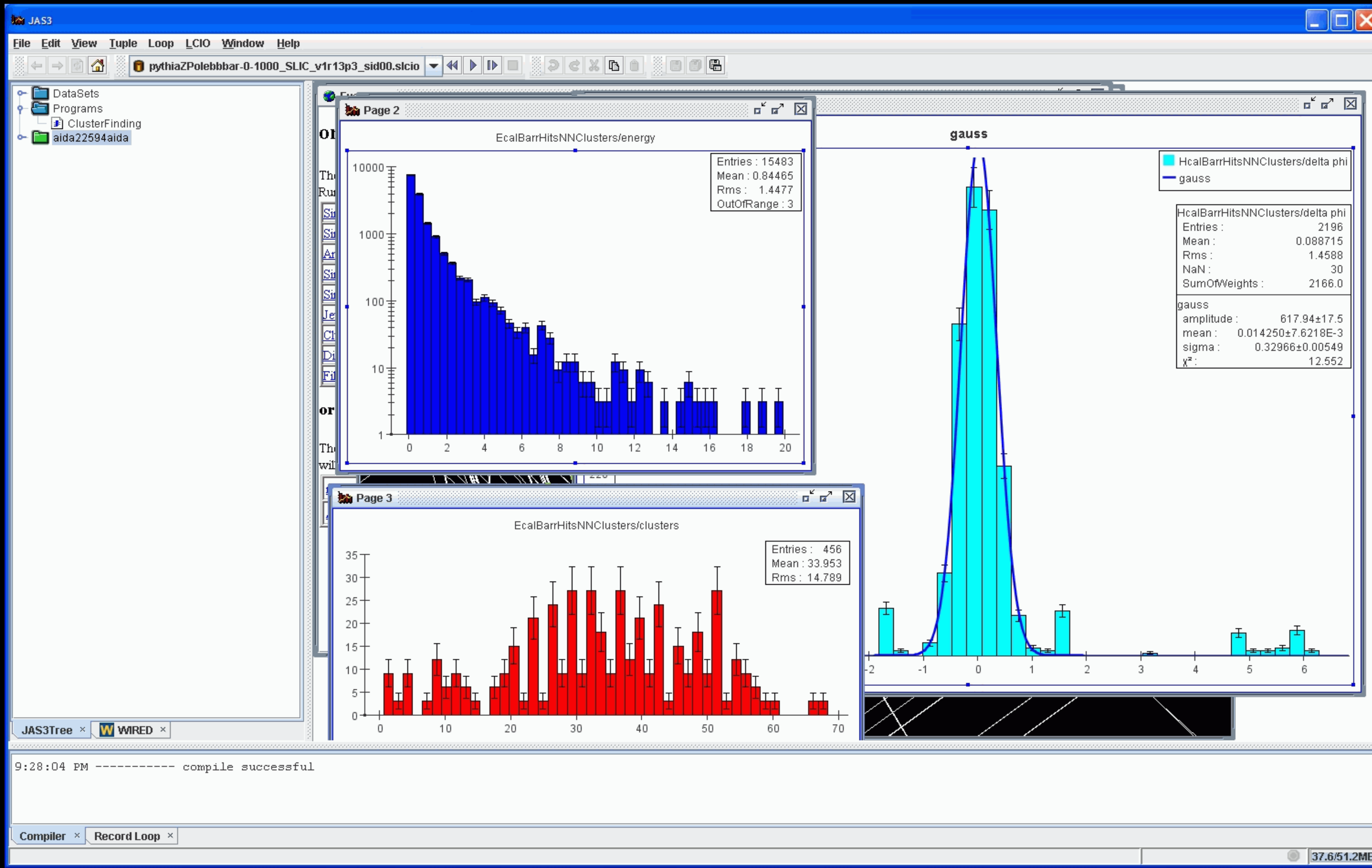
JAS editor/compiler



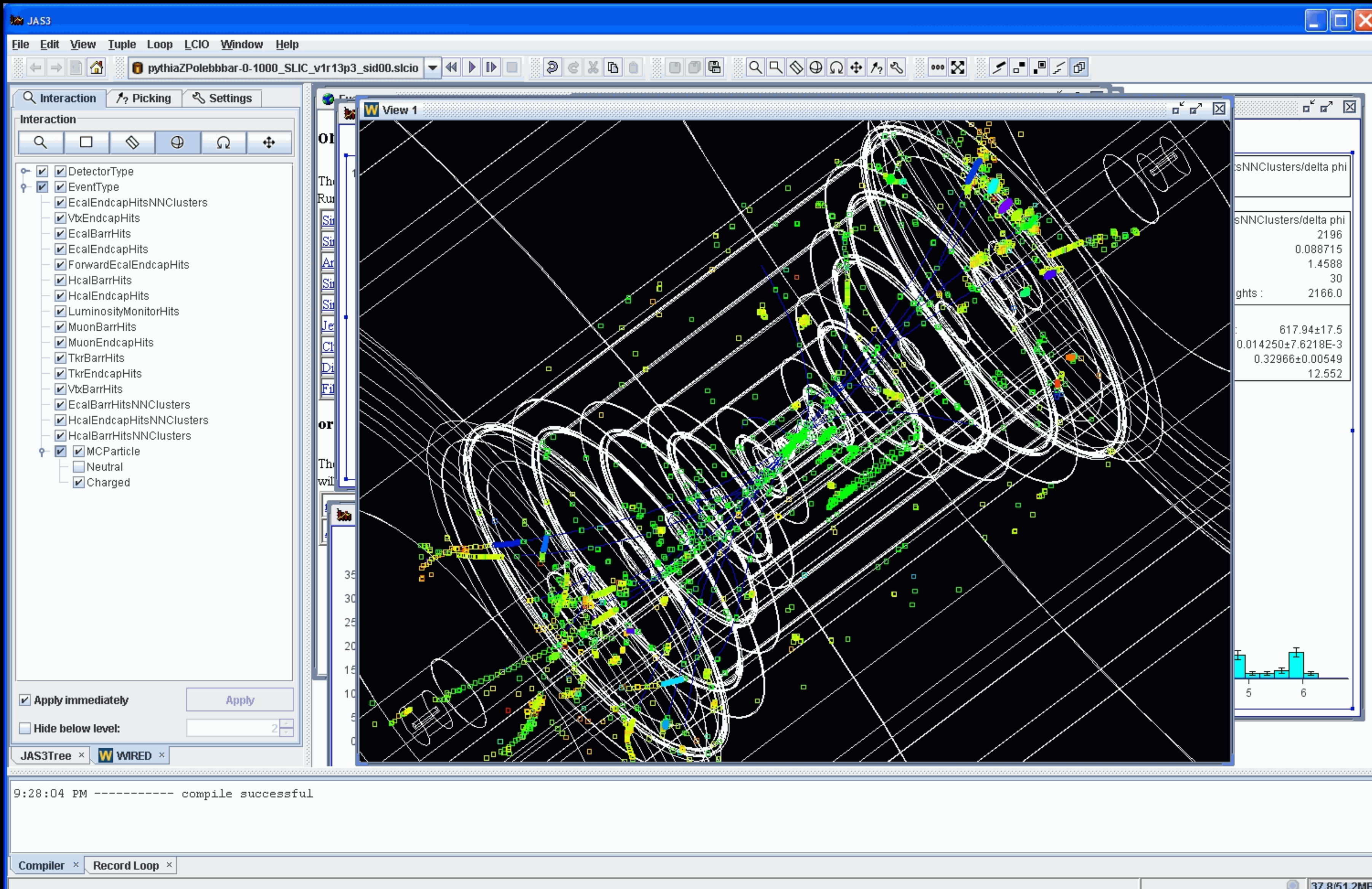
JAS event browser



JAS histogramming/ fitting



Wired LCD Event Display



JAS/Wired online

- Many components used by Babar and Fermi for online monitoring and data QA.
- Could be straightforwardly adopted for HPS needs.
- Tightly integrated system of sim/reco/online has many benefits.

Validated

- This suite of software tools provides:
  - Physics event generation & bindings to most legacy generators through the stdhep format.
  - Full detector response simulation using precompiled binaries & runtime geometry definition (no coding!).
  - Full detector digitization (x-talk, noise, diffusion, etc.)
  - Hit-level overlay of arbitrary background events.
  - Access to other LCIO-compliant software frameworks.
  - Full ab-initio event reconstruction and analysis suites.
  - Tested on hundreds of millions of events.
- "From zero to analysis in 15 minutes."

Using root

- Can analyze output AIDA files using RAIDA.
  - non-official root binding to AIDA
- Can analyze output LCIO files two ways:
  - Using root LCIO Dictionary
  - Using rlcio files, LCIO event data model written as root files
    - output files are larger
    - read times are longer
- Roll your own
  - write out native root files yourself.
    - neither recommended nor supported!

User base

- ILC physics and detector community
  - primarily US and UK members of SiD
- CLIC physics and detector community
  - CERN-based SiD’ studies
- MuC physics and detector community
  - FNAL-based
- JLAB heavy-photon search proposals
  - HPS?
  - DarkLight?
- FNAL dual-readout crystal calorimetry R&D group

Simulation Summary

- ALCPG sim/reco supports an ambitious international detector simulation effort. Goal is flexibility and interoperability.
- Provides a complete and flexible detector simulation package capable of simulating arbitrarily complex detectors with runtime detector description.
- Reconstruction & analysis framework was used to characterize the Silicon Detector and was essential to that concept's successful validation in the LOI process.
- LCIO provides interoperability with tools developed in other regions (e.g. jet flavor tagging (LCFI), particle flow (Pandora)), other languages (FORTRAN, java, C++, python) and other analysis frameworks (e.g. Marlin, root).

Additional Information

- Wiki - http://confluence.slac.stanford.edu/display/ilc/Home
- lcsim.org - http://www.lcsim.org
- ILC Forum - http://forum.linearcollider.org
- LCIO - http://lcio.desy.de
- SLIC - http://www.lcsim.org/software/slic
- LCDD - http://www.lcsim.org/software/lcdd
- JAS3 - http://jas.freehep.org/jas3
- AIDA - http://aida.freehep.org
- WIRED - http://wired.freehep.org