CbmRoot: Simulation and Analysis framework for CBM Experiment

M. Al-Turany, D. Bertini and I. Koenig
GSI Darmstadt, Germany

Abstract

The Compressed Baryonic Matter (CBM) is an experiment at the future FAIR (Facility for Antiproton and Ion Research) in Darmstadt. The goal of the experiment is to explore the phase diagram of strongly interacting matter in high-energy nucleus-nucleus collisions. For simulation the Virtual Monte Carlo concept was choosed [1], this concept allows performing simulations using Geant3, Geant4 or Fluka without changing the user code or geometry description. The same framework is then used for the data analysis. An Oracle database with a build-in versioning management is used to efficiently store the detector geometry, materials and parameters.

DESIGN AND IMPLEMENTATION

The schematic design of the CBM framework (CbmRoot) is shown in Fig.[1].

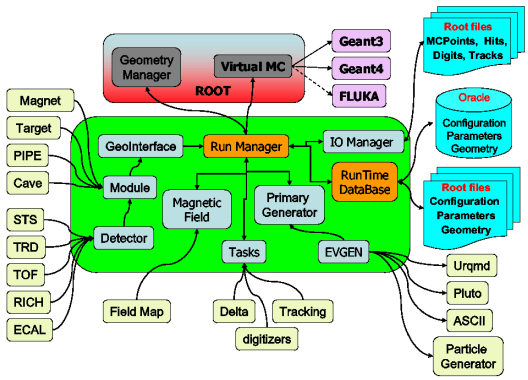

Figure 1: Schematic design of the CBM simulation and analysis framework

In this design, the framework is fully based on the ROOT system [2].The user can create simulated data and/or perform analysis with the same framework. Moreover, Geant3, Geant4 and Fluka transport engines are supported, however the user code that creates simulated data do not depend on a particular monte carlo engine.

BASIC FUNCTIONALITIES

The framework delivers base classes which enable the users to construct their detectors and /or analysis tasks in a simple way, it also delivers some general functionality like track visualization (see Fig.[2]). Moreover an interface for reading magnetic field maps is also implemented. in the following a general description of the software is presented, for further technical details please see the home page of the CBM collaboration [3].

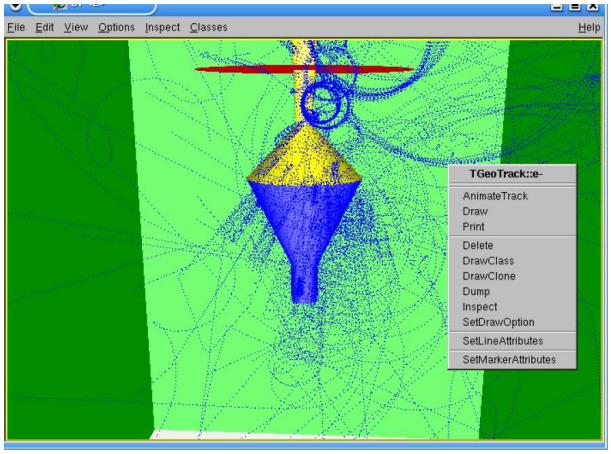

Figure 2: Track visualization

Input/Output procedures

The storage of all information collected by the different sensitive detectors is done on an event by event basis (an event means in this context one interaction between one beam particle and the target) . All persistent objects are serialized and stored into binary ROOT files. An interface class (CbmMCPoint) is provided to define the structure of registered hit in a detector. Each detector can then provide a more specific implementation following the CbmMCPoint API. All registered hits will be collected into dedicated lists, one list corresponding to one detector entity. The ROOT class TTree is used to organize the output data into a ”ntuple like” data structure . In the analysis case, the CbmRootManager provides methods to read this information. A partial input/ouput mechanism is also supported.

Parameters definition

In order to analyze the simulated data, several numerical parameters are needed, as for example, calibration/digitization parameters or geometry positions of detectors. One common characteristic to most of these parameters is that they will go through several different versions

corresponding, for example, to changes in the detectors definition or any other condition. This makes necessary to have a parameter repository with a well-defined versioning system. The runtime database (realized through the CbmRuntimeDb class) is such a repository. Different inputs are supported : Ascii format , ROOT binary format and Oracle Database input. Fig.[3] shows the initialization schema used to connect the different parameters with data.

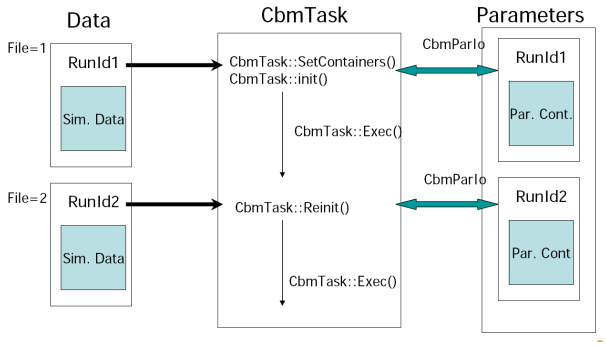

Figure 3: initialization schema

Implementation of the algorithms

The analysis (Reconstruction) is organized in tasks. For each event we need to accomplish various tasks or reconstruction algorithms. The CbmTask is an abstract class defining a generic API allowing to execute one task and to navigate through a list of tasks. The user can create his own algorithm inheriting from CbmTask. Each task defines the relevant input data and parameter and creates its particular output data during the initialization phase. During the execution phase, the relevant input data and parameters are retrieved from the input file and the output data objects are stored in the output file.

SUMMARY

A VMC based framework for CBM has been implemented, the first release was in march 2004. The October 2004 release was used to produce and analyze data for the CBM technical Status report[4]. The Parameter containers and the initialization scheme is now added. Work on digitizers and full tracking is going on.

REFERENCES

[1] R.Brun, F.Carminati, I.Hrivnacova, A.Morsch *The Virtual MonteCarlo* Computing in High Energy and Nuclear Physics, 2003, 24 - 28 March 2003, La Jolla, California.

[2] R. Brun, F. Rademakers, P. Canal, I. Antcheva, D. Buskulic, O. Couet, A. and M. Gheata *ROOT Users Guide* (CERN, Geneva, 2005)

[3] http://www.gsi.de/fair/experiments/CBM

[4] CBM Collaboration *Technical Status Report* (GSI, Darmstadt, 2005)