slic
A Geant4-based detector response simulation program

Norman Graf, Jeremy McCormick
(SLAC)

TIPP2014, Amsterdam
D-Day -1, 2014

Geant4

- Geant4 is the *de facto* HEP standard software for describing the interactions of particles with material and fields.
- Forms the basis for most, if not all, experimental detector response simulation packages.
- Distributed as a “toolkit” composed of C++ header and implementation files
  - Users write their own framework, handle geometry, sensitive detector response, event data model, I/O, etc.
  - Requires expertise in coding both C++ and Geant4
- Steep learning curve for smaller collaborations!

slic Mission Statement

- Develop and distribute an executable program which provides the full functionality and flexibility of the Geant4 toolkit but insulates the end user from having to write any C++ code.
- Need to be able to accommodate new detector geometries and readout technologies.
- The system should be flexible, powerful, yet simple to install and maintain.

Full Detector Response Simulation

- Use Geant4 toolkit to describe interaction of particles with matter and fields.
- Program provides access to:
  - Event Generator input (external file or gps)
  - Full Detector description input (more than just geometry)
  - Detector response output (hits or scoring)
- Geometries fully described at run-time!
  - In principle, as fully detailed as desired.
  - Uses lcdd, an extension of GDML.

Detector Definition

- Goal is 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.
- lcdd provides COMPLETE detector description

Why XML?

- Simplicity: Rigid set of rules
- Extensibility: easily add custom features, data types
- Interoperability: OS, languages, applications
- Self-describing data, validate against schema
- Hierarchical structure ↔ OOP, detector/subdetector
- Open W3 standard, lingua franca for B2B
- Many tools for validating, parsing, translating
- Automatic code-generation for data-binding
- Plain text: easily edited, cvs versioning

LCDD and GDML

- Adopted GDML as base geometry definition

GDML

- expressions (CLHEP)
- materials
- solids
- volume definitions
- geometry hierarchy

LCDD and GDML

- Adopted GDML as base geometry definition, then extended it to incorporate missing detector elements.

LCDD

- detector info
- identifiers
- sensitive detectors
- regions
- physics limits & cuts
- visualization
- magnetic fields

GDML

- expressions (CLHEP)
- materials
- solids
- volume definitions
- geometry hierarchy

LCDD Structure

`<lcdd>`	LCDD Root Element
`<header>`	Information about the Detector
`<iddict>`	Identifier Specifications
`<sensitive_detectors>`	Detector Readouts
`<limits>`	Physics Limits
`<regions>`	Regions (sets of volumes)
`<display>`	Visualization Attributes
`<gdml>`	GDML Root Element
`<define>`	Constants, Positions, Rotations
`<materials>`	Material Definitions
`<solids>`	Solid Definitions
`<structure>`	Volume Hierarchy
`</gdml>`	
`<fields>`	Magnetic Field
`</lcdd>`	

lcdd Features

- **Regions**: production cuts
- **Physics limi**ts: track length, step length, etc.
- **Visualization**: color, level of detail, wireframe/solid
- **Sensitive detectors**
  - calorimeter, optical calorimeter, tracker
  - segmentation
- **ID**s
  - volume identifiers (physvolid)
- **Magnetic field**s
  - dipole, solenoid, field map
- utilities
  - information on Geant4 stores
  - GDML load/dump

“Compact” Description

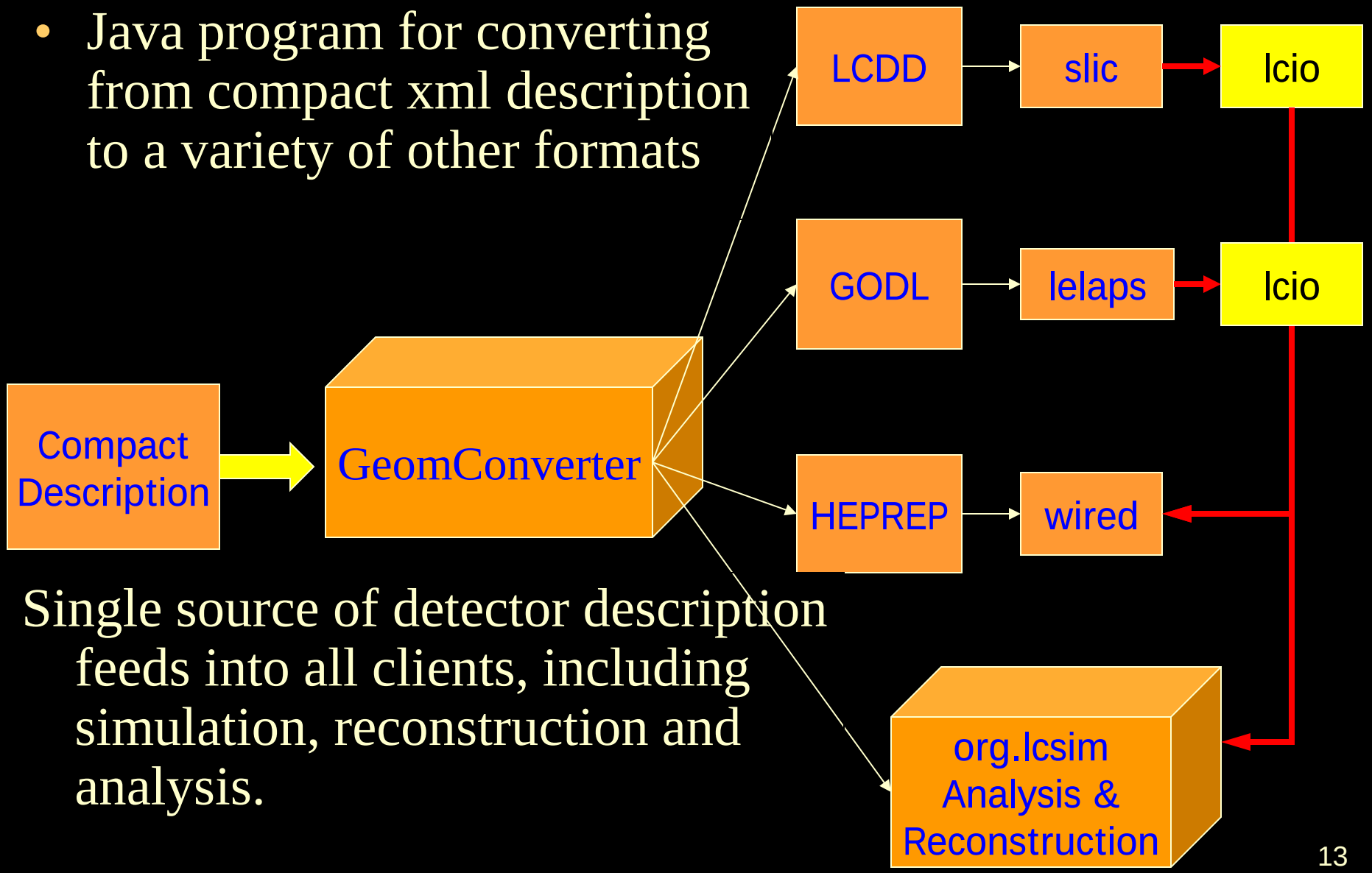
- The lcdd file is very descriptive, but therefore also very verbose.
- Can be written by hand, but prone to human error.
  - Also, specific to the simulation and not easily accessible to reconstruction and visualization.
- Developed a “compact” detector description which encapsulates the basic properties of a detector and which is further processed by code to produce the input specific to different clients.

Compact Detector Description

- A number of generally useful detector types (at least for HEP collider detectors) have been developed, such as:
  - Sampling calorimeters
  - TPCs
  - Silicon trackers
  - Generic geometrical support structures
- Can also incorporate GDML snippets
  - Allows inclusion of more complicated volumes derived for instance from engineering (CAD) drawings.

org.lcsim

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



Single source of detector description feeds into all clients, including simulation, reconstruction and analysis.

DD4hep

- Advanced European Infrastructure for Detectors at Accelerators (AIDA) WP2 Common Software deliverable.
- Full detector description package which supports simulation, reconstruction, analysis
- DD4hep uses the TGeo geometry classes to instantiate geometry tree, can use all TGeo features directly
- Generates lcdd file for input to slic

Compact Description - Example

```
<detector
    id=“3”
    name=“HADBarrel”
    type=“CylindricalBarrelCalorimeter”
    readout=“HcalBarrHits”
    vis=“HADVis”>
  <dimensions inner_r = "141.0*cm" outer_z = "294*cm" />
  <layer repeat=“40”>
    <slice material=“Steel235” thickness=“2.0*cm”/>
    <slice material=“RPCGasDefault” thickness=“0.12*cm”
      sensitive=“yes” region=“RPCGasRegion”/>
  </layer>
</detector>
```

global unique identifier

global unique name

detector type

readout collection

visualization settings

layering

absorber

sensitive layer

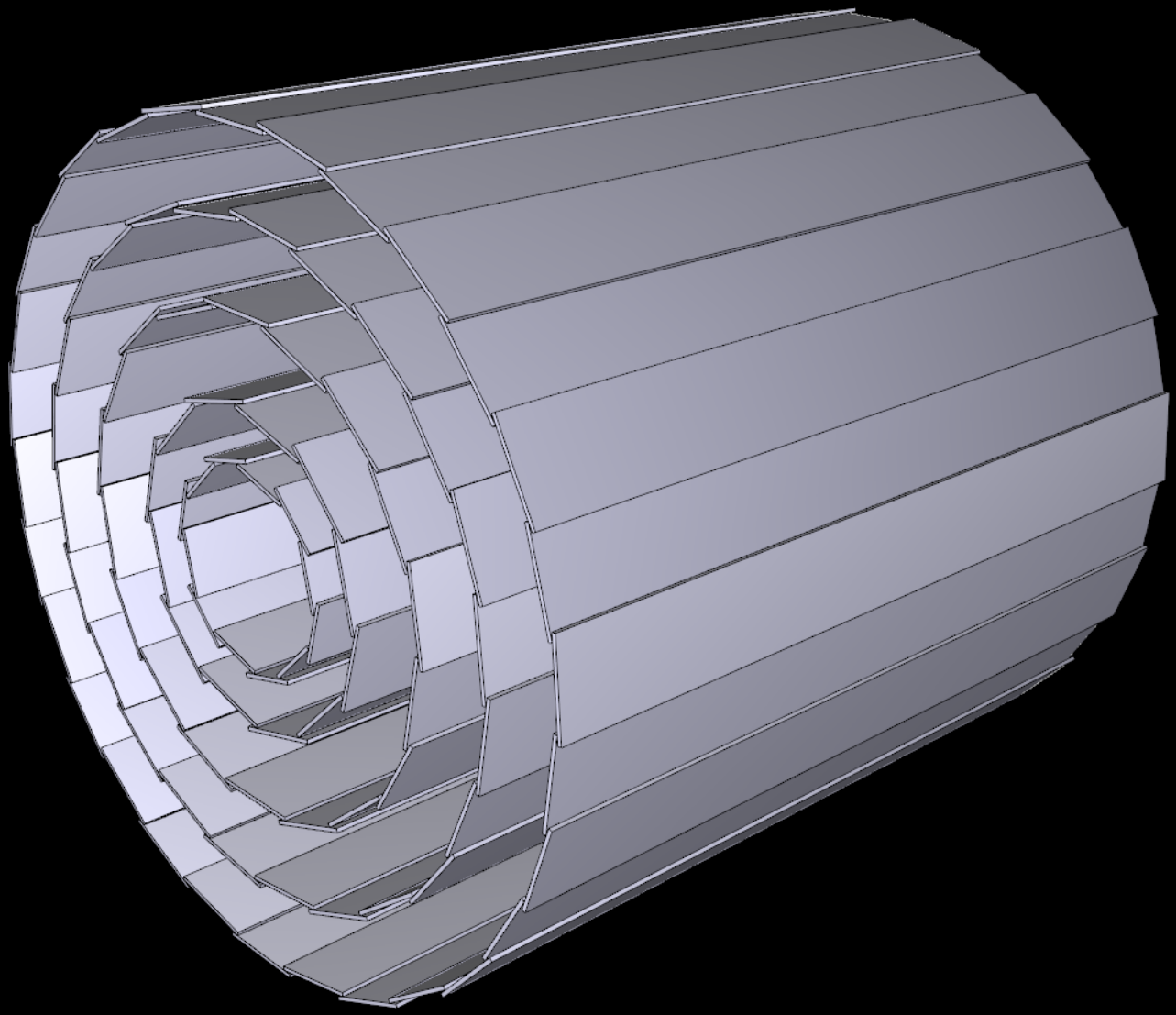
xml: Defining a Tracker Module

```
<module name="VtxBarrelModuleInner">
   <module_envelope width="9.8" length="63.0 * 2" thickness="0.6"/>
   <module_component width="7.6" length="125.0" thickness="0.26"
                     material="CarbonFiber" sensitive="false">
                     <position z="-0.08"/>
   </module_component>
   <module_component width="7.6" length="125.0" thickness="0.05"
                    material="Epoxy" sensitive="false">
                    <position z="0.075"/>
   </module_component>
   <module_component width="9.6" length="125.0" thickness="0.1"
                     material="Silicon" sensitive="true">
                     <position z="0.150"/>
   </module_component>
</module>
```

xml: Placing the modules

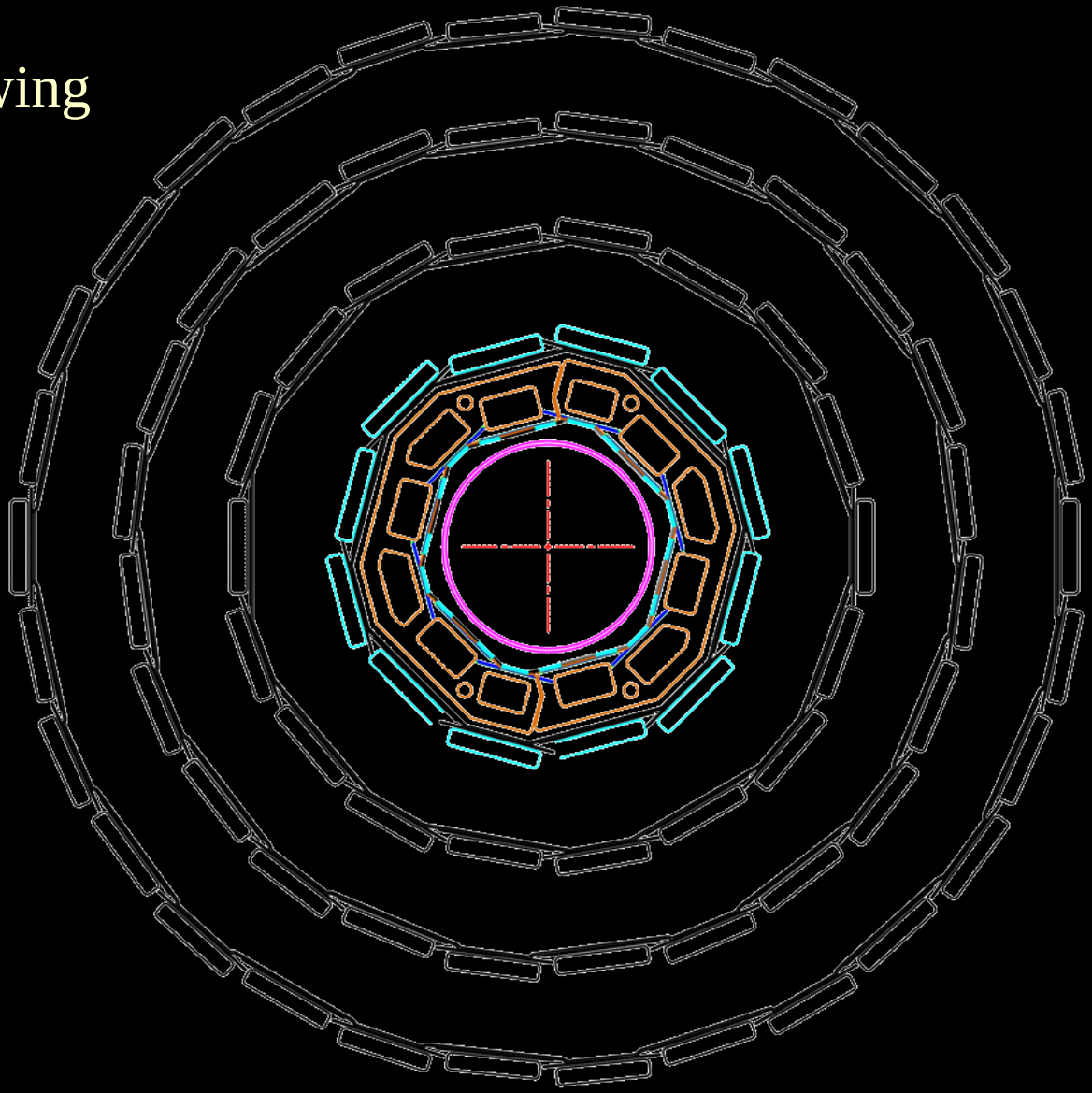
```
<layer module="VtxBarrelModuleInner" id="1">
        <barrel_envelope inner_r="13.0" outer_r="17.0" z_length="63 * 2"/>
        <rphi_layout phi_tilt="0.0" nphi="12" phi0="0.2618" rc="15.05" dr="-1.15"/>
        <z_layout dr="0.0" z0="0.0" nz="1"/>
</layer>
<layer module="VtxBarrelModuleOuter" id="2">
        <barrel_envelope inner_r="21.0" outer_r="25.0" z_length="63 * 2"/>
        <rphi_layout phi_tilt="0.0" nphi="12" phi0="0.2618" rc="23.03" dr="-1.13"/>
        <z_layout dr="0.0" z0="0.0" nz="1"/>
</layer>
<layer module="VtxBarrelModuleOuter" id="3">
        <barrel_envelope inner_r="34.0" outer_r="38.0" z_length="63 * 2"/>
        <rphi_layout phi_tilt="0.0" nphi="18" phi0="0.0" rc="35.79" dr="-0.89"/>
        <z_layout dr="0.0" z0="0.0" nz="1"/>
</layer>
<layer module="VtxBarrelModuleOuter" id="4">
        <barrel_envelope inner_r="46.6" outer_r="50.6" z_length="63 * 2"/>
        <rphi_layout phi_tilt="0.0" nphi="24" phi0="0.1309" rc="47.5" dr="0.81"/>
        <z_layout dr="0.0" z0="0.0" nz="1"/>
</layer>
<layer module="VtxBarrelModuleOuter" id="5">
        <barrel_envelope inner_r="59.0" outer_r="63.0" z_length="63 * 2"/>
        <rphi_layout phi_tilt="0.0" nphi="30" phi0="0.0" rc="59.9" dr="0.77"/>
        <z_layout dr="0.0" z0="0.0" nz="1"/>
</layer>
```

A Barrel Vertex Detector



Example Vertex Detector

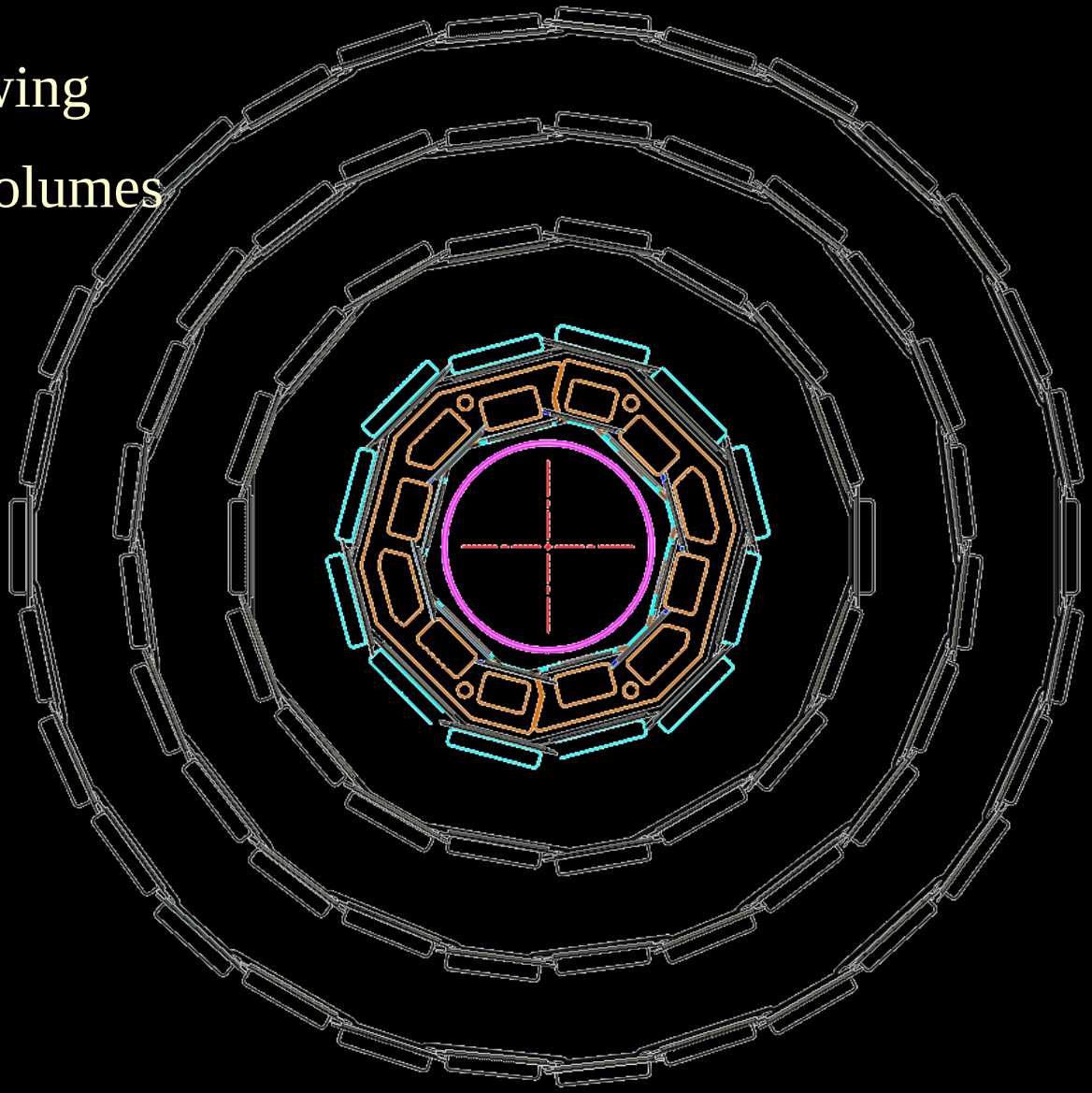
CAD Drawing



Example Vertex Detector

CAD Drawing

GEANT Volumes

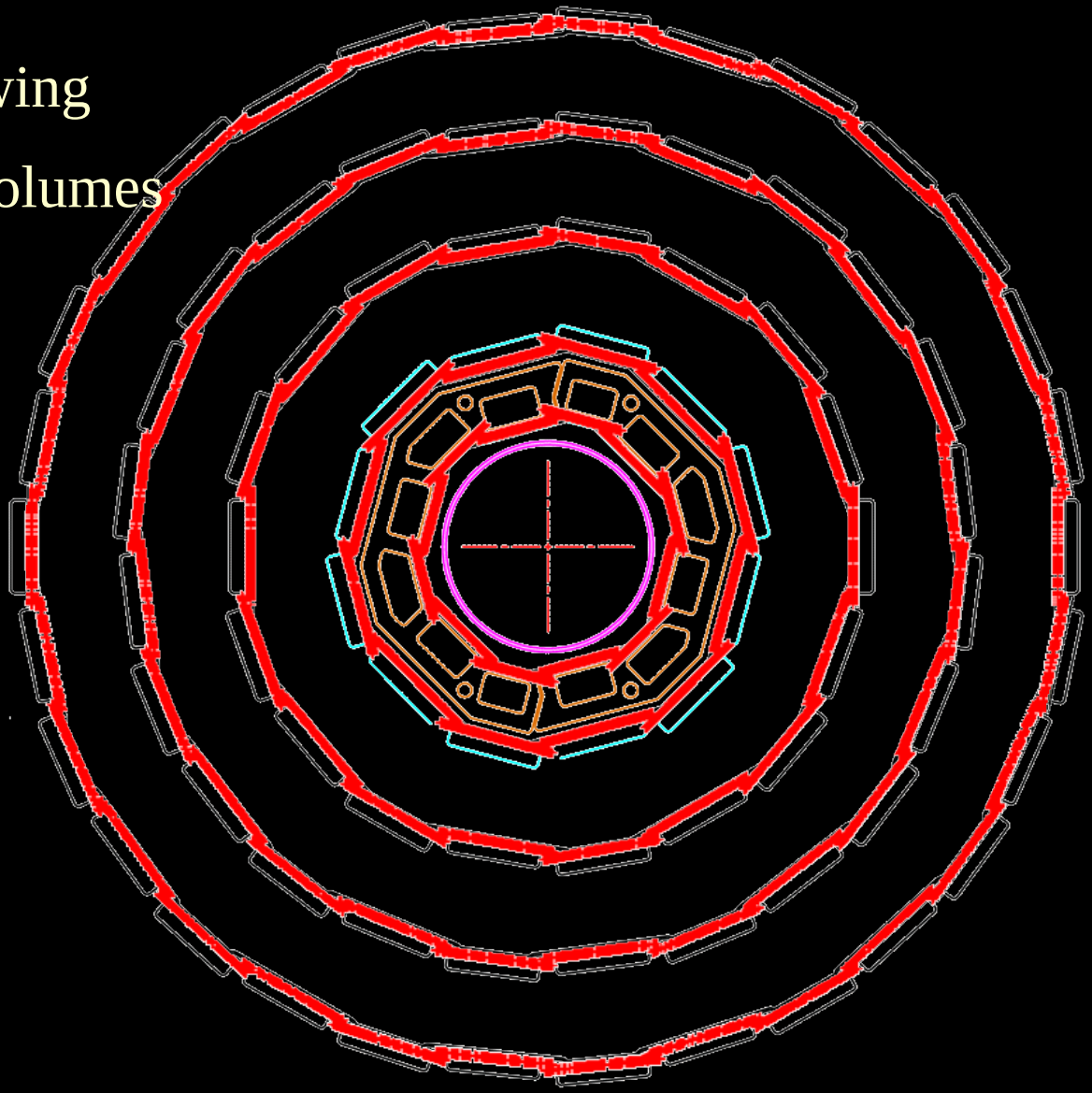


Example Vertex Detector

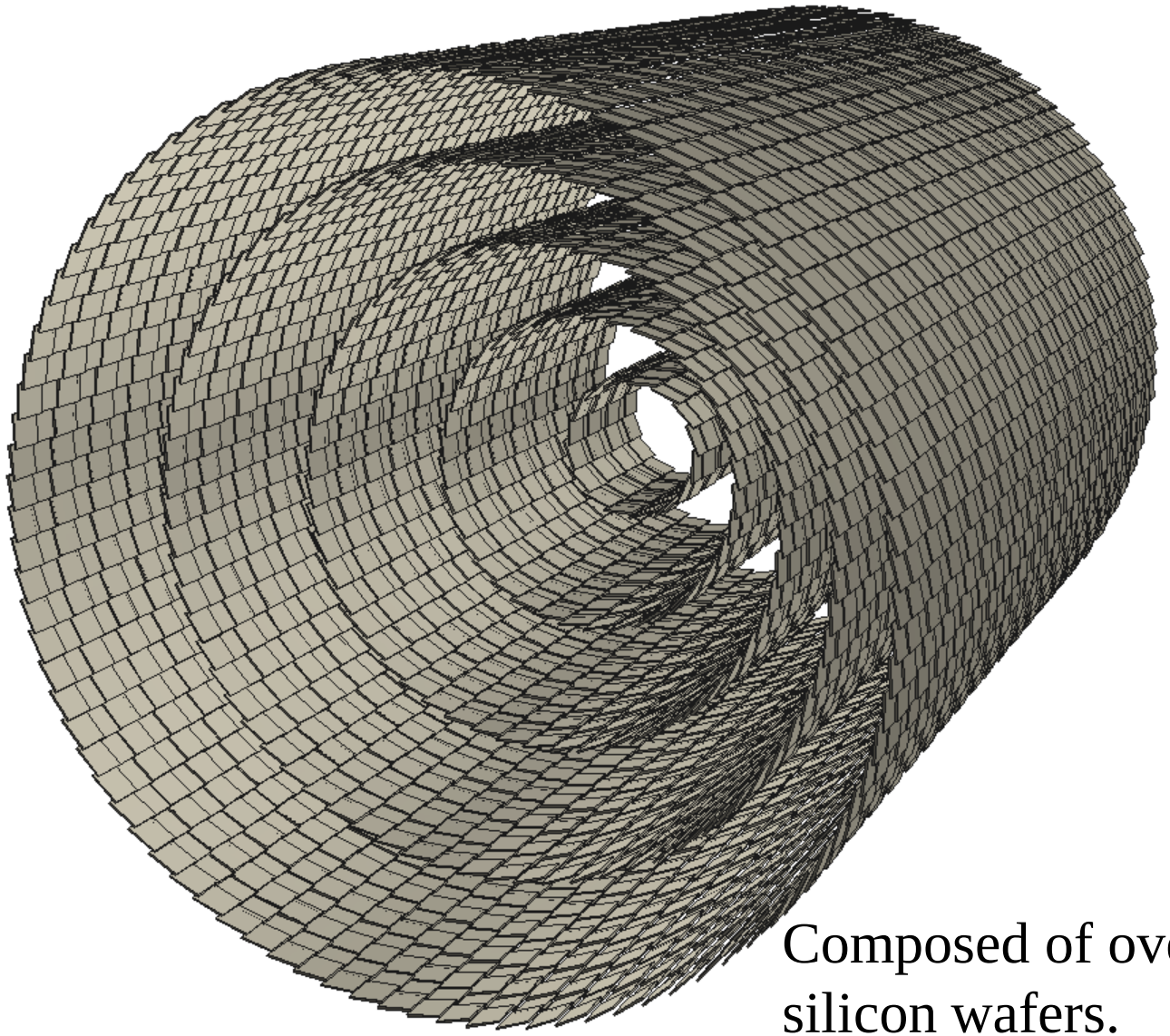
CAD Drawing

GEANT Volumes

LCIO Hits

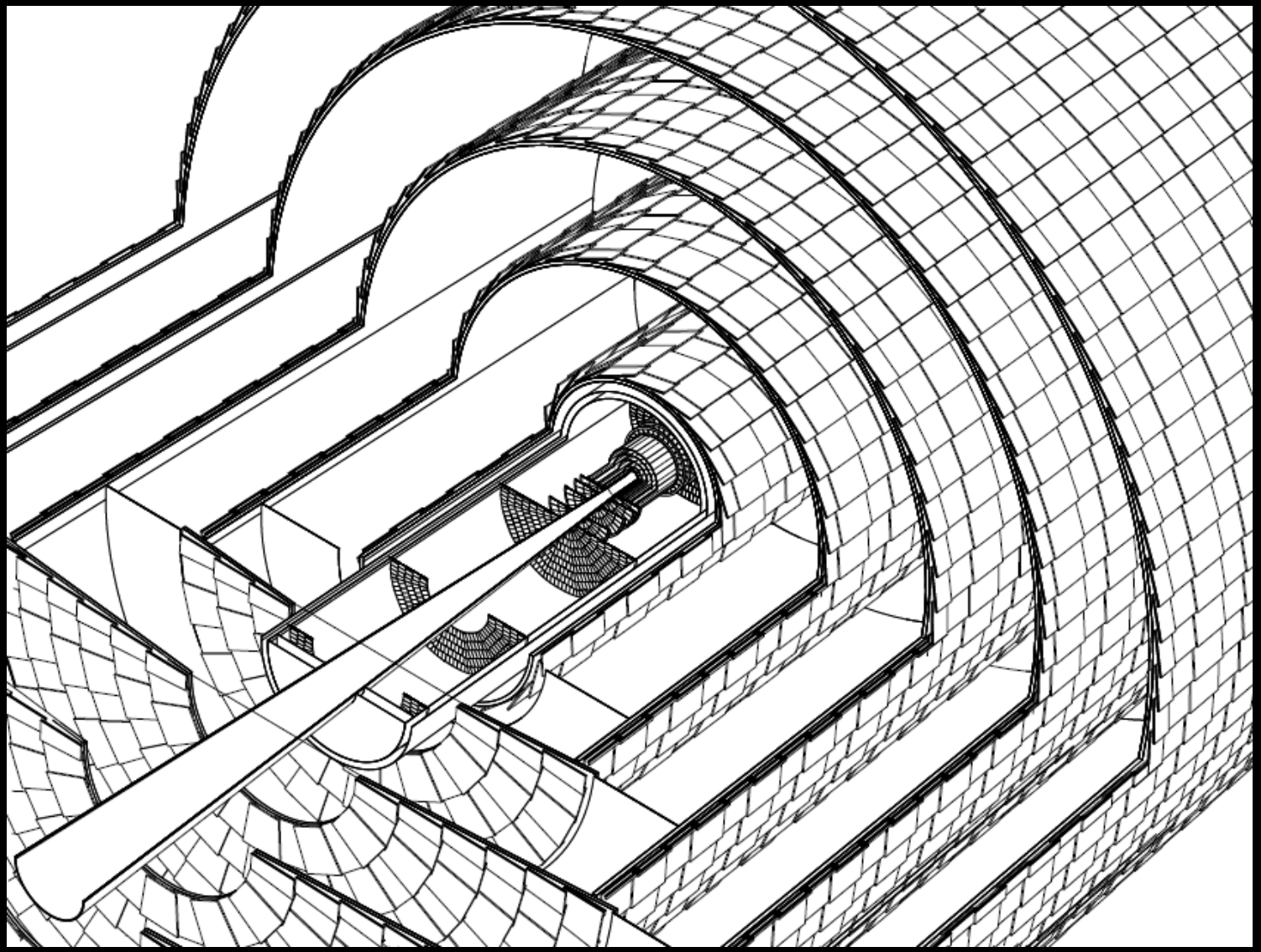


Barrel Outer Tracker



Composed of overlapping silicon wafers.

Complete Silicon Tracker



Generic Hits Problem Statement

- We wish to define a generic output hit format for full simulations of the response of detector elements to physics events.
- Want to preserve the “true” Monte Carlo track information for later comparisons.
- Want to defer digitization as much as possible to allow various resolutions, readout technologies, etc. to be efficiently studied.
  - single silicon wafer hits can be later processed as either strip or pixel, varying pitch, readout technology, etc.
- Can also use full machinery of Geant4 scoring surfaces via run-time macros

Types of Hits

- "Tracker" Hits
  - Position sensitive.
  - Particle unperturbed by measurement.
  - Save "ideal" hit information.
- "Calorimeter" Hits
  - Energy sensitive.
  - Enormous number of particles in shower precludes saving of each "ideal" hit.
  - Quantization necessary at simulation level.

Tracker Hit

- **MC Track handle**
- Encoded **detector ID** (detector dependent)
- **Hit position** in sensitive volume
- **Track momentum** at hit position.
- **Energy deposited** in sensitive volume.
- **Time** of track's crossing.
- **Path length** in sensitive volume.

Sufficient information to do hit digitization.

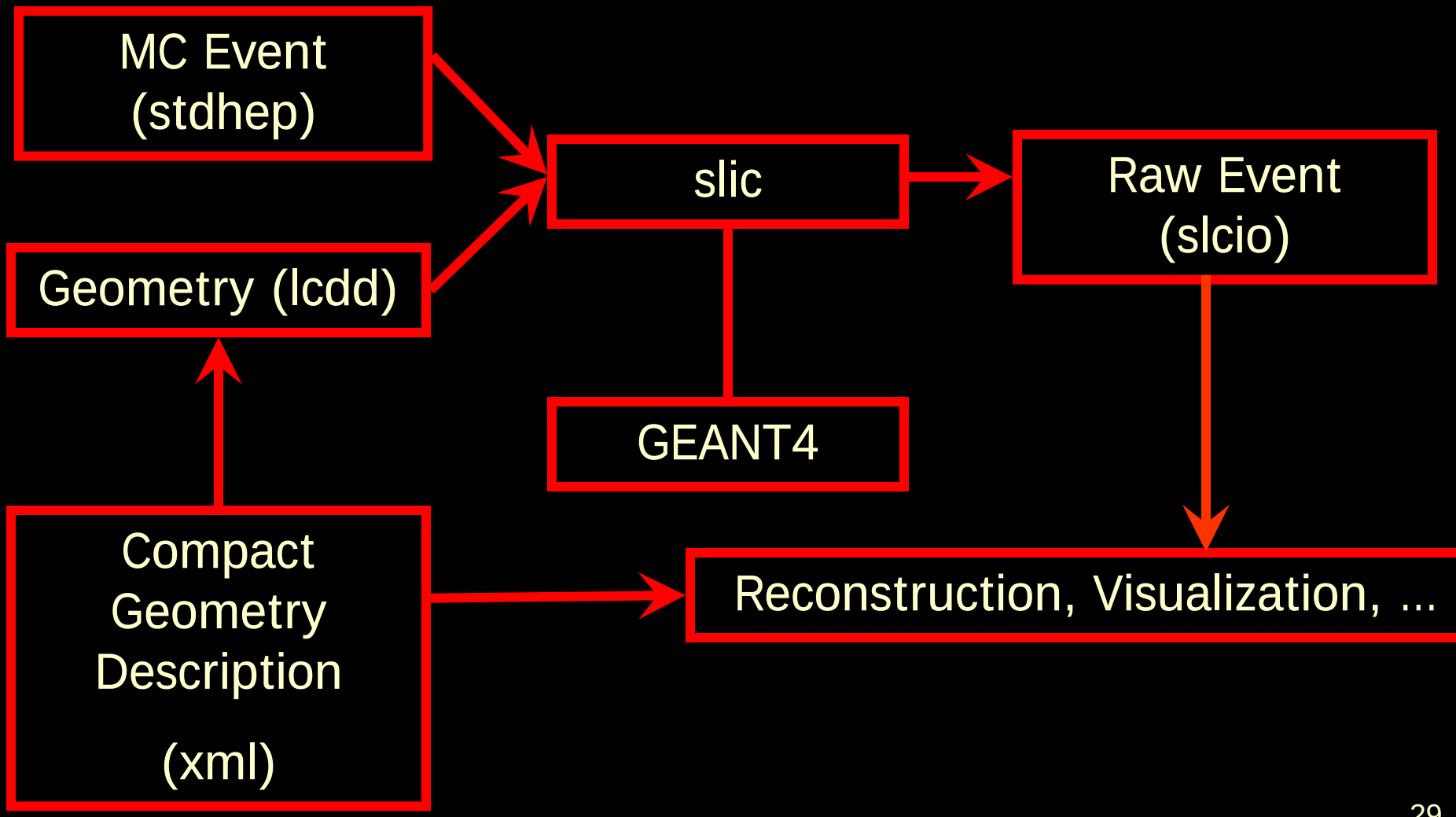
Calorimeter Hit

- Encoded detector ID (detector dependent)
- MC IDs for tracks contributing to this cell.
- Energy deposited.
- Time of energy deposition.
- Repeated for each energy contribution.
- Support recently added for optical calorimeters
  - Can store Cerenkov and scintillation light.

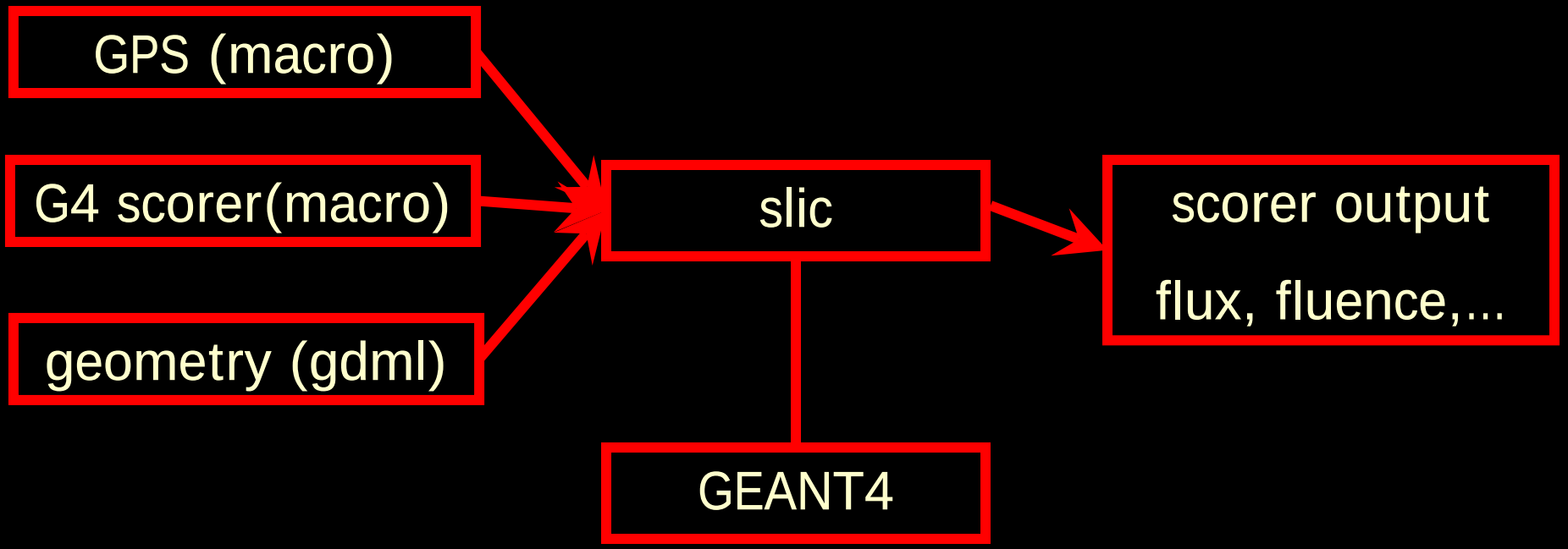
LCIO

- Persistency framework for LC simulations.
- Currently uses SIO: Simple Input Output
  - on the fly data compression
  - random access
  - C++, Java, python implementations available
- Changes in IO engine designed for (e.g. root).
- Extensible event data model
  - Generic Tracker and Calorimeter Hits.
  - Monte Carlo particle hierarchy.

Detector Full Simulation



Detector Full Simulation



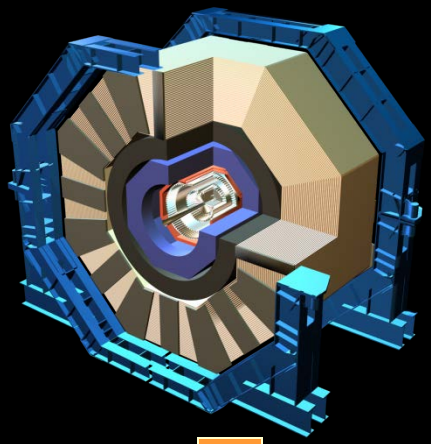
Detector Variants

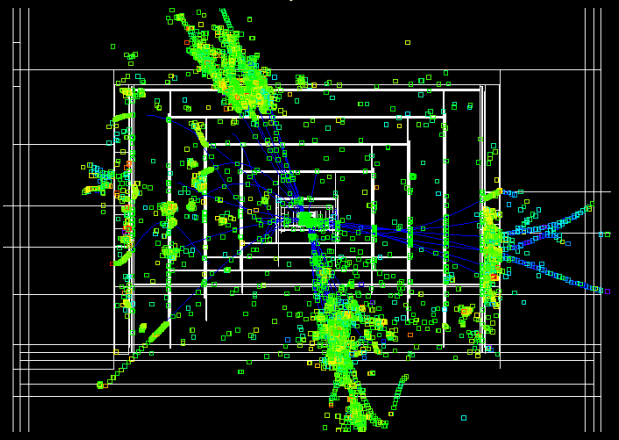
- Runtime XML format allows variations in detector geometries to be easily set up and studied:
  - Sampling calorimeters:
    - absorber materials, dimensions
    - Readout technologies, e.g. RPC, scintillator
    - Layering (radii, number, composition)
    - Readout segmentation (size, projective vs. nonprojective)
  - Total absorption crystal calorimeters
    - Optical properties
  - Tracking detector technologies & topologies
    - TPC, silicon microstrip, silicon pixels

ILC Full Detector Concepts

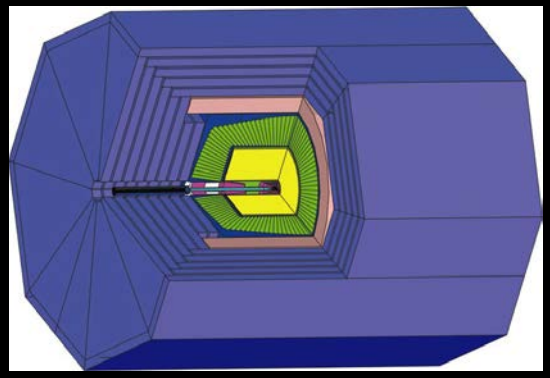
same event run with single executable with different input xml files

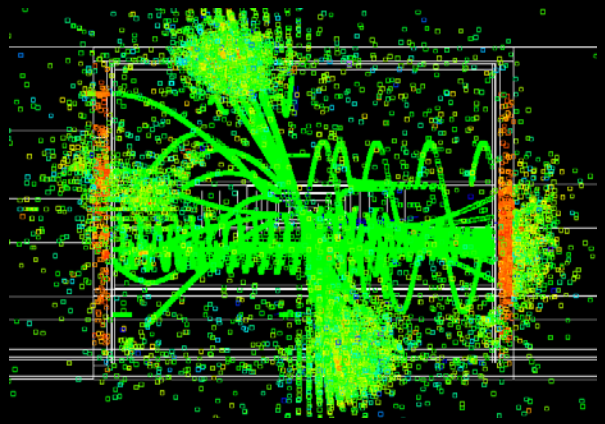
SiD



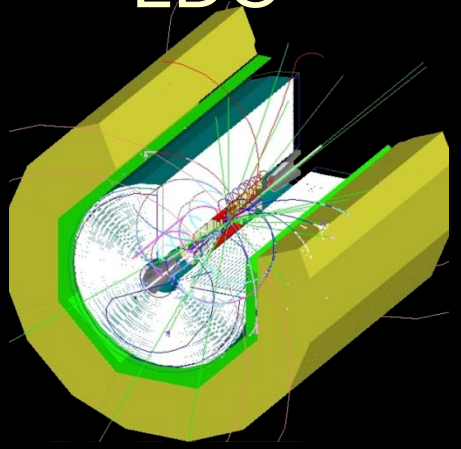


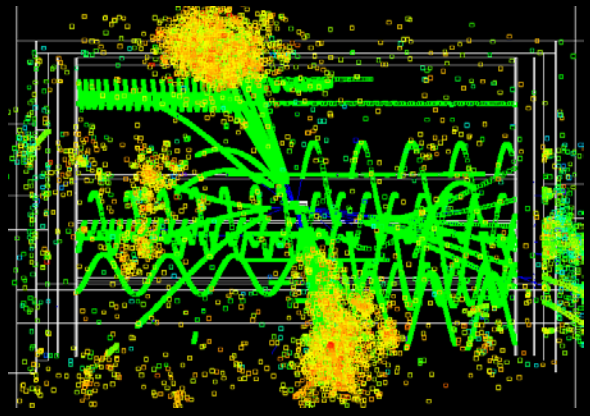
GLD





LDC





slic & lcdd: Summary

- Provides a complete and flexible detector simulation package capable of simulating arbitrarily complex detectors with runtime detector description.
- Being used by LC (ILC & CLIC) detector community for simultaneous and iterative evolution of different detector concepts and their variations.
- Being used by HPS experiment @ JLAB
- Could be used by other experimental collaborations or communities (astro, medical) for rapid prototyping or simulation.