FROM GEANT 3 TO VIRTUAL MONTE CARLO: APPROACH AND EXPERIENCE

Y. Fisyak, J. Lauret, V. Perevoztchikov, M.Potekhin
Brookhaven National Laboratory, Upton, NY 11973, USA

Abstract

The STAR Collaboration is at present using simulation software based on GEANT 3. The emergence of the new Monte Carlo simulation packages, coupled with the evolution of both STAR detector and its software, requires a drastic change of the simulation framework.

We see the Virtual Monte Carlo (VMC) approach [2] as providing a layer of abstraction that facilitates such transition. It has the potential to replace the present legacy software, and help avoid its certain shortcomings, such as the use of an obsolete algorithmic language to describe the detector geometry. It may also allow us to introduce a more flexible in-memory representation of the geometry. In particular, we consider the geometry description classes in the ROOT system [1] as a good choice for the in-memory geometry model.

We present our first experience in evolving the simulation software from the GEANT 3 based platform, towards the VMC, and directions for future development in this area.

CHARACTERIZING THE TRANSITION

Features of the existing STAR simulation software

The current STAR simulation software provides the following tools for the user/developer:

- Detector (geometry) description language
- Hit structure declaration (part of the above)
- User-prescribed volume enumeration (important for analysis)
- Comprehensive I/O system
- Logic for treatment of secondary particles (and the mechanism of saving “interesting” particles for analysis)
- Persistent documentation system (self-documenting Zebra banks)

We would like to preserve this feature set in any future implementation of the simulation software in STAR.

Motivations for the transition

(a) Software maintenance:

> Maintenance of the currently existing software is not trivial and requires highly specialized expertise. We need, therefore, a hedge against our simulation framework becoming obsolete in terms of user expertise (such as detailed knowledge of the ZEBRA memory management system and many other legacy components).

(b) Software integration:

> There is a need to make the STAR software more robust by implementing a unified geometry description for simulation and reconstruction

(c) Migration to a different simulation platform:

> GEANT 3 is being gradually phased out in many projects. The Virtual Monte Carlo can be seen as a bridge between GEANT 3 and GEANT 4, and possibly other MC platforms.

In addition, we would like to increase the transparency of the simulation code and tools to the users, in part due to benefits of OO programming and the more familiar language platform (C++ vs. Fortran) to be used in the VMC.

In our opinion, the VMC has the potential to become a long term MC solution for STAR, with a stable API, and a geometry description system that facilitates the ongoing detector development and upgrades.

Alternative geometry managers

Apart from geometry managers built into toolkits such as GEANT, there is one included in the ROOT system [1], containing a system of geometry related classes.

The Root geometry system

- is feature-rich and allows one to build complex geometries, suitable for building an application such as VMC
- can also be used in event displays and, in recent developments, was augmented with sophisticated graphic functionality [4]

- is “platform neutral” in the sense that it is not really tied to any particular application such as GEANT
- can provide a consistent geometry interface with reconstruction, and used for geometry navigation in reconstruction
- opens the possibility of having the geometry persistent in ROOT files, XML (future development), and may facilitate the creation of CAD systems interfaces

INFRASTRUCTURE ISSUES

We need to preserve certain features of the simulation software such as structured detector description, hit structure declaration, user-defined volume enumeration, and comprehensive I/O with proper interface to the reconstruction software. We touch upon these topics below.

Detector Description

Detector Description is the method by which the detector geometry model, and the properties of its constituents, is defined by the user. For example, it can be a piece of Fortran or C++ code, a certain CAD system file, a XML file etc.

Detector Description (DD) is an important component of the simulation software infrastructure, for the following reasons:

- a robust DD system helps the ongoing detector development and upgrade process, especially in a running experiment like STAR, where there are few resources immediately available
- having it in place early during the transition process allows one to have the geometry developed by the time of the proposed switchover

In recent years, a few experiments have implemented quite different DD systems. These roughly fall in two categories:

- C++ or some other algorithmic language code with a proper API to interface the database
- XML-based solution

We consider the XML-based detector description being of particular interest. This is based on our experience in STAR experiment, which shows that having a structured, declaration-based description of geometry helps a great deal in the creation and maintenance of the geometry code.

Extending this approach to a new platform, supported by the industry, seems to be a logical step forward. In addition, there is a welcome possibility to use XML as an exchange format.

Based on the survey of a few existing XML-based detector description languages (insofar, none of which has become adopted as an industry standard), the functionality of such XML solution should include

- constants
- arithmetic expressions
- iterators (expressing symmetry in the system)
- arrays of parameters
- export of pure XML for visualization
- database interface

The issues we are facing in considering the XML-based DD include having to make a choice of a particular language (schema), and the fact that none of the existing parsers is useable right away in the ROOT geometry context, meaning this functionality needs to be implemented from scratch. Another problem lies with the GEANT 3 and 4 geometry implementation differences [3], which has implications for the development of the schemas.

Volume navigation and enumeration

In the context of the event analysis, it is often extremely desirable to have a user-defined enumeration of the geometrical volumes. For example, towers of a calorimeter can have a special enumeration map that assists in calibration, histogramming and track matching. The possibility of having such numbering present and propagated with the data, in the geometry model is therefore also a requirement.

Hit declaration

One remarkable feature of the existing STAR simulation software is the ability of the user to declare the 'hit structure' in a formal way, defining which variables and how need to be stored in a particular 'hit', associated with a sensitive volume. Very little code needs to be written by the user, as the necessary code will be automatically generated.

We consider that standard declaration of hits an extremely useful feature, which helps a lot in the detector development and code maintenance, and would like to have similar functionality going forward.

CURRENT STATUS OF THE STAR VMC PROJECT

We have developed a set of core C++ classes for the Virtual Monte Carlo platform in STAR. A test application, allowing us to test-drive these classes, has been written.

It is worth mentioning that the current VMC software (based on ROOT) allows one to select whether a standard

GEANT 3 geometry should be used, or the ROOT one. In this particular case, we opted for using the Root geometry.

We also performed:

- validation of the ROOT geometry (see below)
- Restructuring of the TGeant3 class library [2] as used by the Alice collaboration, in order to factor out specific components and make the system more modular.

The validation of the Root geometry, with regard to the existing GEANT 3 geometry, was done in the following manner: the GEANT based geometry model of the STAR detector was converted into a ROOT-based model using utilities such as g2root [2]. Then, both a ROOT VMC and GEANT applications were run using the same underlying detector model, and were instrumented to collect certain metrics such as distribution of track lengths in individual volumes etc. No significant differences were detected. Combined with validation done by the Alice group [3], this gives us confidence in the integrity of volume navigation in Root, in the simulation application context.

CONCLUSION

We in the STAR experiment are exploring the ways to migrate the simulation software to the Virtual Monte Carlo platform as outlined in [2]. We have created a set of core classes as the first step in developing this new software, and have successfully run a test application that uses a ROOT-based geometry model of the STAR detector.

We have also identified a few critical directions for the development. These include a Detector Description system, which we believe would be best implemented as a XML-based language, as well as some other elements of the infrastructure.

ACKNOWLEDGEMENTS

We would like to thank A .Gheata for the practical help with the VMC software, and R. Brun and P. Nevski for many useful discussions.

REFERENCES

[1] http://root.cern.ch/
[2] http://root.cern.ch/root/vmc/VirtualMC.html
[3] A. Gheata, "The Virtual Monte Carlo, Status and Applications", these proceedings.
[4] V. Fine, "Visualization of the ROOT Geometries", poster presentation at CHEP'04