SLAC-PUB-15297

lcsim: a Detector Response Simulation Toolkit

Norman A. Graf*, Jeremy McCormick

***Abstract*—As the complexity and resolution of particle detectors increases, the need for detailed simulation of the experimental setup also increases. Designing experiments requires efficient tools to simulate detector response and optimize the cost-benefit ratio for design options. We have developed efficient and flexible tools for detailed physics and detector response simulation which builds on the power of the Geant4 toolkit but frees the end user from any C++ coding. The primary goal has been to develop a software toolkit and computing infrastructure to allow physicists from universities and labs to quickly and easily contribute to detector design without requiring either coding expertise or experience with Geant4. Maximizing the physics performance of detectors being designed for the International Linear Collider (ILC), while remaining sensitive to cost constraints, requires a powerful, efficient, and flexible simulation, reconstruction and analysis environment to study the capabilities of a large number of different detector designs. The preparation of Letters Of Intent for the ILC involved the detailed study of dozens of detector options, layouts and readout technologies; the final physics benchmarking studies required the reconstruction and analysis of hundreds of millions of events. We describe the Java-based software toolkit (org.lcsim) which was used for full event reconstruction and analysis. The components are fully modular and are available for tasks from digitization of tracking detector signals through to cluster finding, pattern recognition, track-fitting, calorimeter clustering, individual particle reconstruction, jet-finding, and analysis. The detector is defined by the same input files used for the detector response simulation, ensuring the simulation and reconstruction geometries are always commensurate by construction. We discuss the architecture as well as the performance.**

I. Introduction

DETECTORS designed to exploit the physics discovery potential of lepton collisions at the Terascale will need to perform precision measurements of complex final states. One needs to fully reconstruct hadronic final states, with the ability to tag quark flavors with high efficiency and purity. Exceptional momentum resolution is required, leading to either large volume gaseous or low-mass silicon trackers. The excellent vertexing capabilities demanded by the flavor-tagging requirement point to a multi-layered giga-pixel vertex detector with micron point resolution. Calorimetry capable of very high di-jet mass resolution points towards highly segmented imaging calorimetry or to dual-readout total absorption crystals. The mission of the Linear Collider Detector (LCD) Simulation and Reconstruction group is to provide full simulation capabilities for the Linear Collider physics program, from physics event generation, to detector design simulations, through to event reconstruction and analysis. Its goal is to facilitate contributions from physicists in different locations with various amounts of available time. We do so by using standard data formats, providing a general-purpose framework for physics software development, and providing a suite of reconstruction and analysis code which is easy to install and use. Figure 1 presents a flowchart of the overall detector design, characterization and optimization process in which this software plays an essential role.

Manuscript received November 19, 2012. This work was supported in part by the U.S. Department of Energy.

N. A. Graf and Jeremy McCormick are with the SLAC National Accelerator Laboratory, Menlo Park, CA 94025 USA (* corresponding author, telephone: 650-926-5317, e-mail: Norman.Graf@slac.stanford.edu).

Maximizing the physics performance of detectors being designed for the ILC, while remaining sensitive to cost constraints, requires a powerful, efficient, and flexible simulation, reconstruction and analysis environment to study the capabilities of a large number of different detector designs. The preparation of Letters Of Intent and the currently ongoing Detailed baseline design (DBD) exercise for the ILC involved the detailed study of dozens of detector options, layouts and readout technologies; the final physics benchmarking studies required the reconstruction and analysis of hundreds of millions of events.

We describe the detector response program, slic, which we have developed to simulate the physics capabilities of several experimental concepts. It makes use of the full functionality of the Geant4 [1] toolkit, but defines the complete detector geometry at runtime, sparing the end user from having to develop any C++ code.

We also present the Java-based software toolkit (org.lcsim) which was used for full event reconstruction and analysis. The components are fully modular and are available for tasks from digitization of tracking detector signals through to cluster finding, pattern recognition, track-fitting, calorimeter clustering, individual particle reconstruction, jet-finding, and analysis. The detector is defined by the same xml input files used for the detector response simulation, ensuring the simulation and reconstruction geometries are always commensurate by construction. We discuss the architecture as well as the performance.

II. Modeling the Detector Response

As part of the American Linear Collider Physics Group's (ALCPG) [2] simulation and reconstruction effort, we set out to provide a full detector response simulation program which would free our end users from having to write code or to be expert in the details of the physics simulation to design and study a detector. The system should be powerful, yet simple to install and maintain and be flexible enough to accommodate new detector geometries and technologies. All of the detector properties, not just the geometry, should be definable at runtime with an easy to use format. The output simulated detector response should be made available in a simple, well-documented format, allowing further processing

Presented at IEEE 2012 Nuclear Science Symposium, Medical Imaging Conference
Anaheim, California, October 29 - November 3, 2012

Work supported by US Department of Energy contract DE-AC02-76SF00515.

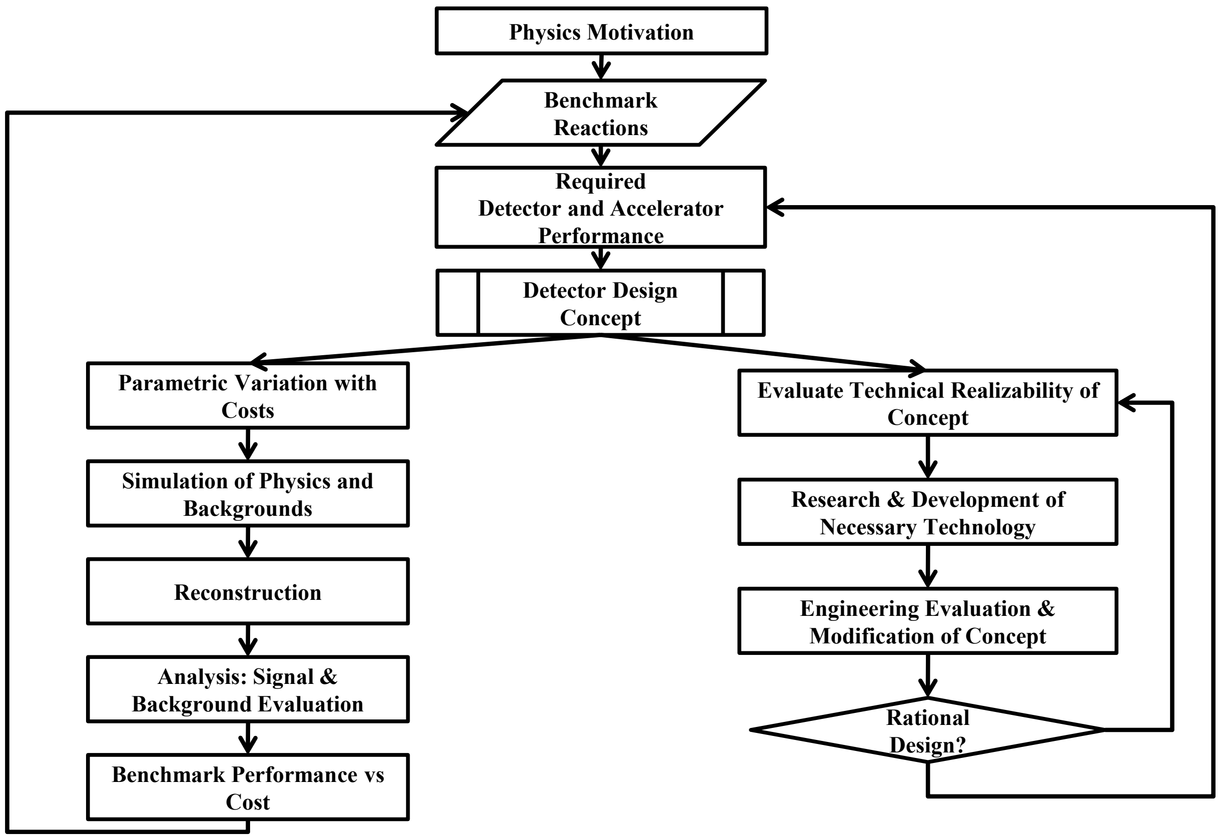
Physics Motivation
Benchmark Reactions
Required Detector and Accelerator Performance
Detector Design Concept
Parametric Variation with Costs
Simulation of Physics and Backgrounds
Reconstruction
Analysis: Signal & Background Evaluation
Benchmark Performance vs Cost
Evaluate Technical Realizability of Concept
Research & Development of Necessary Technology
Engineering Evaluation & Modification of Concept
Rational Design?

Fig. 1. A flow chart of the overall detector design, characterization and optimization process.

or event reconstruction to be undertaken with a minimum of effort.

A. Full Simulation using Geant4

The Geant4 [1] toolkit is the de facto high-energy physics standard for simulating the interaction of particles with fields and materials. However, the end user is normally required to write their own C++ program to access the libraries, and the learning curve for setting up the detector geometry and defining sensitive elements and readout can be quite daunting. We have developed the detector response package, slic [3], which is based on the Geant4 toolkit but defines generic input and output data formats, and uses an xml file format described in the next section to define the complete detector. This allows the end user to fully describe the detector geometry and readout at runtime using a plain text file. We provide executable programs for Windows, Mac OSX and several flavors of Linux. For persistence of the detector response, we developed LCIO [5], a simple event data model and persistency framework. The output from the detector response simulation consists of collections of generic tracker and calorimeter hits along with the complete Monte Carlo particle hierarchy, including secondaries produced in the simulation. LCIO is performant, with on-the-fly data compression and random access, well documented, with C++, Java, python and FORTRAN bindings.

B. Detector Description using GDML and lcdd

The detector geometry, typically the most complex part of a detector simulation, is described at runtime using an input text file rather than procedural C++ code. We selected XML as the textual file format for the geometry description for the following reasons:

- 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

Instead of developing a geometry package from the ground up, we extended the existing Geometry Description Markup

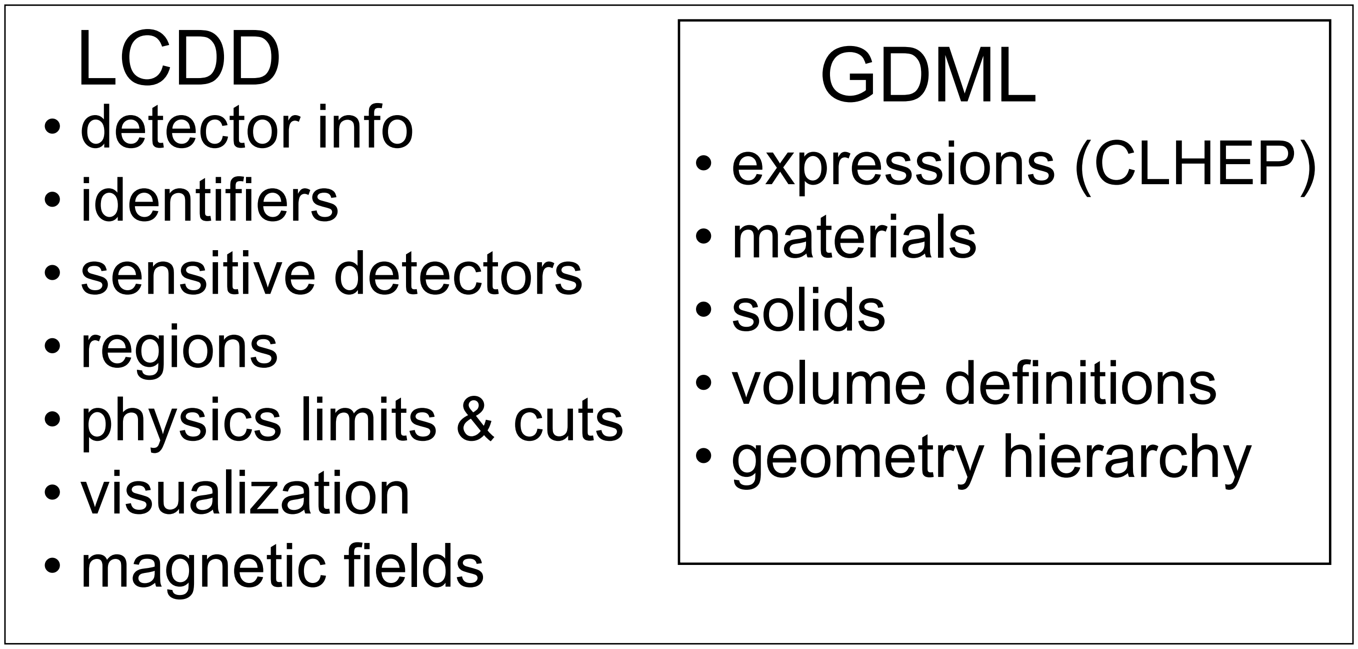

Fig. 2. The lcdd xml format, illustrating its relationship to GDML.

Language (GDML) [4].GDML is an application-independent geometry description format based on XML developed at CERN for high energy physics applications. It can be used as the primary geometry implementation language or it can provide a geometry data exchange format for existing applications. It provides expressions, materials, solids, volume definitions and a geometry hierarchy. We adopted this as the basic geometric description and extended it to include detector identifiers, definitions of sensitive detectors, regions, physics limits and cuts, physics list definitions and electromagnetic fields, either as simple parameterizations (e.g. solenoidal or dipole fields) or as field maps. This expanded, complete detector description language is known as lcdd [7] and its relationship to GDMl can be seen in Figure 2. It is composed of the following tags:

<**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**>

The full detector designs must be optimized for maximum performance while being constrained by a reasonable cost. At the level of the full detector, different combinations of subdetector parameters need to be studied. It may therefore be true that although any particular subdetector may not be optimal, the detector as a whole may have optimal performance. Although the geometry interface targets basic Geant4 shapes, and therefore any detector could be modeled, the focus of this particular presentation is on high energy physics collider detectors. Although modern collider detectors have converged on a common cylindrical topology, the exact composition of the trackers and calorimeters varies between concepts. Since the number of possible full detector designs that need to be modeled by the simulation software is quite large, detector simulation software for the International Linear Collider (ILC) [9] needs to provide capabilities that facilitate an iterative development cycle allowing rapid prototyping of these detector geometries, as shown in Figure 1. Changes to the detector geometry must not only be easy to make but also easily to propagate to all packages that depend on this information, from the simulator through reconstruction and analysis algorithms to the event display. The detector geometry should be treated as conditions; client programs need random access to full detector designs based on the current physics event.

The runtime XML format allows variations in detector geometries to be easily set up and studied.

- Absorber materials & readout technologies for sampling calorimeters (W, Cu, Fe + RPC, GEM, scintillator, ...)
- Optical processes for dual-readout or crystal calorimeters (Cherenkov, scintillation)
- Layering (radii, number, composition)
- Readout segmentation (cell size, projective vs. nonprojective)
- Tracking detector technologies (e.g. TPC, Si μstrip, Si pixel)
- Tracking detector topologies (e.g. nested vs Barrel + endcap)

A recent development allows the importation of models developed with CAD systems. This allows elements of arbitrary complexity to be easily incorporated into the detector design. However, there is a price to be paid in terms of computing time. Furthermore, these geometrical elements cannot be assigned as a sensitive element and are not available to the later stages of analysis, e.g. the reconstruction and event display. They are, therefore, best suited to support material. From the users perspective, the LCDD format can be too verbose and complex for hand editing. Typically, researchers want to change a few parameters such as inner radius or layering scheme and have these changes automatically propagated to create the detailed full geometry. The Compact Detector Description, henceforth referred to as the compact description, is a high-level format designed to facilitate this type of usage. The GeomConverter Java package converts compact descriptions to LCDD. It also supports the geometry formats used by the fast simulation programs and the event display. It can also create Java runtime objects representing the detector geometry for the org.lcsim reconstruction and analysis framework. The following is a simple example of a compact description of a sampling Si/W barrel calorimeter:

```
<detector
      id="2"
      name="EMBarrel"
      type="CylindricalBarrelCalorimeter"
      readout="EMHits">
<dimensions
      inner_r="150.1*cm"
      outer_z="208.0*cm" />
<layer repeat="20">
   <slice material="Tungsten"
          thickness="0.25*cm" />
   <slice material="Silicon"
          thickness="0.032*cm"
          sensitive="yes" />
</layer>
</detector>
```

The task of expanding this into the twenty layers of absorber and readout, positioning each layer correctly and assigning a sensitive volume to each layer of silicon would be handled by the GeomConverter package, freeing the end user to concentrate on the essential topological description of the detector. Extremely complicated detector geometries can be simulated quite easily with the compact description. And since the geometry is completely defined at runtime, it is straightforward to study a number of detectors with a minimum of effort. Figures 3 through 8 show the same physics event being simulated with three different ILC detector concepts using the same executable program. The only difference was the input detector geometry description file.

C. I/O

For persistence of the detector response, we developed LCIO [5] , a simple event data model and persistency framework used by the ILC community. Because there is no efficient way of encapsulating all of the myriad sensitive detector technologies in a single application, we chose to write out the simulated deposition of energy in sensitive detectors, and allow the end user to apply all digitization (e.g. development of the signals and electronic response) at the reconstruction or analysis stage. High energy physics collider detectors, although incredibly complex, can be broken down into two distinct types of measurement devices: non-destructive, position-sensitive devices usually referred to as trackers, and destructive, energy-sensitive detectors identified as calorimeters, information from which is usually measured only in aggregate, i.e. within some reasonably large readout volume. The output from the detector response simulation then consists simply of collections of generic tracker and calorimeter hits along with the complete Monte Carlo particle hierarchy, including secondaries produced in the simulation. LCIO is performant, with on-the-fly data compression and random access, well documented, with C++, Java, python and FORTRAN bindings. The software architecture is shown schematically in Figure 9.

III. Event Reconstruction

The Java based toolkit org.lcsim [6] is used for full event reconstruction and analysis. Java provides both a very powerful object-oriented language for development and transparent cross-platform portability. The code can be run standalone or within the Java Analysis Studio (JAS3) an integrated development environment. The reconstruction and analysis software is based on a plug-and-play architecture which employs Drivers to perform discrete actions. Drivers interact with the Event by fetching objects or collections from the Event, processing them, and optionally adding results back to the Event. As currently implemented, the event reconstruction is done serially by applying the run-time configurable list of event processors one after the other. However, the Java language provides excellent support for multi-threaded computing, and nascent studies of a parallel reconstruction paradigm are very promising.

The Driver class interface provides the following methods:

- `startOfData()`
- `getConditionsManager()`
- `process(EventHeader  event)`

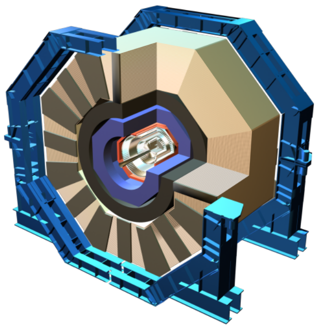
Fig. 3. A view of the SiD concept.

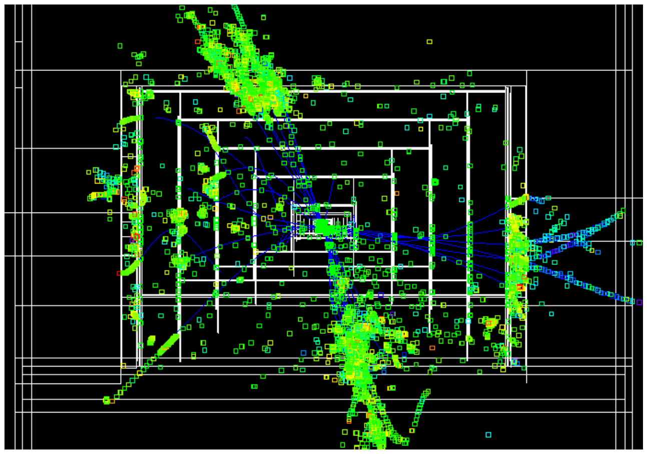
Fig. 4. A multijet event simulated in the SiD concept.

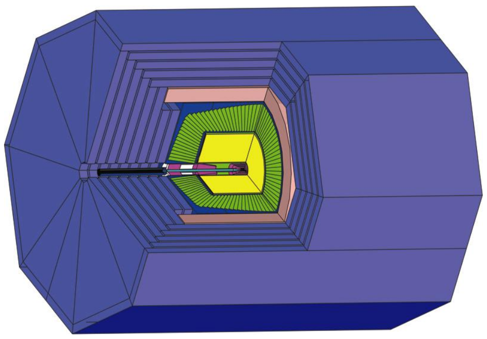
Fig. 5. A view of the ILD concept.

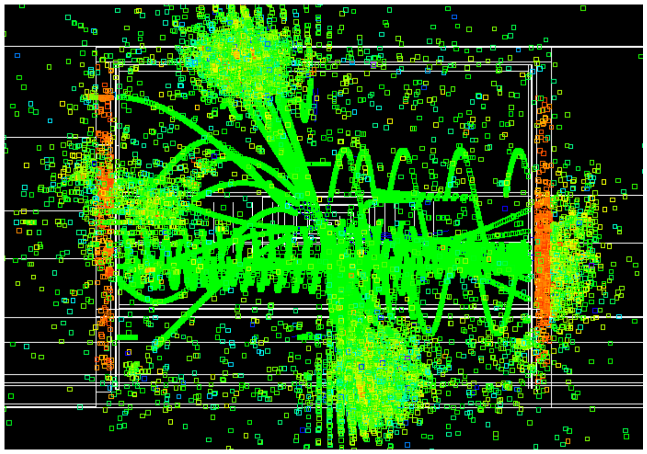
Fig. 6. A multijet event simulated in the ILD concept.

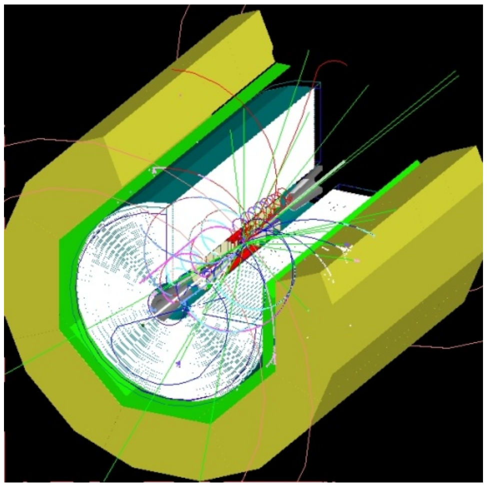
Fig. 7. A view of the GLD concept.

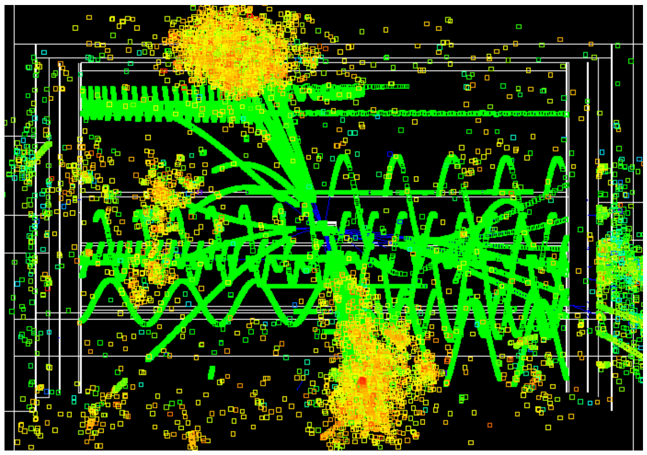
Fig. 8. A multijet event simulated in the GLD concept.

- `detectorChanged(Detectordetector)`
- `endOfData()`

although, in practice, most user classes only override the `process(EventHeader event)` method. The code runs either within the Java Analysis Studio (JAS) IDE for interactive use or standalone for batch or Grid production. The "write-once, run anywhere" feature of Java means that the exact same libraries run on all platforms (Windows, Mac, Linux(es) using the Java Virtual machine. A number of packages provide such functionality as: overlaying beam backgrounds at the detector hit level (including time offsets), digitization of readout electronics (CCD pixels, silicon mi-

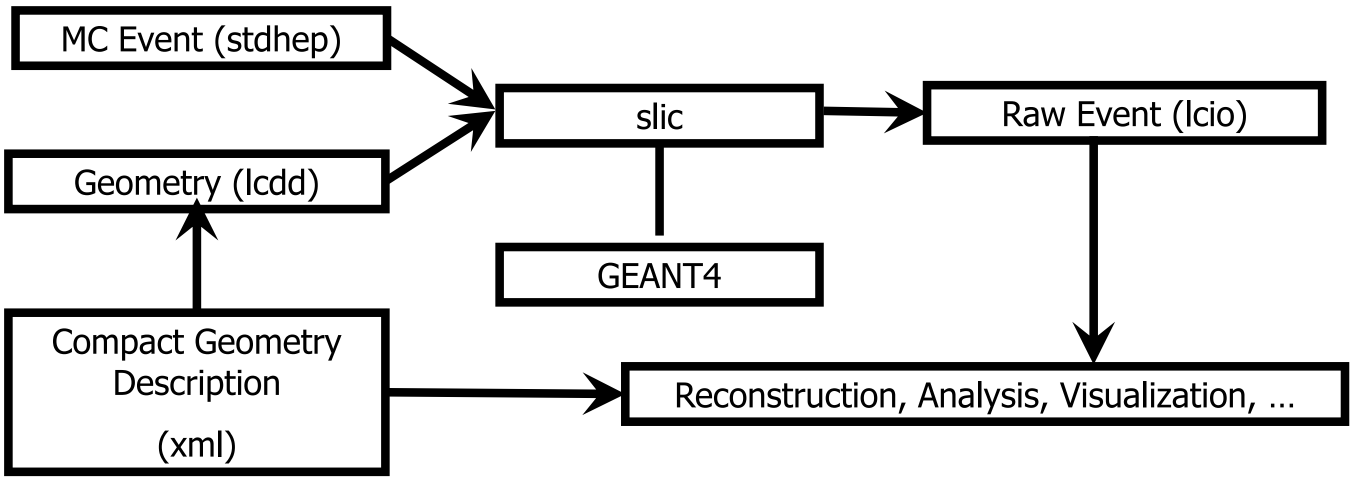
MC Event (stdhep)
Geometry (lcdd)
Compact Geometry Description (xml)
slic
GEANT4
Raw Event (lcio)
Reconstruction, Analysis, Visualization, ...

Fig. 9. The simulation software architecture.

crostrips, TPC pad hits), *ab initio* track finding and fitting, multiple calorimeter clustering algorithms, and individual particle reconstruction, also known as Particle Flow Analysis (PFA). Data analysis tools include an event browser and the Wired 3D interactive event display, whereas physics analysis tools include jet finding, event shape, vertexing and particle ID algorithms.

A. Monte Carlo Hit Digitization

Because we are interested in studying the effects of many tracking detector parameters, we chose not to perform the detector digitization during the Geant4 response simulation. Instead, we write out the simulated deposition of energy in sensitive detectors, and allow the end user to apply all digitization (e.g. development of the signals and electronic response) at the reconstruction or analysis stage. This allows us to rapidly and efficiently study the effects of pixel size and strip pitch as well as various readout technologies without having to rerun events through the full Geant4 simulation. After the effects of drift and diffusion of deposited charge is simulated, the electronics simulation adds noise, propagates the signal to the readout, applies a threshold and gain, and provides “raw” data in the form of channel IDs and ADC counts. Various clustering algorithms are provided which perform 1D clustering for strips and 2D clustering for pixel detectors. The clustering provides hit centroids and cluster-size dependent uncertainties which then serve as input to track finding and fitting.

B. Track Finding and Fitting

A number of track finding strategies are available within org.lcsim, supporting standalone pattern recognition for one-dimensional (Si micro-strip), two-dimensional (Si pixel) and fully three-dimensional (TPC) hits. The efficiency for finding tracks is very high, even in the presence of backgrounds and at low momenta. Conformal-mapping pattern recognition is also available for 3D hits, applicable to a TPC. Pattern recognition initiated by extending track-stubs found in the highly segmented calorimeters into the trackers is also provided. Track fitting incorporating multiple scattering and energy loss via a weight matrix approach is currently used, with a full Kalman Filter approach becoming available.

C. Calorimeter Reconstruction

A number of clustering algorithms have been implemented which are used to support the Particle Flow Algorithm (PFA) approach to event reconstruction. One attempts to unambiguously associate all clusters in the calorimeter to their orginating particle. For charged particles, one then uses the measured track momentum to replace the much less precise energy measurement. Clusters unassociated with tracks are either photons, measured reasonably well in the electromagnetic calorimeters, or neutral hadrons, which only represent a fraction of the total event energy. Using this technique, individual jet energy resolutions of three to four percent can be achieved.

IV. Job Control

Exploring large regions of detector design phase space requires an easy way to configure not only the detector geometry, but also the reconstruction processors and their associated control parameters. Again, we have adopted an xml format for our run control steering files. In it we can not only define the input and output files, but also the number of events to process, the Drivers to run, along with arguments to any of the algorithms.

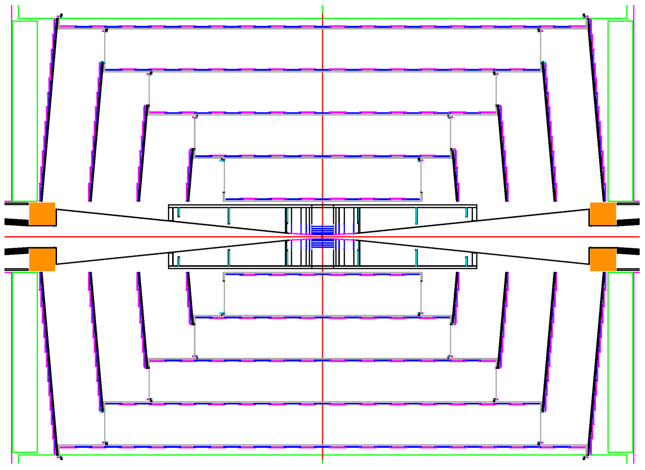
Fig. 10. A cross section of the SiD central tracking region as implemented in CAD software.

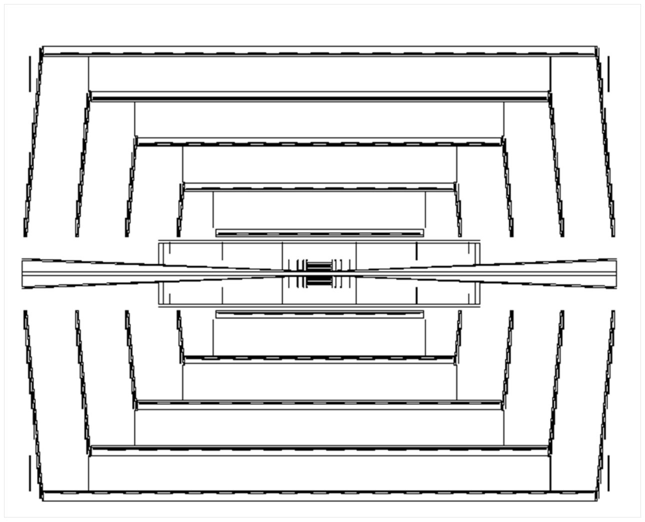
Fig. 11. A cross section of the SiD central tracking region as implemented in Geant4.

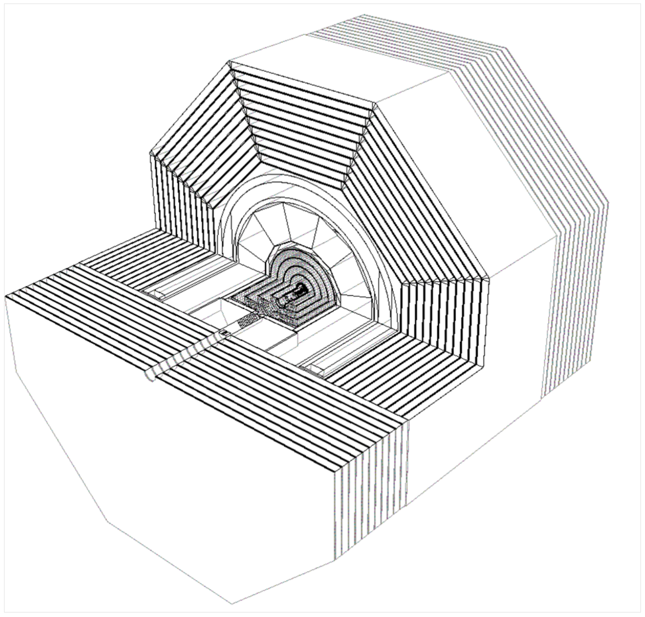
Fig. 12. A cutaway view of the Silicon Detector as implemented in Geant4 using the compact.xml file description.

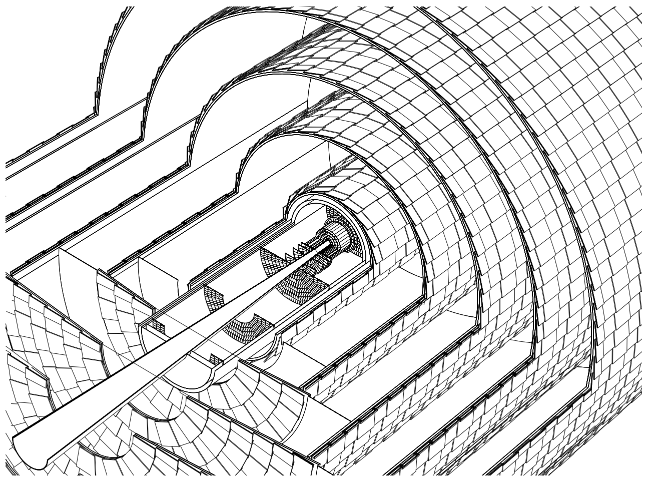
Fig. 13. A cutaway view of the Silicon Detector inner tracker as implemented in Geant4, showing the level of complexity and detail obtainable using the compact.xml file description.

V. Monte Carlo Data Challenges

The set of applications described here has been successfully used to design the Silicon Detector [8], one of the two concepts currently being investigated to study the physics of high energy electron-positron collisions at the International Linear Collider (ILC) [9]. The optimization process for the tracker design started out with a number of simplified geometries, with the complexity of the simulations increasing as the designs became more mature. The current model is quite sophisticated, including most of the details of the engineering models for the support and assembly of the detectors, as well as the electronic readouts currently being considered. Figure 10 shows a cross section of a CAD model of the central tracker. The corresponding Geant4 model is shown in Figure 11. A cutaway view of the full Silicon Detector, as implemented in Geant4 is shown in Figure 12, with a blowup of the central tracking region shown in Figure 13.

The Letter of Intent process required a number of physics analyses to be conducted with full-detector simulation, *ab initio* event reconstruction, and analysis. The physics benchmark processes were deliberately chosen to highlight the intrinsic detector performance, to facilitate comparisons between the concept designs. Although still far from real, the physics benchmarking requirements presented the community with a large-scale, end-to-end exercise which stressed most aspects of the software systems, including:

- Event Generation
- Detector Simulation
- Event Reconstruction
- Physics Analysis

On the order of a hundred million events were fully simulated and reconstructed as part of the data challenge. Extensive use was made of the LCG (primarily DESY, RAL Tier 1 and IN2P3) and OSG (primarily on the FermiGrid) grids. In general, no problems were encountered with the concept software. All the classes were bundled into a single jar file, and the JVM was shipped to grid nodes if needed.

VI. Interoperability and User Base

Although the org.lcsim software suite is fairly fully featured, there exist tools written in C++ which offer additional functionality, or which could be used to cross-check results. The common event model and persistency format allow tools developed in other regions, other languages or other analysis frameworks to be used to process events. LCIO files can be analyzed using root by importing a root LCIO dictionary. The org.lcsim package was developed by and for the ILC physics and detector community, but is being adopted by a wider community. Recently it has been adopted by CERN as one of two frameworks for detector simulations for the CLIC physics CDR [10]. It has also been adopted for the Fermilab-based Muon Collider physics and detector studies [11]. The Heavy Photon Search (HPS) [12] experiment at the Thomas Jefferson National Accelerator facility also used this package for their detector response simulations and is using the software for their data reconstruction and analysis. The Fermilab dual-readout crystal calorimetry group has contributed a number of improvements to the handling of optical processes as part of their detector studies, and a group at SLAC National Accelerator Laboratory has used org.lcsim for ATLAS pixel upgrade simulations.

VII. Summary

The ALCPG physics and detector software group supports an ambitious physics and detector response simulation, reconstruction and analysis effort. The goal is flexibility and interoperability which is neither technology nor concept limited. It provides a complete and flexible detector simulation package capable of simulating arbitrarily complex detectors with runtime detector description. The reconstruction and analysis framework exists, core functionality is available, an individual particle reconstruction has been developed, and various analysis algorithms have been implemented. It has been used extensively in support of studies demonstrating the scientific merit and feasibility of detectors at the International Linear Collider, specifically the successful validation of the Silicon Detector (SiD) Concept. In addition, it is being adopted by an increasingly larger group of users and being adapted to a broader range of applications.

References

[1] S. Agostinelli et al., Nucl. Instr. and Meth. A, 506 (2003) 250
J. Allison et al., IEEE Trans. On Nucl. Sci., 53 (2006) 270
http://geant4.cern.ch/

[2] http://physics.uoregon.edu/ lc/alcpg/

[3] http://www.lcsim.org/software/slic

[4] R. Chytracek, J. McCormick, W. Pokorski, G. Santin, IEEE Trans. On Nucl. Sci., 53 (2006) 2892
http://gdml.web.cern.ch/GDML

[5] http://lcio.desy.de
and N. A. Graf, et al. these proceedings.

[6] http://lcsim.org

[7] http://www.lcsim.org/software/lcdd

[8] http://silicondetector.org

[9] http://www.linearcollider.org

[10] http://clic-study.org/

[11] http://www.fnal.gov/pub/muon_collider/

[12] https://confluence.slac.stanford.edu/display/hpsg/Heavy+Photon+Search+Experiment