A method for detector description exchange among ROOT GEANT4 and GEANT3*

YOU Zheng-Yun(尤郑昀)[1,2;1)] LIANG Yu-Tie(梁羽铁)[1] MAO Ya-Jun(冒亚军)[1;2)]

1 (School of Physics, Peking University, Beijing 100871, China)
2 (Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA)

Abstract GEANT3 and GEANT4 are the detector simulation software programs that are widely used in most nuclear and particle physics experiments. ROOT is a program for framework, data analysis, online and offline software. Detector description is an important function in all these programs. Due to different detector construction methods and respective detector data representative, it is difficult to exchange the detector data among them. A new method based on GDML is developed to automatically convert the detector data among ROOT, GEANT4 and GEANT3. Any existing detector geometry in one program can be mapped to the geometry in the other two programs. In the software development of an experiment, different applications can share and reuse the same detector description. The application of this method in the PHENIX experiment upgrade and PHENIX Forward Vertex Silicon detector design is introduced.

Key words detector description, GDML, GEANT3, GEANT4, ROOT

PACS 07.05.Tp, 07.05.Fb

1 Introduction

GEANT3[1], GEANT4[2] and ROOT[3] are software programs that are widely used in nuclear and particle physics experiments all over the world. GEANT is a system of detector description and simulation tool that helps physicists to design and optimize the detectors, develop and test the reconstruction and analysis programs, and interpret the experiment data. The FORTRAN based GEANT3 was developed by CERN from 1982 to 1994, and was used in simulation programs of many experiments like CDF, STAR and Pioneering High Energy Nuclear Interaction eXperiment (PHENIX). The C++ based GEANT4 with more powerful functions is developed since 1999. ROOT is also developed by CERN to meet the software requirement of the next generation data analysis in experiments like the Large Hadron Collider (LHC). With the development of object-oriented technology, the C++ based ROOT is being used for framework, online and offline software in all major high energy and nuclear physics laboratories around the world, to monitor, store and analyze data.

Detector description is one of the most important tasks in simulation tools like GEANT. In Monte Carlo calculations of GEANT, a detector should be described precisely and closely enough to reality, so as to better simulate the response of the detectors to the experimental conditions. The geometrical structure and properties of all the components of detectors must be specified in detail. The specifications are often provided through function calls in the language of the software libraries being used (FORTRAN for GEANT3 and C++ for GEANT4). The detector data exchange between GEANT3 and GEANT4 is difficult since they are based on different computer languages. Usually codes need to be rewritten while upgrading from GEANT3 to GEANT4. Rewriting source code is not only a large amount of work for complex detectors, but also difficult to keep consistent in these two sets of description.

Detector description is also necessary in the reconstruction programs of online and offline software. To reconstruct the original tracks from the digital signal output, positions of the detecting units should be known. Some more questions like “In which volume is a given point?” and “Where is the boundary of a volume?” should be answered. Mathematical calcu-

Received 21 September 2007
∗ Supported by NSFC (10675004, 10375002) and Doctoral Fund of Ministry of Education of China
1) E-mail: youzy@hep.pku.edu.cn
2) E-mail: maoyj@hep.pku.edu.cn

lations to resolve the coordinate transformation and boundary search of the units are usually defined by the code developers. However, for complex detectors with many levels of structure or detector units with irregular shapes, this task becomes difficult. A new ROOT geometry package[4] with geometrical modeling capabilities has been developed in the ROOT framework since its 3.14 version. The architecture of this modeler is a combination between a GEANT-like containment scheme and a normal constructive solid geometry (CSG) binary tree at the level of shapes. The ROOT geometry package helps developers to construct complex detectors and provides many functions to support geometry service in reconstruction software.

In the software development of an experiment, detector description is used in different applications like simulation, reconstruction, visualization, alignment and physics analysis. It is natural that only one set of detector description should be shared and reused for all applications so as to keep them consistent in an experiment. One example is that the reconstruction software processes both the simulation data and the real data. The share of geometry between the simulation and reconstruction software can reduce the systematic error that is brought by two sets of detector description, so it is more reliable to compare the simulation data and the reconstruction data, especially for complex detectors. However, in most experiments, detector is described separately for each application. Because of different construction methods and different computer languages dependency, there is no efficient way to exchange the detector description among these software programs.

2 Detector description exchange with GDML

GDML (Geometry Description Markup Language)[5] is an XML (eXtensible Markup Language)[6] based detector description language, which is developed by CERN and is particularly for GEANT4. All necessary information to describe a detector is provided through a set of “tags” and “attributes” that are parsed and passes through the code that provides the Monte Carlo function calls in the proper formats. The GDML description files are human readable and plaintext which is editable by any ordinary text editor. GDML is platform (Windows or Linux or other operating systems) independent and language (FORTRAN or C++ or other languages) independent; it provides an efficient way to use a common source of information for different programs so long as the corresponding interfaces are developed.

A method based on GDML is developed to manage the detector description data and exchange them among different programs. We develop an interface from GDML to ROOT geometry package and expand the functions of the GDML-GEANT4 interface to support more complex shapes. The GDML files can be read by GEANT4 to construct the detector geometry for simulation, and at the same time, these files can be read by ROOT to construct the detector in ROOT for reconstruction, event display, analysis and other ROOT-based applications. On the other hand, the reverse conversion is also realized. For an existing detector model that is already described with GEANT4 or ROOT detector construction codes, the GDML interfaces provide the functions to export the detector data from GEANT4 or ROOT program to GDML files, so as to reuse them in other programs.

GDML uses the XERCES-C++ parser and C++ code for producing the code syntax. The category tags for parsing include:

(1) ⟨namespace⟩ to set the fields where GDML is used;

(2) ⟨define⟩ to set values like constant, position and rotation;

(3) ⟨materials⟩ to specify all medium and materials used in detector;

(4) ⟨solids⟩ to describe more than 20 kinds of basic shapes and complex Boolean shapes;

(5) ⟨structure⟩ to build the whole detector with tree structure, by connecting a volume with its shape, material, children volumes and mother volume.

Each time a tag is read, the GDML-ROOT interface calls the corresponding definition or construction function in ROOT. When the file is completely read in, the whole detector is constructed in application as ROOT objects at the same time. The GDML-GEANT4 interface works in a similar way. The GDML detector description is program independent; those who prepare the description files do not need to know the functions in GEANT4 or ROOT. So long as the GDML description file is provided, the interface can automatically construct the detector in GEANT4 or ROOT’s respective construction method. Automatic conversion not only reduces by large amount the work of developers, but also guarantees the consistency of detector description in different applications.

For some large nuclear and particle physics experiments that have already run several years, the simulation programs were usually developed with GEANT3. Sometimes there is a requirement to upgrade the simulation program to GEANT4. A method to use the existing GEANT3-based detector description for the GEANT4-based or ROOT-based application is also found. GEANT3 has the functions to export the whole detector information, including structures, shapes, materials and volumes to a zebra-format file, or to import the detector information from a zebra format file. With the function "g2root", which is provided in the new ROOT geometry package, the zebra file that contains the existing GEANT3 detector information can be converted to the ROOT geometry representative. The GDML-ROOT interface has the ability to transform the ROOT representative to the GDML file and then the GDML file can be read through the GDML-GEANT4 interface to construct the detector in GEANT4, in which the detector description keeps exactly the same as that in GEANT3. The upgrading from GEANT3 to GEANT4 is easily achieved without rewriting the detector construction codes in GEANT4.

The chain to exchange the detector description data among GEANT3, GEANT4, ROOT and GDML is shown in Fig. 1. All conversions are automatically realized. The existing geometry in one program can be mapped to representative in other programs and can be reused in different applications.

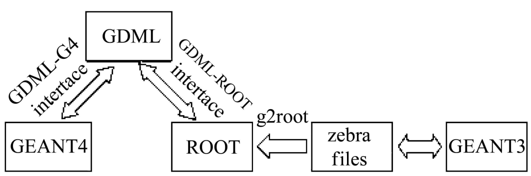

Fig. 1. Chain to exchange the detector description among GEANT3, GEANT4, ROOT and GDML.

3 Applications in PHENIX experiment

We first successfully used the GDML-based detector description in the software of BESIII experiment, for the GEANT4-based simulation and ROOT-based reconstruction, event display and data analysis[7]. Here we introduce how this method is used in PHENIX[8], which is a high energy physics experiment running on RHIC (Relativistic Heavy Ion Collider) at Brookhaven National Laboratory. The physics goals of PHENIX experiment include the study of the attributes of nuclear matter under extreme hot and dense conditions and spin structure of nucleon. PHENIX project began in 1993 and started collecting data in 2001.

3.1 Simulation upgrade from GEANT3 to GEANT4

During the early stage of the PHENIX software development in the 1990s, GEANT3 was the most popular simulation tool and was used for PHENIX simulation program. With GEANT4 fully developed and being used in many particle experiments in recent years, there is an intention to upgrade the PHENIX simulation tool from GEANT3 to GEANT4. The conversion from the GEANT3-based detector description to the GEANT4-based simulation program is first tested in the forward detectors of PHENIX. In the forward direction, PHENIX has four sub-detectors installed, namely, the Muon Tracker, the Muon Identifier, the Magnet and the Muon Piston Calorimeter, and two sub-detectors in upgrade, namely, the Forward Vertex Silicon detector and the Nose Cone Calorimeter. The current PHENIX simulation program has these six sub-detectors with description in the GEANT3 detector construction functions. With the chain shown in Fig. 1, all information of the forward detector is exported to a zebra format file, and then it is converted to ROOT description and exported in GDML format. Fig. 2 shows the PHENIX forward detector converted from GEANT3 description. The detector is displayed with ROOT OpenGL graphic engine.

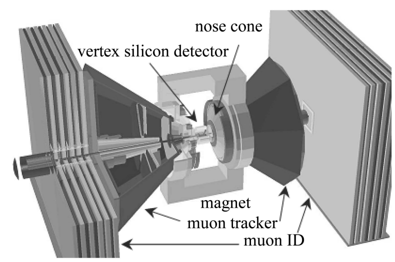

Fig. 2. The PHENIX Forward Detector displayed with ROOT OpenGL graphic engine.

In the GEANT4-based new simulation program, the detector construction is easily realized by importing the GDML file with the GDML-GEANT4 interface. Names of all definitions, solids, logical volumes and physical volumes are kept the same as those in the GEANT3 program. Copy numbers of physical volumes are also kept. It is convenient for the developers to use this information in the following steps of simulation like digitization, in a similar way that they once used in GEANT3. The detector description in the new GEANT4-based simulation program also keeps exactly the same as the one in the old GEANT3-based simulation program, which is guaranteed by the fully automatic conversion method.

3.2 ROOT geometry for reconstruction

ROOT has been used in PHENIX since its early version (v2.03). There was no geometry package in ROOT at that time; most of the functions in geometry service of reconstruction were realized by the developers. Without the support from a detector modeling system, the developers had to use approximate description for some complex detectors and many geometry related functions required a lot of work. In the latest version (v5.14) of ROOT that is used in PHENIX software, implementation of the new geometry package makes it convenient to develop geometry service in a more precise and more efficient way.

From the detector description converted from GEANT3, detector is already constructed in ROOT. All that developers need to do is to retrieve the information from this description and ROOT offers a lot of useful functions to realize it. For example, boundary searching is a main function that is used in reconstruction to judge whether a point is inside one volume or not. As shown in Fig. 3, for a silicon sensor sector in the disk of forward vertex sub-detector, the previous method to judge its boundary was to define the range of values on r and ϕ directions. However, this is only approximate description since the actual shape of the sensor is trapezoid. ROOT provides the function "Contains" to judge any complex shape's boundary, which is more precise and is easier to use. Defining coordinates transformation function is another hard wok for developers, because tree structure detector description usually creates a lot of levels of coordinates. ROOT also provides a function to easily transform the position of a point to any level of coordinate in the detector tree. Another useful ROOT function is volumes overlap checking, which helps to find the description bugs that may lead to error detector response in PHENIX simulation. By using the ROOT geometry package, the geometry service design in reconstruction becomes easier, but more precise and more robust.

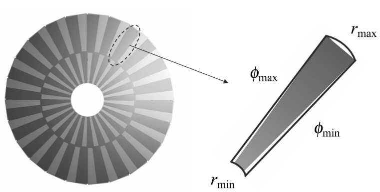

Fig. 3. Approximate boundary definition for a silicon sensor trapezoid.

3.3 Optimization in new sub-detector design

The method of exchanging the detector description between GEANT and ROOT is especially useful in the optimization of the design for a new subdetector. The Forward Vertex Silicon detector, as is shown in Fig.4, is a new sub-detector to be installed on PHENIX for precise vertex measurement. The structure of this sub-detector is under design and changes frequently. There are different versions of geometry of this sub-detector in simulation program. But it is difficult for the reconstruction geometry service to be updated and keep step with the changes all the time. By converting the GEANT3 detector data to ROOT, the reconstruction geometry service automatically keeps consistent with the simulation without modifying the source code. The source code only works on "index" of detector units. All geometry information of the units, including position, rotation, shape and material, are retrieved from the detector description file using the index. Different versions of detector design can exist at the same time so long as correct version of detector file is designated as the input of reconstruction geometry service. This method makes it easy to optimize the detector design by comparing the detector performance after reconstruction.

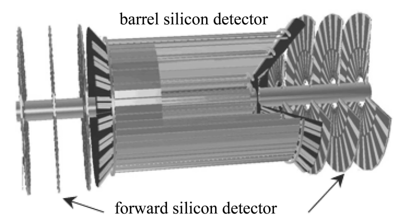

Fig. 4. The vertex silicon detector design in PHENIX.

4 Summary

In nuclear and particle physics experiments, detector description is used in many applications. GDML is a very convenient and efficient tool for describing and managing the geometry of complex detectors. A method to exchange the detector description data among GEANT3, GEANT4 and ROOT is realized, which reduces by a large amount the work of the developers. This method provides an easy way to share the detector description among different software programs and helps the optimization of the design for a new detector.

The authors gratefully acknowledge the collaboration and cooperation of the Offline Software Group of the PHENIX experiment, especially the PHENIX Group in Los Alamos National Laboratory for working on the forward vertex detector upgrade.

References

1 CERN Program Library Long Write up W5013. CERN, 1993
2 Agostinelli S et al. Nucl. Instrum. Methods A, 2003, **506**: 250
3 Brun R, Rademakers F. Nucl. Instrum. Methods A, 1997, **389**: 81
4 Brun R et al. Nucl. Instrum. Methods A, 2003, **502**: 676
5 Chytracek R et al. IEEE Trans. Nucl. Sci., 2007, **53**: 2892(Part 2)
6 http://www.w3.org/XML/
7 YOU Z Y et al. Using GDML for Detector Description in BESIII Experiment. Nucl. Instrum. Methods A, to be published
8 Adcox K et al. Nucl. Instrum. Methods A, 2003, **499**: 469