Contents lists available at ScienceDirect

Nuclear Instruments and Methods in Physics Research A

journal homepage: www.elsevier.com/locate/nima

# A uniform geometry description for simulation, reconstruction and visualization in the BESIII experiment[☆]

Yu-tie Liang[a,*], Bo Zhu[a], Zheng-yun You[a], Kun Liu[a], Hong-xue Ye[a], Guang-ming Xu[a], Si-guang Wang[a], Wei-dong Li[b], Huai-min Liu[b], Ze-pu Mao[b], Ya-jun Mao[a]

[a] *School of Physics and State Key Laboratory of Nuclear Physics and Technology, Peking University, Beijing 100871, China*
[b] *Institute of High Energy Physics, CAS, Beijing 100049, China*

ARTICLE INFO

*Article history:*
Received 11 February 2009
Received in revised form
21 February 2009
Accepted 23 February 2009
Available online 10 March 2009

*Keywords:*
Geometry description
ROOT
Geant4

ABSTRACT

In the BESIII experiment, the simulation, reconstruction and visualization were designed to use the same geometry description in order to ensure the consistency of the geometry for different applications. Geometry Description Markup Language (GDML), an application-independent persistent format for describing the geometries of detectors, was chosen and met our requirement. The detector of BESIII was described with GDML and then used in Geant4-based simulation and ROOT-based reconstruction and visualization.

## 1. Introduction

The geometry description is an important part of every Monte Carlo simulation application in experiments of nuclear physics and particle physics. The main task of geometry description is to specify the geometrical structures and properties of all the components of the detectors, such as the solid type, position, and materials of the detector. In general, applications, such as simulation engines, reconstruction, analysis and visualization use their own geometry description because these applications usually have their own native, different geometry description formats. The re-implementation of geometry for different applications may be not identical, which will cause systematic errors in the experiment.

The Beijing Spectrometer III (BESIII) [1] is a general purpose detector, which will be running at the Beijing Electron-Positron Collider II (BEPCII) for many important tau-charm physics targets. BESIII consists of four sub-detectors: Main Drift Chamber (MDC), Time-Of-Flight (TOF), Electro-Magnetic Calorimeter (EMC), and Muon Counter (MUC). As a complicated spectrometer, the geometry description of the BESIII detector was a very tough task. In the framework of BESIII Offline Software system (BOSS) [2], geometry was designed to be implemented once and then used in simulation, reconstruction and visualization. This way could not only save time and manpower in implementation of geometry, also assure identical geometry for different applications.

In this paper, we start with the implementation and application of geometry in BESIII. Then we introduce the consistency check of geometry interchange.

## 2. Implementation and application of geometry in BESIII

According to our requirement, GDML (Geometry Description Markup Language) [3,4] is chosen.

GDML is an XML (eXtensible Markup Language) [5] based detector description language, which provides a simple reading and writing mechanism, as well as extensibility. It is platform independent and language independent. All necessary information to describe a detector is provided through a set of ‘tags’ and ‘attributes’. The GDML description files are human readable and plain text, which is editable by any ordinary text editor.

There are two main bindings for GDML exchange with other applications, Geant4 [6] binding and ROOT binding. In the offline software of BESIII, Geant4 is used in simulation, and some ROOT [7,8] functions are used in reconstruction. So GDML is very suitable for our requirement. Fig. 1 shows the design of using GDML in BESIII.

GDML was in the early stage of development when it was first introduced into BESIII. The ROOT binding was not available, and one solid type in BESIII was not supported. In the process of using

[☆] Supported by National Science Foundation of China (10835001, 10775006, 10375002 and 10675004), the Research Fund for the Doctoral Program of Higher Education of China (20070001008), and the China Postdoctoral Fund.

* Corresponding author.
*E-mail address:* liangyt@hep.pku.edu.cn (Y.-t. Liang).

GDML, we improved it specially for BESIII. We developed the ROOT binding with C++, added new solid type, and developed the parameterization for trapezoids. With our effort, it was used smoothly in BESIII. Now the ROOT binding has been developed with python in the ROOT package, but we still use the ROOT interface with C++.

In BESIII, the MUC was described with GDML. The MUC is on the outside of BESIII. It has a barrel part and two endcaps (east endcap and west endcap). There are nine layers in the barrel and eight layers in each endcap. For each layer, two RPC layers and one pickup strip layer are compacted as a sandwich. In Fig. 2, the left graph is the structure of the description for the MUC. It is very close to the structure of the real detector. The right graph is the schematic view of an RPC. In GDML, two bakelite layers and one gas layer are described for each single RPC-layer. Dead area in the edge of the gas layer and little blocks in the gas to support two bakelite layers are also described. So the geometry description is close to reality.

For other sub-detectors, their geometries were implemented in the Geant4 C++ coding and then exported to a GDML file automatically with the GDML writer. So, all GDML files were collected and became the geometry source in all applications. Fig. 3 is the whole BESIII detector visualized using the ROOT 3D graphics.

The simulation, reconstruction and visualization start with these GDML files as geometry source.

In simulation, the GDML files are loaded into Geant4 using the GDML-Geant4 interface. These files only contain the information of geometry and material. The information about sensitive detectors is set in Geant4 code after geometry is constructed and detector efficiency is picked up from a database.

In reconstruction, the ROOT geometry is generated with the GDML-ROOT interface. We developed a geometry service in the framework of BOSS [2] to pick up useful information from the ROOT geometry, such as the position of fired strips. Taking the track reconstruction of the MUC as an example, fired strips from Raw Data are collected and then original tracks are reconstructed after getting the positions of fired strips from the geometry service. There are some other benefits of using ROOT geometry in reconstruction. For example, boundary searching is often used in reconstruction to judge whether a point is in a volume or not. This is difficult to achieve by mathematical calculation for complex detectors, but using the ROOT package [9] it is easy because ROOT provides the function 'Contains' to judge the boundary of any shape. The ROOT package also provides a function to easily transform the position of a point to any level of

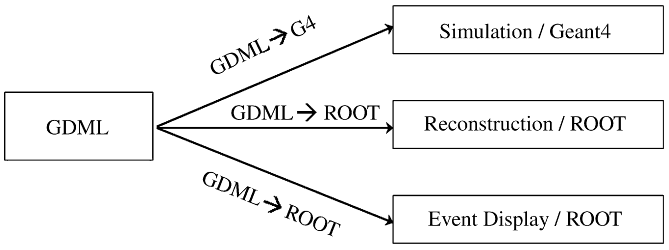

**Fig. 1.** Design of the application of GDML to BESIII.

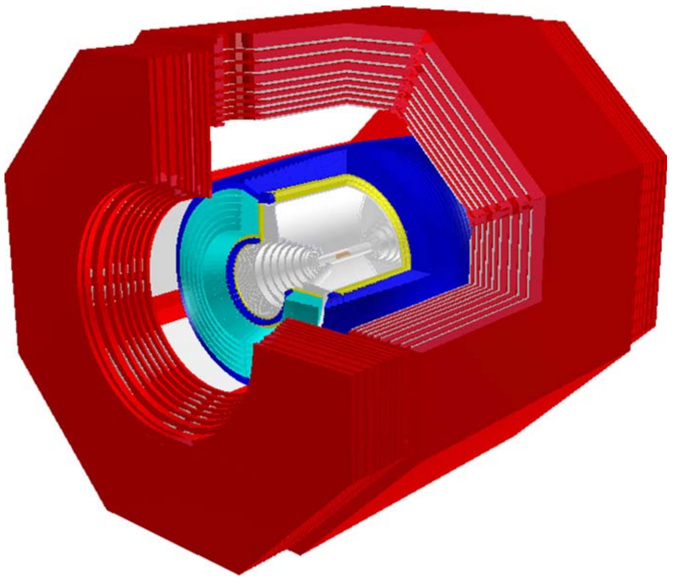

**Fig. 3.** The BESIII detector visualized using ROOT.

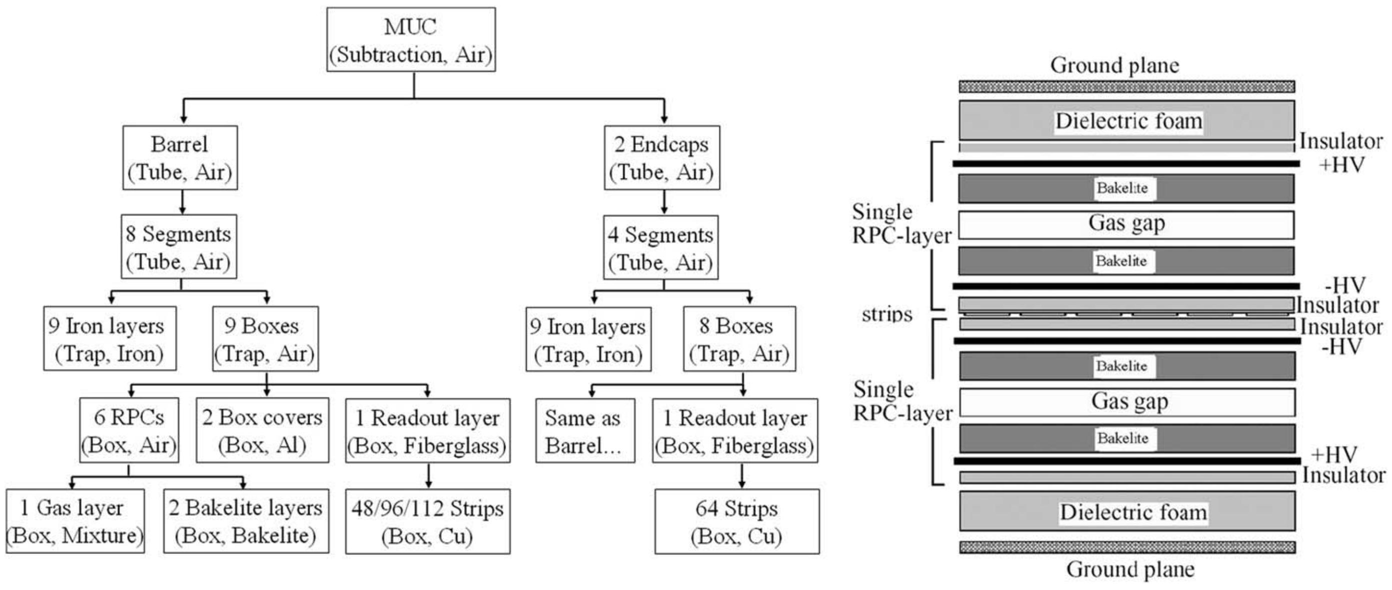

**Fig. 2.** Structure of the MUC and the RPC in GDML.

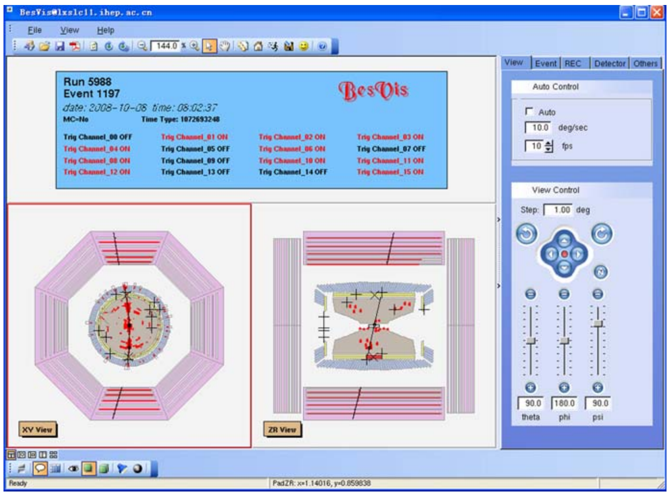
Run 5988
Event 1197
date: 2008-10-08 time: 08:02:37
MC=No Time Type: 1072693248
Trig Channel_00 OFF Trig Channel_01 ON Trig Channel_02 ON Trig Channel_03 ON
Trig Channel_04 ON Trig Channel_05 OFF Trig Channel_06 ON Trig Channel_07 OFF
Trig Channel_08 ON Trig Channel_09 OFF Trig Channel_10 ON Trig Channel_11 ON
Trig Channel_12 ON Trig Channel_13 OFF Trig Channel_14 OFF Trig Channel_15 ON

**Fig. 4.** One muon pair event in the BESIII visualization tool.

coordinate in the detector tree. Up to now, the MUC, EMC and TOF are using ROOT geometry in the reconstruction.

Event display is an important tool in an experiment of particle physics. It can help to debug offline software, to monitor data quality, to analyze data and understand the experiment, etc. In BESIII, the visualization tool, BESVIS, is developed with ROOT. It reads GDML files and generates ROOT geometry. It can display an event in OpenGL mode or X3d mode. The 2D display, which is not supported by ROOT, can also be achieved by projection of geometry onto the XY (pipe) plane or ZR (vertical section) plane. The speed of reading and parsing GDML file is slow (about 1 min). The simulation and reconstruction are not seriously affected by this but for the event display, 1 min for initialization is a little long. So we export the ROOT geometry into a root file which contains all the geometry information. Later, BESVIS will read the root file to initialize geometry which only costs 3 s. Fig. 4 shows the display of an event.

## 3. Consistency of geometry interchange

In BESIII, the MDC, TOF and EMC implement geometry in Geant4 geometry format. Their GDML files are exported from Geant4 geometry. So we have two optional geometry construction methods for simulation: one is from Geant4 code and the other is from GDML file. It is natural that these two geometries should be identical, taking no account of computing precision. We did simulation with these two geometries, and found that the results were different, which means that two geometries are different. We studied this and showed that the difference is due to computing precision indeed. In simulation, a gamma was generated and passed through the MDC, TOF, and EMC. Geant4 kernel divides the simulation process into many steps. For each step, we retrieved all information about particles in the detector, such as energy deposit, step length, etc. Any tiny difference in these two geometries will cause big difference in the step information. We found that the difference in geometry was from the definition of materials, G4Trap, rotation matrix and unit conversion of the angle.

| Difference due to | Description of the difference |
|---|---|
| Materials | Input mass number, but store molar mass in memory |
| G4Trap | Input angle($\theta$, $\phi$, $\alpha$), but store tan($\alpha$), etc. in memory |
| Rotation matrix | Input three values rotated around XYZ, but store nine matrix elements |
| Unit of angle | Input radians, but GDML use degrees |

Taking materials as an example, when we define Beryllium in Geant4, Beryllium = new G4Material(name = "Be", z = 4.0, 9.012182*g/mole, 1.848*g/cm3); proton number, mass number (9.012182) and density are used. In memory, the molar mass (9.012182*g/mole) is stored, but not the mass number. When this material is written to GDML file, the mass number is calculated from the molar mass; Beryllium − >GetA() = a/(g/mole) = 9.01218. So, we could see the difference of mass number which is caused by computer precision. After fixing all these difference in two geometries, we simulated 1000 gamma with energy 1 GeV. We found that all the information for each step was identical with the two geometries. We conclude that GDML as an interchange format is reliable.

## 4. Summary

GDML is an XML-based application-independent detector geometry description language. It has been used in many fields, such as high energy physics, space research, and medical physics, etc. [4]. In 2003, when GDML was in the early stage of development, we introduced it into BESIII. We developed the GDML-ROOT interface with C + +, added a new solid type, and developed the parameterization for trapezoids. In BESIII, the simulation, reconstruction and visualization use the same geometry with the help of GDML. It has been proved to be flexible, convenient, and reliable after using GDML for 6 years in BESIII. We also checked the consistency of geometry interchange. The way of using GDML in BESIII is very successful and has been used in the Daya Bay Reactor Neutrino Experiment.

## References

[1] Preliminary Design Report of the BESIII detector, January 2004, IHEP-BEPCII-SB-13.
[2] The Offline Software for the BESIII Experiment, in: Proceeding of CHEP06, Mumbai, India, 2006.
[3] R. Chytracek, et al., IEEE Trans. Nucl. Sci. NS-53 (2007) 2892 (Part 2).
[4] Radovan Chytracek, et al., IEEE Trans. Nucl. Sci. NS-53 (5) (2006).
[5] ⟨http://www.w3.org/XML⟩.
[6] S. Agostinelli, et al., Nucl. Instr. and Meth. A 506 (2003) 250.
[7] R. Brun, F. Rademakers, et al., An object-oriented data analysis framework, arXiv: ⟨http://root.cern.ch⟩.
[8] R. Brun, F. Rademakers, Nucl. Instr. and Meth. A 389 (1997) 81.
[9] R. Brun, et al., Nucl. Instr. and Meth. A 502 (2003) 676.