Report on GlueX Software Workshop Oct. 21-22, 2005

D. Lawrence
Jefferson Lab

October 28, 2005

Abstract

On October 21st and 22nd,2005 a workshop focused on reconstruction software for GlueX was held at Jefferson Lab. Approximately 15-20 people attended in person, while another 3-4 atrended (at least partially) electronically. This report summarizes the discussions in an attempt to document some of the more important ideas to come out of the workshop.

1 Workshop Announcement

The following is the test circulated to announce the workshop and describe its goals:

> At the spring 2005 GlueX collaboration meeting it was suggested that a software workshop be organized with a focus on reconstruction software for the GlueX detector. In answer to that, a workshop has been planned for Oct. 21-22 to be held at Jefferson Lab.
>
> The main goal of the workshop will be to bring together developers of the reconstruction packages for each of subsystems which make up the GlueX detector. Each will present a plan for how they have (or intend to) address the reconstruction in 3 specific areas as outlined in the charge. Namely: data factories, calibration, and monitoring.
>
> Considerable discussion is expected leading to refinements of the individual packages and a coherency of the overall reconstruction software.
>
> All are encouraged to attend with a particularly warm invitation being extended to GlueX collaborators not usually involved in the software group meetings.
>
> Persons without a JLab ID Badge and those traveling from abroad should contact the organizer (davidljlab.org) to make sure the neccessary arrangements have been made prior to your arrival.

2 Charge of the Workshop

The following is the charge of the workshop circulated when the workshop was scheduled. This applies mainly to detector systems and other reconstruction packages that will deliver data using factories in the DANA framework:

> The GlueX experiment currently under development requires reconstruction software which must meet a large number of requirements in order to satisfy the needs of the experiment. The reconstruction software alone will be of sufficient size and complexity that it will require contributions from a large group of individuals. An even larger group will use the software to perform physics analyses on the GlueX data. To facilitate both the development and use of the reconstruction software in an efficient manner, a development strategy is needed. Developing a clear plan will require input from both developers and users. To meet this need, a workshop will be organized and the participants charged with the following:
>
> 1.) Present a list (for each subsystem) of the data factories which will be used for reconstruction. Their inter-dependancies should be defined as well as the subset that will be accessed from outside of the subsystem package. Naming schemes and general approach will be modified where needed to provide a consistent model across all subsystems.
>
> 2.) For each subsystem, define the list of calibration constants and any other parameters which will be required for reconstruction. Present a plan for how the calibration constants for each subsystem will be derived and what the dependancies are (raw data, other subsystems, previous calibration, etc ...).
>
> 3.) Define what is required to monitor operation of each subsystem. Specifically, what histograms should be produced online to ensure the quality of the incoming data for each subsystem independently and for the GlueX detector as a whole. This will form the basis of the test suite which will be automated to run nightly to ensure the quality of the reconstruction software in the repository while preparing for the experiment.

3 Factory-based Reconstruction Software

Plans were presented for each of the major detector subsystems (with the exception of the UPV subsystem for which the presenter was prevented from attending due to a last minute situation). In addition to the individual detector systems plans for the Tracking and Particle ID packages were also presented. Figures 1 to 8 are the flowcharts that were presented. It should be noted that additional text with descriptions of the factories were also presented in many cases, but are not included here. Please see the document database on the GlueX portal for more info.

4 Discussion Notes

Here are some of the more significant discussion topics from the workshop. The notes are my best recollection of the important points of the discussions. Obviously, these are also colored by my own perspective.

4.1 Monte Carlo and Real Data Merging in Factories

The majority of presentations that included factory flowcharts showed a similar pattern. Namely they looked a lot like the data flow diagram from the design reports (e.g. figure 10.1 from design report version 4). This is not too suprising since M.C. and real data are a little different in their natural forms but at some point must be forced to conform to a common form in order to utilize the same reconstruction software.

Some discussion was had as to the preferred way to converge the two legs (M.C. and real data) in the factory framework. The two options discussed are best described by the TOF and FCAL flowcharts (figures 4 and 7).

In the FCAL scheme, the FCALHit and FCALHit:MC factories produce objects of the same class. The FCALCluster factory then needs only an *if* statement on the factory tag. The FCALHit:MC factory has the responsibility of morphing the FCALMCResponse data into the form of FCALHit objects while the FCALHit factory morphs the FCALRaw into FCALHit objects.

In the TOF scheme, the DTOFHit factory is reponsible for morphing both the DTOFM-CResponse and DTOFRaw data into DTOFHit objects. This scheme has one less factory, but combines two (albeit similar and related) jobs into a single factory.

Both methods would work just fine without either giving a clear advantage in either ease of implementation or increased functionality. There was agreement though that it would be nice to settle on a single scheme for consistency and the FCAL scheme was chosen for essentially aestheic reasons.

Another part of the discussion was on what objects should include M.C. truth information. In retrospect of the discussion here, I don't think there were as many differences in philosophy as I thought at the time. There was general agreement that the reconstruction code should be written to be agnostic as to whether it is real or simulated data. In other words, factories that do the real work of the reconstruction, should not be based on objects that contain any truth information. Factories in the M.C. leg will neccessarily have some truth information, but access to it must be done in a way that is not visible to the reconstruction code. For example, when doing efficiency studies with simulated data, one should use a "matching" factory to correlate the truth data with the reconstructed data. In many cases, parameter matching is sufficient to develop the reconstruction algorithms. Other times, hit-matching must be used in which case, special steps must be taken to carry the truth information through the reconstruction code in a way that it is "hidden" from the reconstruction factories themselves.

4.2 Databasing (Calibration and Otherwise)

There was quite a bit of discussion here. Databasing and remote access technologies have developed rapidly over the past several years and continue to do so. There are many related issues ranging from database table design, to access methods and user authentication. Some features that were generally agreed on were:

- Full historic record. One must be able to ask the database for the set of values that it would have returned had we asked for them at a specific date/time.
- Calibration sets are identified by a context. This can include an inheritance mechanism. For example: A calibration can be specified as "everything that calibration X is, except with these changes".
- Changesets. The ability to specify groups of calibration constants as a set that can be applied or retracted as a whole.
- Private/public constants. Allow people to put calibrations into a private area of the database that will not affect others. This should also include a simple way to"promote" a private calibration (changeset) to "public".

It was decided that a database task force should be formed to explore the issues further and develop an initial design. The persons named were: Nikolay Kolev, Greg Riccardi, Mark Ito, Richard Jones, and David Lawrence.

4.3 Geometry

The first part of the discussion involved HDDS and how best to describe the detector geometry. In particular, how to ease the transition into Geant4. At the same time, the single-source geometry description system in use by the Geant3 simulation needs to be maintained.

It seems the global efforts to produce an XML specification for geometry descriptions a'la Geant4 (GDML) are not developing in a direction that may be the most useful for us. At the very least though, an HDDS to GDML convertor could be written without too much effort. More R & D is needed to see if it is practical, or if the GDML to Geant4-C++ it will produce wil be extremely inefficient by "unrolling" repeating structures.

The other side of geometry information is how to access it from the DANA framework. A plan was presented to do this from C++ using the libxml2 parsing library, already available on most Linux/Unix platforms. Some discussion on the API was had, however. This centered on whether it is best to present the user with a complete set of methods to access the data, or a few generic methods that allow the user to "query" the XML. It's not clear if any performance considerations come into play here. More R & D on this will be needed, likely including some benchmark testing.

4.4 I/O and Introspection

Elliott Wolin showed some of the work currently being done by the JLab DAQ group on upgrading the EVIO tools for accessing CODA formatted data files. The data acquisition system for GlueX will produce data in CODA format so we will need software tools to at least read it into DANA. It may also be desireable to write out factory produced objects into a file, also in CODA format.

This sparked some discussion about what information to keep in the file. Support for defining a dictionary bank in CODA had been discussed in the DAQ group to provide a means of describing clomplex structures in the file. The idea of including some data format information in the file was strongly endorsed in the discussion.

One would like the files to have some information about the format of the data they contain. This is because analyses deal with complex data types that are continually evolving and expanding. The current HDDM does this by placing the complex structure format information in the header of the file as XML. This allows programs to check if they have been compiled with the same structure format as the data files. It also allows one to extract the structure information from the file in order to make a program capable of reading it.

One philosophical issue that came up in this discussion was the difference between having a *data-centric* vs. an *object-centric* view. In a data-centric view, one designs the complex structures around how the different pieces of data are related to one another (HDDM takes this view). In an object-centric view, one designs the objects that are used to hold and manipulate the data in an object-oriented programming model. The complex structures used to store the data in files is then a representation of the objects. In other words, do you start by defining your data structures and then design your objects to conform to them? Or, do you design your objects and derive a data format from those?

The storage and reconstitution problem is a little easier when taking a data-centric view. At the same time, the lion's share of the analysis work will be done in a C++ object-oriented environment which makes an object-centric view desirable.

This discussion brought out several points that will need to be considered while further R & D is performed. It's clear that there is still a bit of a gap to fill in order to conform the current HDDM structures to an object-oriented model. A few people are actively working on how best to fill that gap.

4.5 Milestones and 3-year R & D plan

Keeping in line with the recent efforts to develop a 3 year R & D plan for GlueX, we discussed briefly coming up with a set of software milestones. While the milestones themselves were not discussed, a few areas were identified where strong development is expected to occur over the next three years. The lists these:

- Calibration/Parameters Database
- Geant4 Simulation
- Analysis Framework
- Reconstruction Software

5 Summary

Overall, the consensus was that the workshop proved very productive. Even though there were many of the same people who attend the weekly GlueX Software meetings (many electronically), having everyone gathered in the same room for 1.5 days provided a much better environment for seeing the big picture and defining a focus for solftware development over the next several months.

We also decided it would be good to try and hold workshops periodically. The rule of thumb would be 3-4 weeks before a collaboration meeting so that people can focus

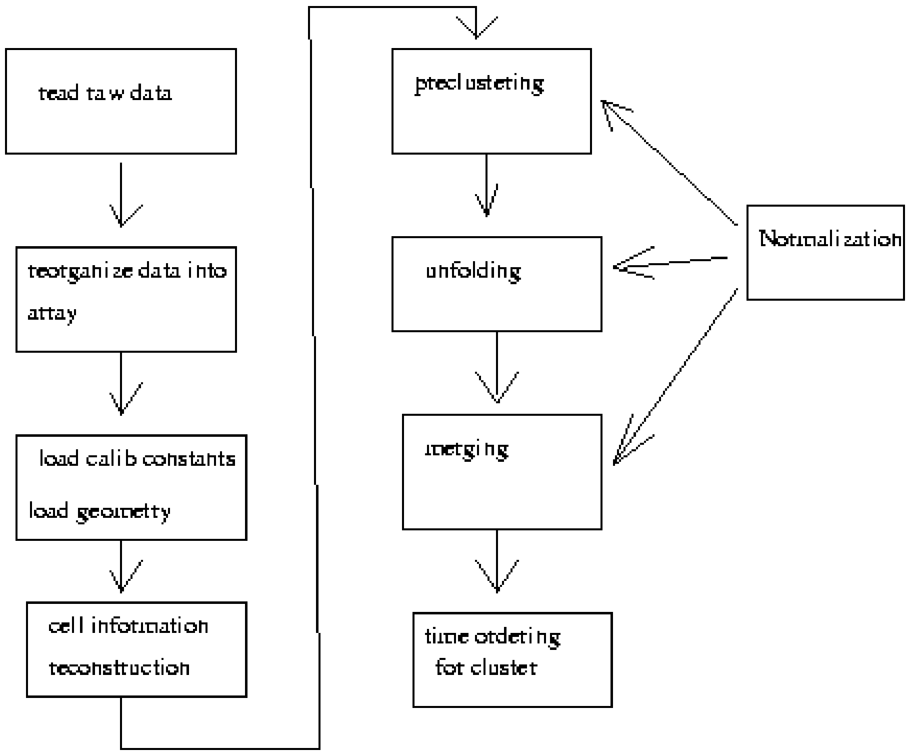

Figure 1: BCAL flowchart (adapted fom KLOE's)

solely on the software and a better assessment of the software status can be given at the collaboration meeting.

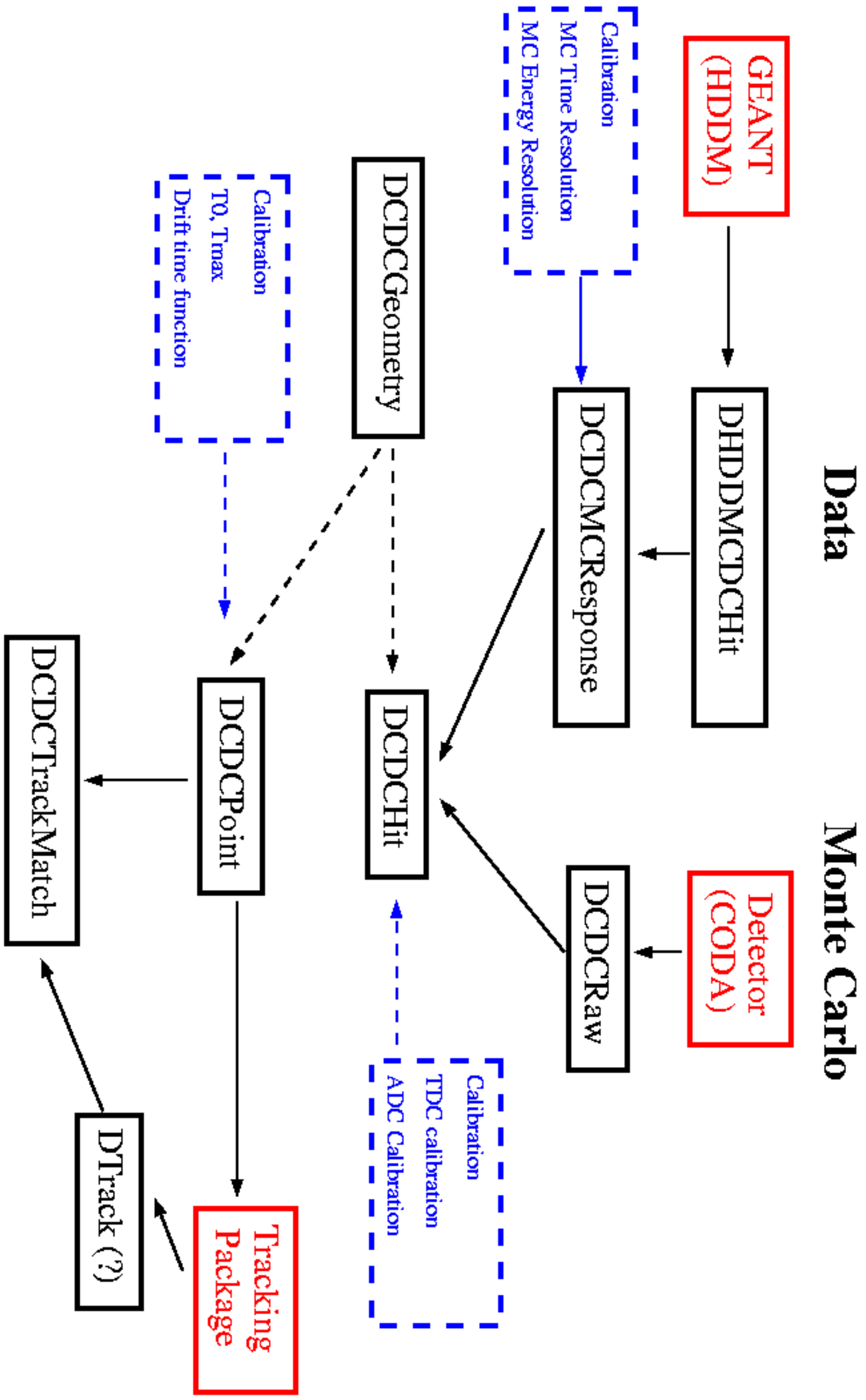

Figure 2: CDC flowchart

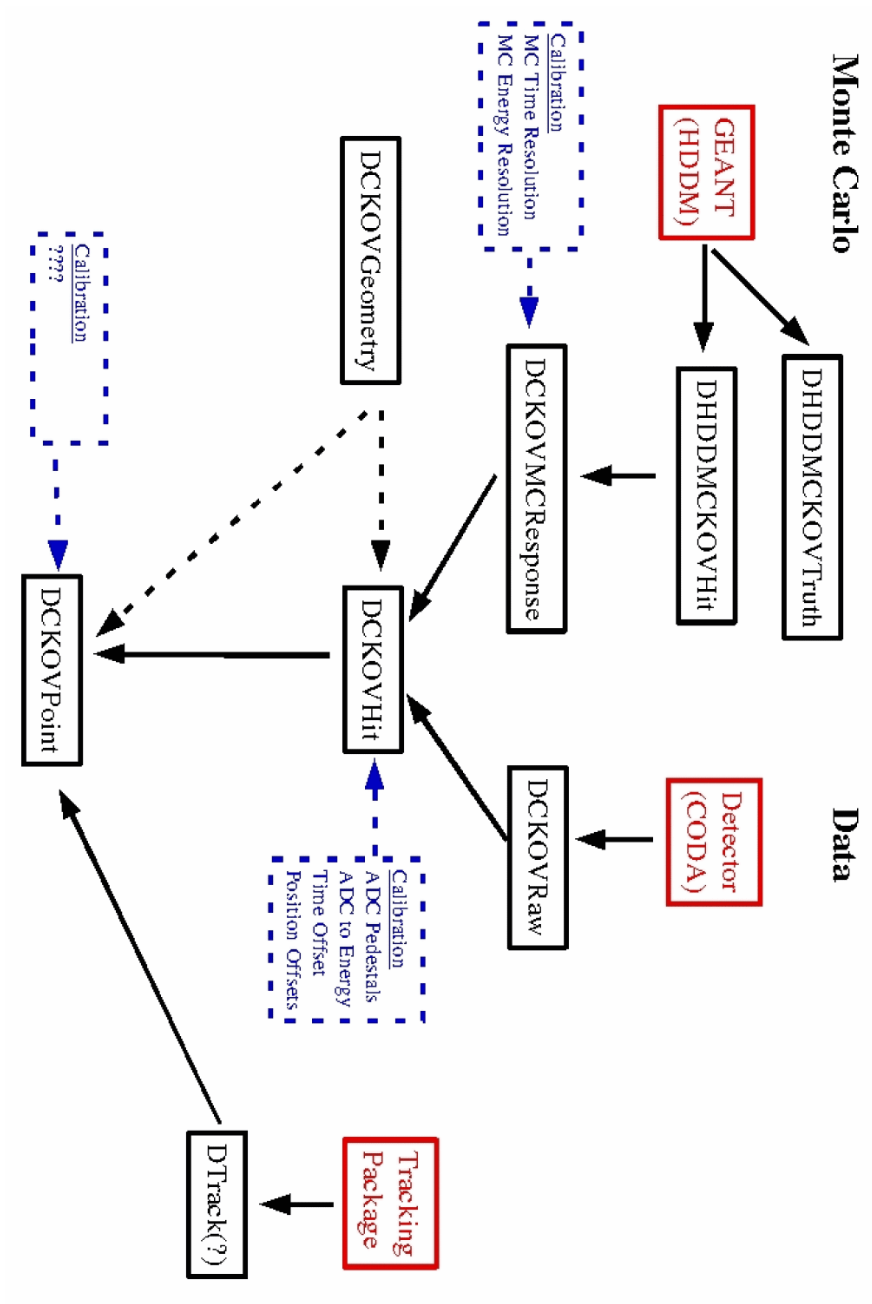

Figure 3: Cherenkov flowchart

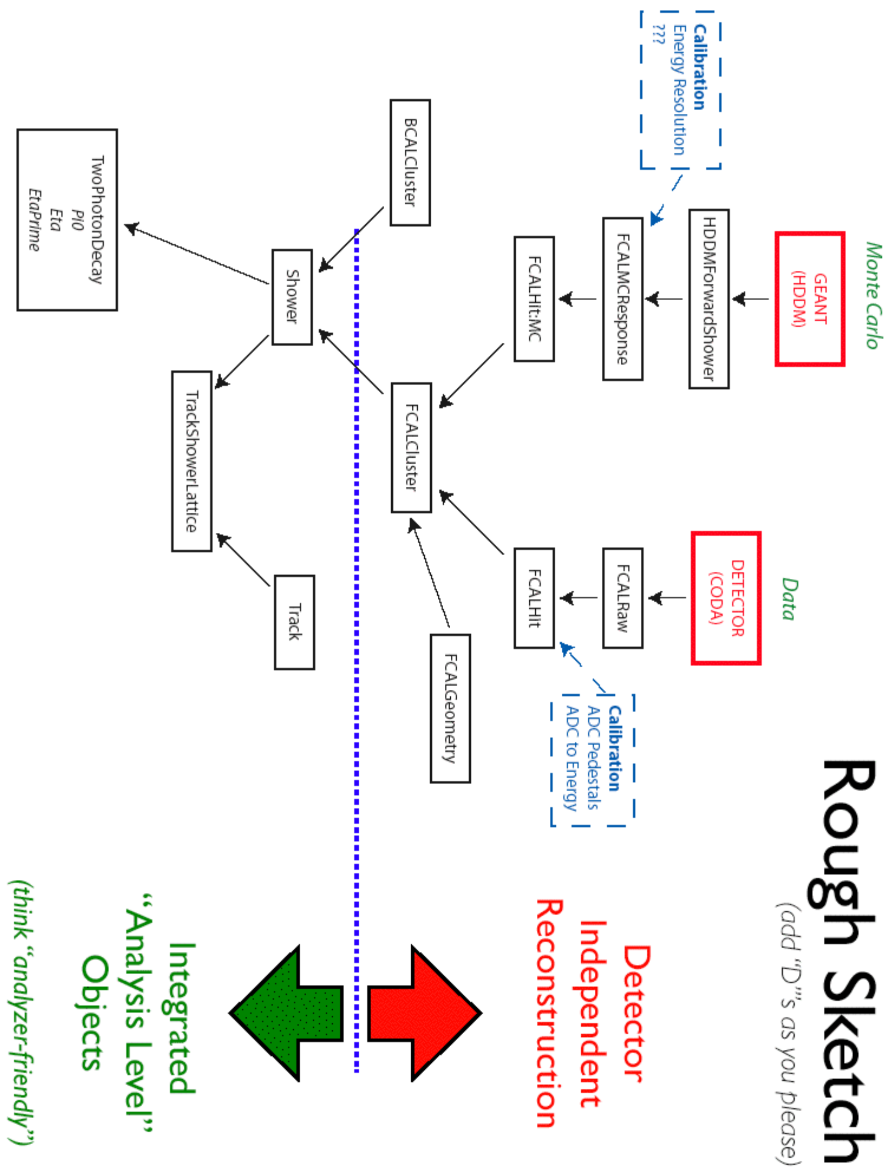

Figure 4: FCAL flowchart

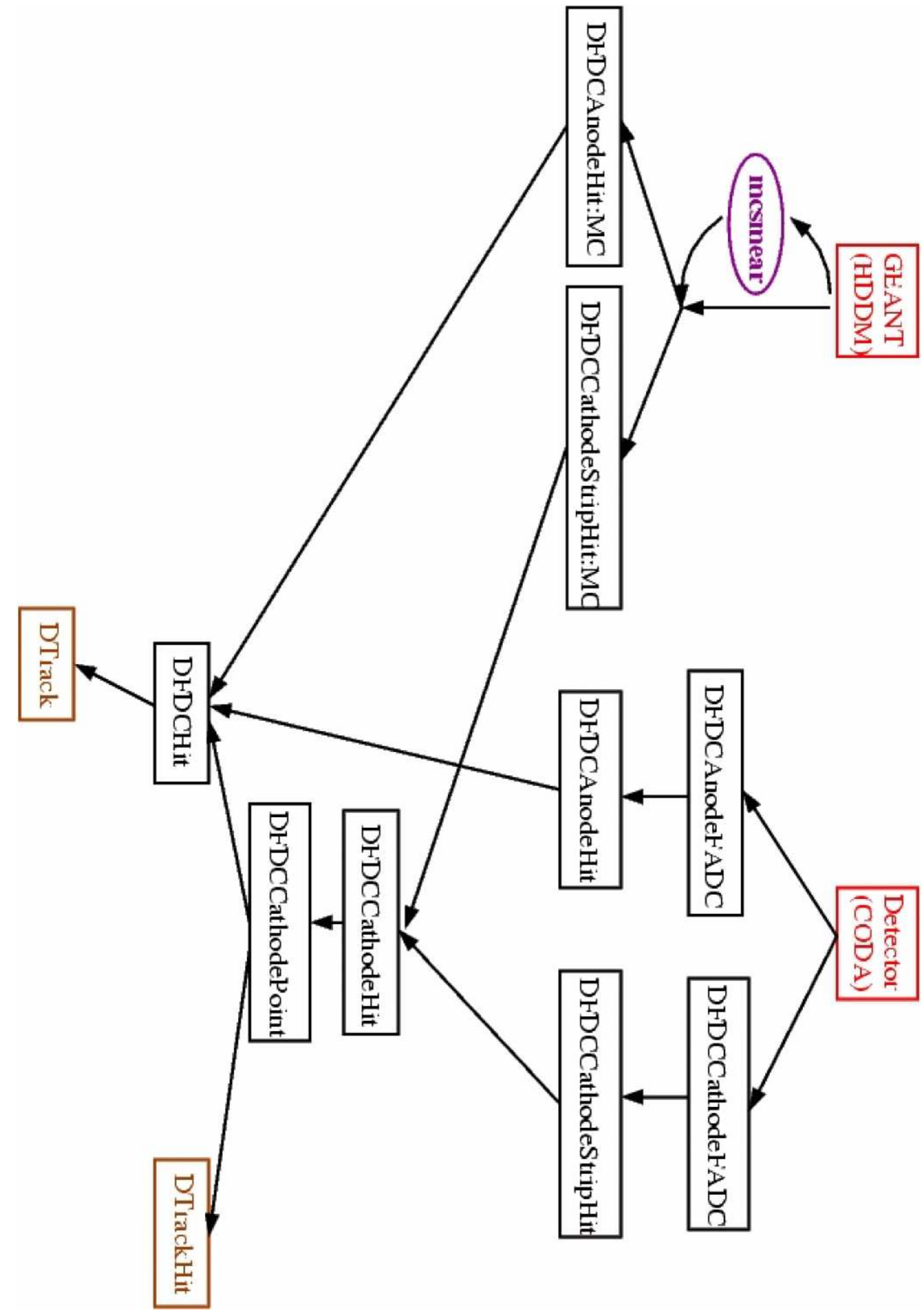

Figure 5: FDC flowchart

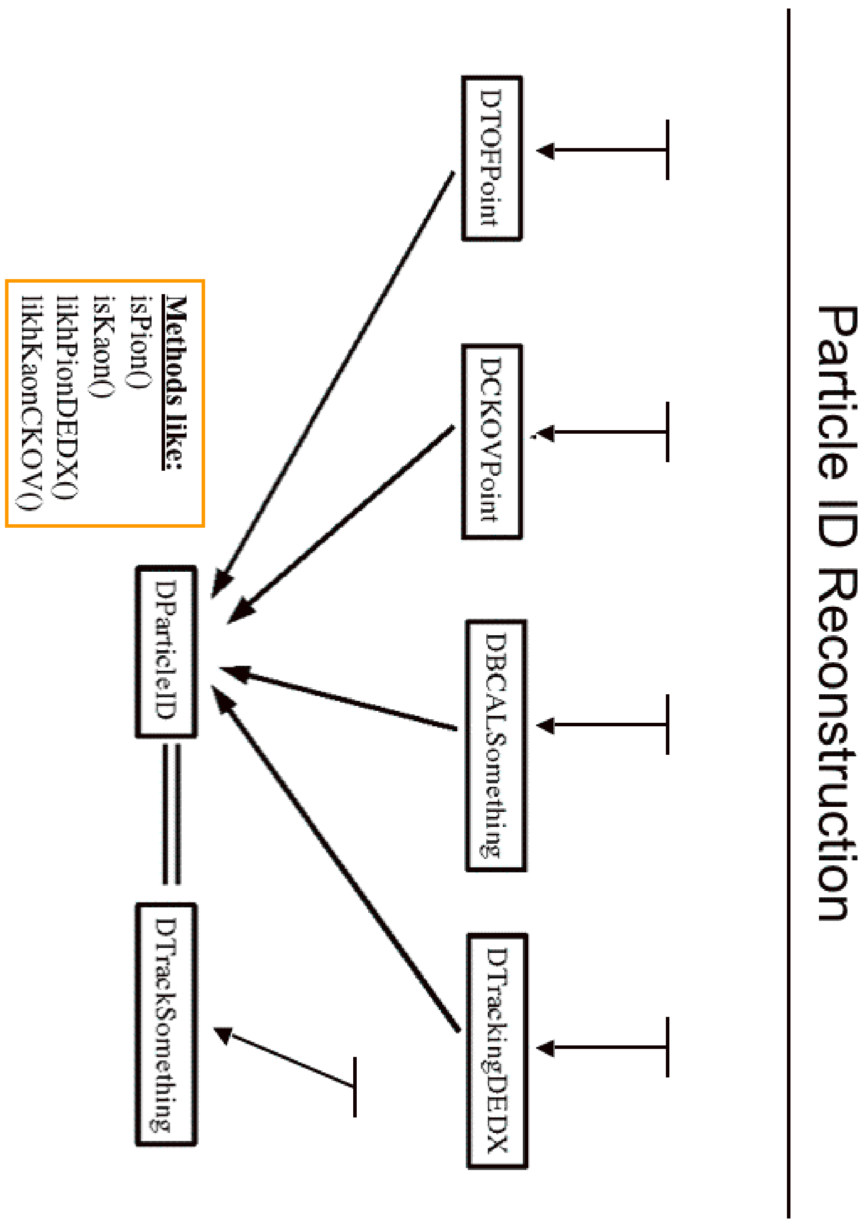

Figure 6: PID flowchart

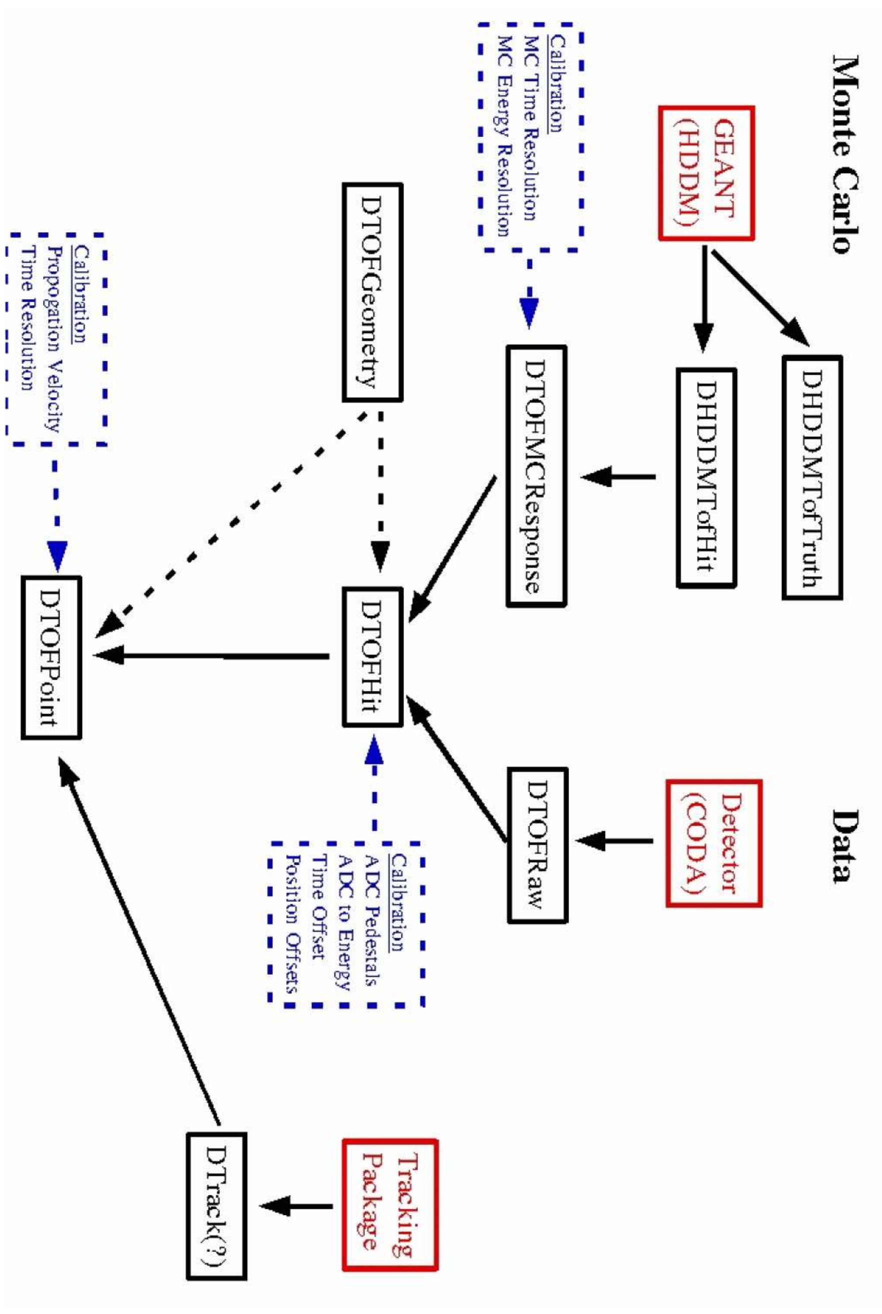

Figure 7: TOF flowchart

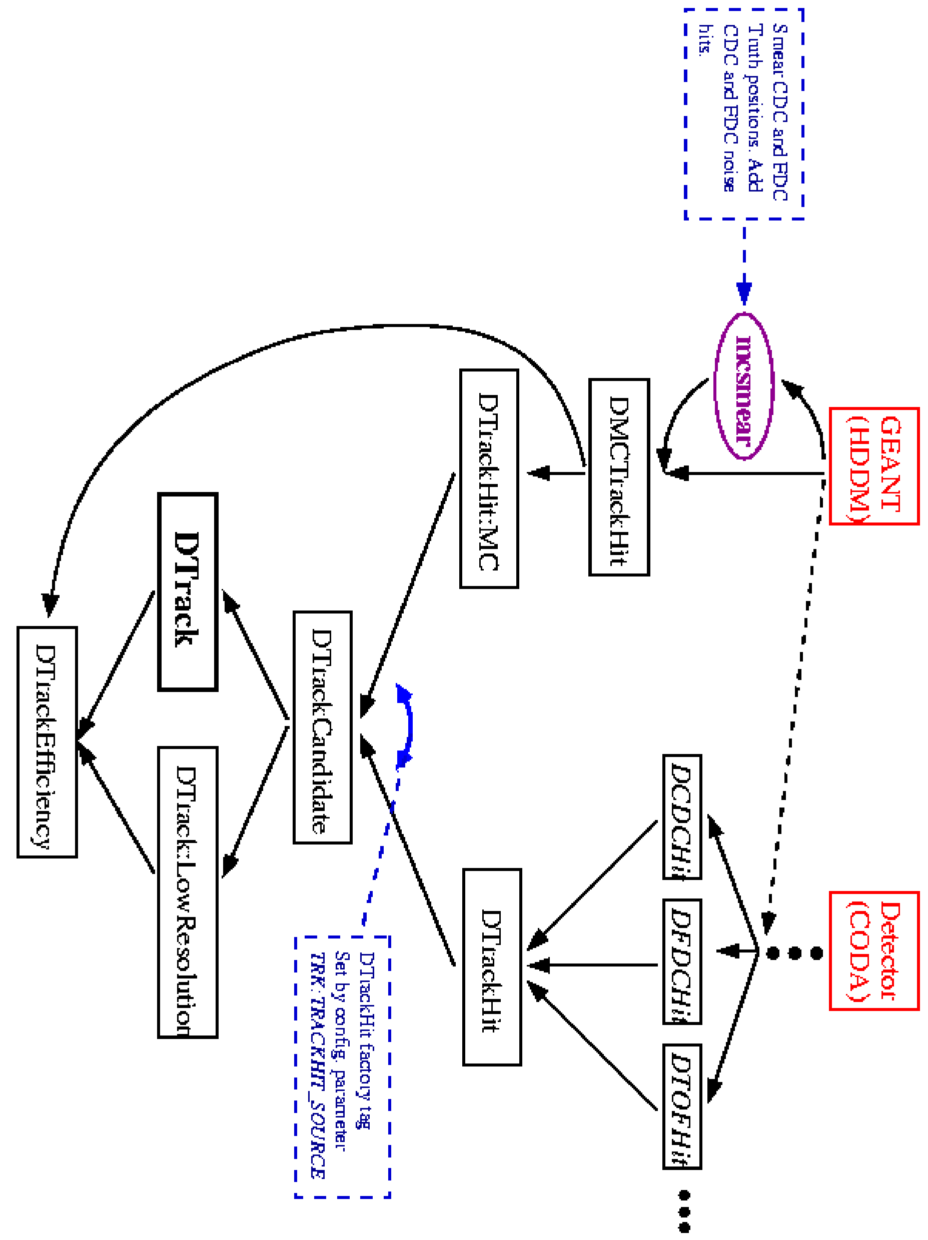

Figure 8: Tracking flowchart