DETECTOR SIMULATION WITH MOKKA/GEANT4: PRESENT AND FUTURE[a]

P. MORA DE FREITAS, H. VIDEAU
Laboratoire Leprince-Ringuet C.N.R.S.-I.N.2.P.3./Ecole Polytechnique
91128 Palaiseau, France

The detector performances are crucial for physics at the Future Linear Collider (FLC). To evaluate the detector designs a full event simulation is required, allowing a complete reconstruction and energy flow analysis. We summarise and report the status of MOKKA, a fully generic simulation tool developed on the Geant4 toolkit for all the current FLC projects.

1 Introduction

The study of a detector for the Future Linear Collider (FLC) requires a complete software framework including event generators, simulations, reconstructions and analysis. This talk describes MOKKA [1], a Geant4 [2] full simulation for the FLC detector studies developed since December 1999, extensively used for the TESLA Technical Design Report (TDR) [3] calorimeter energy flow studies and now able to simulate several detector and prototype models thanks to its way to handle the geometry through a database.

2 Detector Geometry database

Simulation is a CPU time bounded job which simulates the passage of particles through matter, where the detector geometry has to be described at a rather detailed level. The FLC detector project still evolves, in particular different proposals have to be considered with the same physics and the same computer context. Moreover simulation, reconstruction and analysis are independent applications which have to share a common detector geometry at run time to insure their coherence. For this reason MOKKA relies on a Common Geometry Access (CGA) interface which makes available the simulated detector geometry for reconstruction, analysis and visualisation jobs.

The CGA data driven architecture is implemented via a relational database describing the geometry of detector modules (trackers, calorimeters) and complete detector models expressed as sets of detector modules. The database itself is currently implemented in a MySQL [4] server where each detector module is the relationship between a specific MySQL database keeping its geometry

[a] LC-TOOL-2003-010

parameters and a geometry driver name. The geometry driver is a piece of code plugged into CGA which is able at run time to decode the given database and to realise the detector module it describes. This approach allows to create several module versions as relationships between a same geometry driver and different geometry datasets, avoiding additional code development when just the dimensions change. Several detector models are in this way available from the central MOKKA database [5].

This generic approach allows to simulate also prototypes. Actually both electromagnetic and hadronic calorimeter prototypes are implemented. Testing the simulation results against the real data will define up to what level we can trust in MOKKA, the Geant4 framework and the physics list shared by all models.

3 Detector models improvements

The main detector improvement in the most recent release concerns the hadronic calorimeter (HCal) for the TESLA project using Resistive Plate Chambers (RPC). The TESLA Hcal module has 40 layers, each layer with plates of 18 mm of Fe and 6.5 mm of sensitive material. For the first TDR studies the sensitive material considered was scintillator. In the new Hcal release the 6.5 mm is filled with a RPC built as 1.2 mm of gas as sensitive material between 1 mm pyrex walls. The rest of the volume is filled with G10.

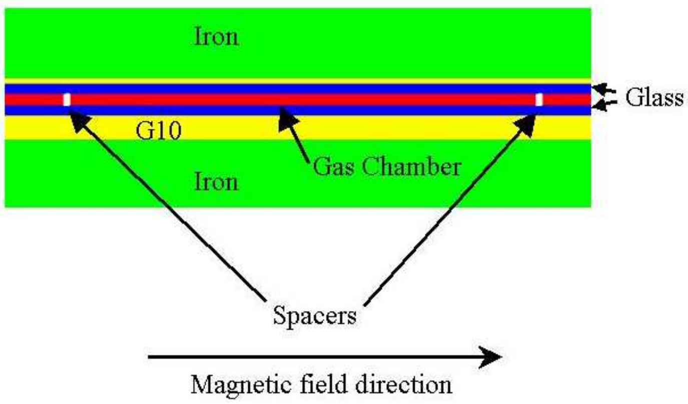

Figure 1: A RPC schema in the new Hcal for TESLA.

Detector performance studies with this new Hcal model have shown dramatic advantages for a gas digital solution, mainly for the event reconstruction. The figure 2 below shows a 20 GeV pion crossing the Hcal module for both models, illustrating the shower shape difference and the potential impact for the particle reconstruction.

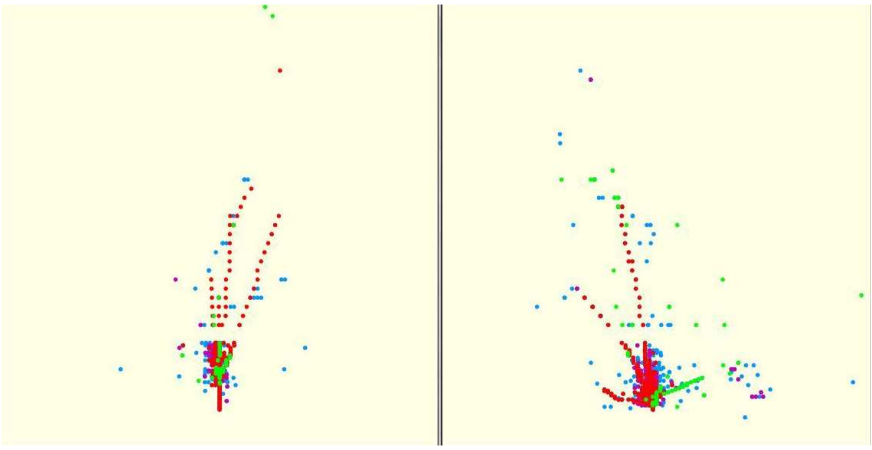

Figure 2: A 20 GeV pion crossing the Hcal module, left using RPCs, right scintillators.

These results have to be confirmed in beam tests, but it shows the power of simulating different detector versions in the same computer environment.

Implementing new detector modules and models should be a "user job": MOKKA is open for a wide collaborative use but also development. Sources and documentation are available from the MOKKA Web page. The "geometry drivers plug-in scheme" is quite simple, providing several facilities to help developers to deal with the Geant4 interface. Currently some collaborators around the world have started developing detector modules for MOKKA in this context.

4 MOKKA framework status

The last MOKKA release introduces a simple event display for debugging. It relies on the Geant4 visualisation features and is able to reload from files simulated events to be displayed. Future developments should implement selection commands to choose tracks, hits and detector modules for display with run time selected colours, a friendly graphic interface for the Geant4 visualisation facility, etc.

Concerning the data output issues, MOKKA is compliant with a persistency abstraction schema. The application doesn't rely on any particular persistency tool (Tree, Objectivity, etc), but such layers can be connected to the MOKKA kernel via an abstract interface. The actual persistency environment should be a user choice at run time. MOKKA provides as native a very simple persistency strategy keeping the output data as ASCII files where the format is completely free, depending only on the detector driver. Runs and events are managed just using standard UNIX tools like directories (for runs) and file names (for detector modules and event number identification). Complete run data is written on tapes using the "tar" standard tool.

Future developments shall implement the Linear Collider Serial Input Output (LCIO) [6] persistency scheme as option.

CGA provides application interfaces (API) for F77, C++ and C for programmers developing reconstruction and analysis code, a Java API is foreseen. It implements also a few reconstruction utilities, for example to calculate the interaction length between two given space points, which should be extended in the future.

The figure 3 below illustrates the current MOKKA software architecture and its dependencies, mainly on the Geant4 framework, with all the extensions described in this paper.

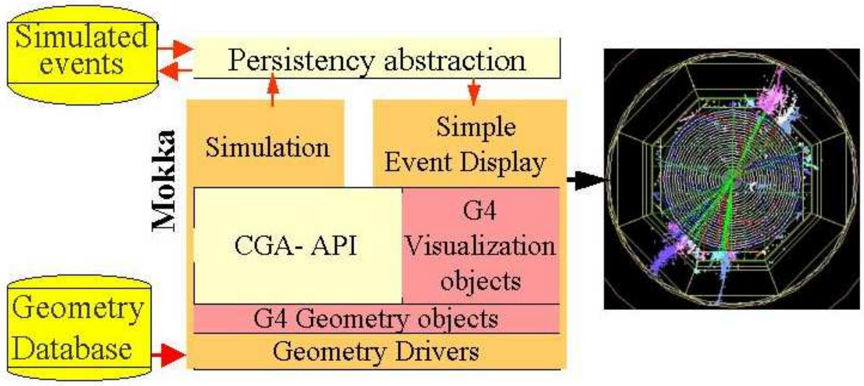

Figure 3: MOKKA general software architecture.

5 Conclusions

The recent evolutions of MOKKA move this simulation tool closer to a general framework providing additional features for reconstruction, analysis and visualisation for the Next Linear Collider detector studies. Thanks to the CGA database approach this framework provides a good platform to compare different detector proposals with the same physics and computer context. Crucial results are already taken in account for the TESLA project concerning the Hcal models.

Thanks to the several CGA APIs and the reconstruction utilities it implements, the CGA becomes a common layer to develop reconstruction and analysis code keeping the detector geometry coherence along these several tasks.

Future developments should extend the MOKKA framework functionalities, mainly concerning its event display and data persistency sub-systems. New detector and prototype models should be implemented by end users thanks to an open wide collaborative environment.

References

1. MOKKA Home Page, http://polywww.in2p3.fr/geant4/TESLA/www/MOKKA/MOKKA.html. See also http://www-lc.fnal.gov/proceedings/proceedings.html#simulation/Freitas.ps for the previous LCW2000 MOKKA report.
2. Geant4 Home Page, http://wwwinfo.cern.ch/asd/geant4/geant4.html.
3. TESLA, The Superconducting Electron-Positron Linear Collider with an Integrated X-Ray Laser Laboratory, Technical Design Report (TDR), http://TESLA.desy.de/new_pages/TDR_CD/start.html.
4. http://www.mysql.com/
5. http://polywww.in2p3.fr/geant4/TESLA/www/MOKKA/software/database/ phpMyAdmin/index.php3
6. LCIO, a persistency framework for future linear collider detector simulations, http://www-it.desy.de/physics/projects/simsoft/lcio/