

GEANT4: SIMULATION FOR THE NEXT GENERATION OF HEP EXPERIMENTS

Giani Simone

CERN, Geneva, CH

DRDCRD44 project

CERN (CH); TIFR (India); Sofia Univ. (Bulgaria); LAPP, IPN Orsay, Saclay Phys.Inst. (France); LPI Moscow, IHEP (Russia); KFKI (Hungary); Hiroshima Inst. of Tech., KEK, Kyoto Univ., Naruto Univ. of Education, Tokyo Metropolitan Univ., International Christian Univ., Fukui Univ. (Japan); LBL, LLNL, Massachusetts Univ., Caltech, KLA, IBM (USA); Tampere Univ., Helsinki Univ., IPT Helsinki (Finland); RWTH, Desy (Germany); Manchester Univ., Cambridge Univ., Cray Systems (UK); NIKHEF (The Netherlands); INFN (Italy)

GEANT4(RD44) is a world-wide collaboration of scientists aiming to create the detector simulation toolkit necessary for the next generation of HEP experiments. A large variety of requirements also come from heavy ions physics, CP violation physics, cosmic rays physics, medical applications and space science applications. In order to meet such requirements, a large degree of functionality and flexibility has to be provided. GEANT4 is exploiting Object Oriented Technology to achieve these goals. The most relevant Object Oriented methodologies have been studied and a large number of tools and libraries have been investigated and evaluated. An overview of the GEANT4 analysis and design model will be given and the main components of the object models discussed. The main functionality of a first prototype will also be described. An overall view of the collaboration's structure and strategy will also be given.

1 Introduction

The DRDC P58 proposal [1] has been presented in September 1994 and approved in November as the DRDCRD44 (GEANT4) project with the following milestones for the first year:

- . produce a **global** Object Oriented Analysis and Design for the **GEANT4** simulation toolkit;
- . develop a C++ [2] prototype of the geometry and tracking in order to evaluate its performance in comparison with GEANT3 [3].

In this paper we present the main lines and results of the work developed in the first nine months of the project. The overall **timescale** for the project is four years, aiming to provide a preliminary working version after the first two years.

2 Organization

The project is a world-wide collaboration of scientists, from CERN, Japan, the US, Russia, the UK, France, Germany, Finland, Italy, Hungary, Bulgaria and India.

Therefore, the communication between the **different** groups had to be addressed by scheduling regular video-conferences and by travel. Small workshops held either at **CERN** or at **KEK** (implying quite frequent intercontinental personal travel for some collaborators) have proved to be extremely useful and vital for discussing and solving technical issues. **Video-conferences** have helped more to monitor progress and to set schedules. A very heavy flow of e-mail must also be considered as important, as **well** as the **aquisition** and sharing of the most relevant literature. Finally, at the **OOA&D** level, the use of CASE tools has definitely helped in communication between widely dispersed collaborators.

At the time of writing, the project accounts for **more** than 70 collaborators having different backgrounds and interests. Physicists, computer scientists, engineers have contributed in **different** areas. Several collaborators are the main authors of existing simulation packages, specialized in different areas and written with **different** software technology. External consultants have been engaged especially on Object Oriented Analysis and Design and on Software Engineering. Particular attention has been paid to using **C++** as an implementation technology of the Object Oriented paradigm. Collaborators working in HEP experiments have contributed in setting the requirements for **GEANT4**. The most effective exploitation of the manpower could occur **only** after the analysis of the problem domain performed by a core team of people. Then **subdomains** and a hierarchy of dependencies between them have been identified, and a number of working groups have been created in close relation with the **subdomains** and the expertise of the different collaborators.

In order to organize the temporal development of the work, setting schedules and milestones, a time-table has been produced and regularly updated: it shows the main activities performed (or to be performed) month-by-month and it emphasises the spiral, iterative approach followed in the software development process that will be discussed in detail in the next section.

3 The software development process

One of the tasks of the **R&D** project is to watch carefully the software development process. In the **GEANT4** context, the main-stream Object Oriented methodologies have therefore been studied and examined. The purpose was not to adopt a particular Object Oriented methodology (given that they are **all** still evolving and we did not want to tie ourselves to a specific CASE tool) but to try to use them and eventually make a synthesis or adapt them to the specific features of our problem domain. A first result was the identification of a set of actions, and the corresponding deliverables, for the so-called phases of Requirements, 00-Analysis and 00-Design. This has allowed us, for example, to move almost transparently from OMT [4] diagrams to **Booch** [5] deliverables. The CASE tool supporting the **Booch** methodology has been revealed to be the most powerful in terms of reverse engineering and in terms of richness of notation. Consequently, we have finally produced all the deliverables in the **Booch** notation.

The emerging ESA **PSS-05** Software Engineering Standards [6] have also been evaluated. They give a set of guidelines (to set the corresponding deliverables) in a very high-level way, from the requirements phase to the version control and mainte-

nance of a software product. We are attempting to adopt PSS-05 principles whilst trying to remain flexible and independent of the approach followed for analysis and design. We have found in PSS-05 a clear added-value in the Requirements management phase, therefore the GEANT4 User Requirements Document [7] is PSS-05 compliant. One of the main authors of PSS-05 is collaborating with RD44 to study the feasibility of Analysis and Design PSS-05 documents, starting from deliverables of an Object-Oriented methodology (which remain the first priority for GEANT4).

The potential re-use of existing software, in particular C++ class libraries, is also considered a fundamental issue in the software development of GEANT4. In the first months of the project, a large variety of commercial and free class libraries have been studied, tested and evaluated, together with packages developed in HEP. The picture emerging from such an investigation is impressive: for a few hundred dollars, up to a few thousand dollars, there are complete C++ software layers available on the market, providing the source or free run-time licences, going from basic utilities software, to heap managers, garbage collectors, mathematical libraries, graphics class libraries (giving platform independence ranging from Xwindow to GL, MSwindows and Macintosh) and GUI builders. Nowadays, it is a matter of choosing the top-quality products, not of looking for something. In addition, from those products, it is possible to build a complete and uniform software environment, not only from the programming language point of view, but also because it happens that the GUI builder one may choose is based on the same utilities class library one may have chosen, and that the CASE tool one is using is able to reverse engineer them both. Also, these C++ packages are often excellent implementations of the Object Oriented paradigm. Finally, for HEP specific code, we have managed to re-use GISMO [8] and CLHEP [9] code without any problem.

Last, but not least, is the RD44 focus on standards (ISO, drafts and de-facto industrial standards). In the engineering and CAD systems domain, the STEP [10] standard for defining data (in particular geometrical and material data) is now mature: Rolls Royce and Boeing are exchanging geometrical models defined with STEP, toolkits to convert automatically the data definition language into C++ are available, and the definition of an application protocol for object data-bases is coming. Consequently, the complete GEANT4 geometry is STEP compliant. On the I/O side, RD44 relies on the RD45 [11] project, which aims to provide object-persistency via ODMG [12] compliant data-base systems. The main ODMG compliant object data-bases offer C++ binding, pointing once again to the uniform environment mentioned above. In terms of basic software layers, the Standard Template Library (STL) [13] is supposed to replace some commercial software used now, as soon as it becomes available on every platform with a C++ compiler.

4 Requirements

Given the large numbers of HEP experiments represented in RD44 (ATLAS, CMS, L3, NA49, STAR, BaBar, ALICE, ALEPH, OPAL, NOMAD, H1, CHORUS), the Requirements Document is aimed at expressing very closely the needs of the physicists working in the collaborations. As mentioned above, the GEANT4 User Requirements Document was produced (and it will naturally evolve) following the ESA

PSS-05 standard. It is available on the Web at the URL
http://asdwww.cern.ch/pl/geant/geant4_public/pub_requirements.ps.
This document is structured in three main parts:

- Introduction
- General Description
- Specific Requirements

The Introduction describes the main purpose and the scope of the software to be produced. The General Description defines the main capabilities and constraints of the system, together with the user characteristics. In addition, a general context diagram shows the main interactions of the system with the user and the external software. The Specific Requirements are classified in Capabilities Requirements and Constraints Requirements, and contain an ordered listing of the requirements statements.

5 Analysis and Design

“The translation barrier between analysis and design is just one symptom of a larger problem. Methodologies that separate analysis and design presume that designs are based upon analyses — that a design cannot begin until an analysis has been completed. This presumes that the analysis does not depend upon the design in any way. In most **case, this is incorrect.**

^u.... we consider the notion that analysis and design are nearly concurrent efforts; knowledge from the analysis flows into design, and knowledge from the design flows into analysis. We will view the analysis and design effort as a continuum of activity, beginning primarily with analysis activities and ending primarily with design activities, but with no clear separation of these activities in any part of the process.”

This quotation from R. C. Martin [14] synthesizes our experience and strategy for Object Oriented Analysis and Design. Those concepts become especially important when designing a complex system like a complete simulation toolkit with the functionality required for **GEANT4**.

Here follows a short description of the main components of the **GEANT4** toolkit. There is a close correspondence between these components and the responsibilities of the working groups.

- . Event and **Track** Management: Manages the generation of events, interfaces to event generators, and any produced secondary particles. Its role is principally to provide particles to be tracked to the Tracking Management, and it covers the Event Management and Event Generator **class** categories.
- Tracking Management: Propagates a particle by **analysing** the factors limiting the step and applying the relevant physics processes, and it covers the Track Management class categories.

- **Particle Interactions in Matter:** Manages the interactions of particles in matter, including the definition of materials, particles, physics processes and their **parameterisations**. It covers the Particle Definition, Physics Process and Materials class categories.
- **Geometry:** Manages the geometrical definition of the detector (solid **modelling** and interactions with CAD systems) and the computation of distances to solids (also in a magnetic field). It covers the Geometry, Magnetic Field and CAD-Interface class categories.
- **Hits and Digitization:** Manages the creation of hits and their use for the digitization phase, and it covers the Hit and Digitization class categories.
- **Visualisation and Visualisation Framework:** Manages the visualization of solids, trajectories and hits, and interacts with underlying graphical libraries. It covers the Visualization class category.
- **Interfaces:** Includes the production of the GUI and the interactions with external software (**OODBMS**, reconstruction etc.). GUI issues and more general interfaces cover the class categories **UI-GUI** and Parameter Management.

The most important deliverables for each component (produced with different levels of detail for what can be called the Analysis phase and for what can be called the Design phase) are:

- the Class Diagrams, describing the logical model, which represents the classes and their relations, together with their behaviour;
- the Class Specifications, which are a textual version of the header files generated by the CASE tool;
- the Scenario Diagrams or Object Interaction Diagrams, describing dynamic models, which represent the real messaging between class instances during a run.

6 The prototype

At the time of writing, four aspects of the prototype have been developed and tested sufficiently to be mentioned here.

The first is the capability to track in BREP models, coming straight from STEP compliant CAD systems, including Non Uniform Rational **B-Splines** (NURBS). For the first time in HEP, it will be possible to compare simulation results obtained in the ad-hoc detector model (as used today by any simulation program for performance reasons) with the simulation results obtained in the engineering detector model. Only if the results are the same, is the ad-hoc version certified to be correct and ready for production use.

The second is the possibility to translate automatically GEANT3 geometries (written in FORTRAN files or stored in RZ Zebra [15] files) into C++ code which would represent such geometries in GEANT4.

The third is the development of the Geometry component, including the tracking capabilities of locating a point in arbitrarily complex geometrical structures, and of determining the distance between a point and the closest volumes in such a structure. The preliminary results of benchmark comparisons with **GEANT3** are very promising.

The fourth is the development of the 'hacking and Event Management' components, allowing generation and tracking of 'geantinos'. The results of benchmark comparisons of these **full** events with **GEANT3** are again very promising.

7 Conclusions

- **OO** has real advantages, especially for large systems, developed by a world wide collaboration, given the emphasis **OO** puts on the components and class interfaces.
- **OO** methodologies and software engineering are both relevant and important for HEP simulation software. Main-stream **OO** methodologies support **C++** development tools.
- **C++** is **performant** and the **C++** compilers produce **well optimized code**. Most of the software needed or desirable is easily available.

8 References

1. A. **Dellacqua** et al., "**Geant4**: an object-oriented toolkit for simulation in **HEP**". **CERN/DRDC/94-29**, 1994.
2. B. **Stroustrup**, "The **C++** programming language", Addison **Wesley**, 1992.
3. **CN/ASD**, "**Geant**, detector description and simulation tool", **CERN**, 1994.
4. J. **Rumbaugh** et al., "Object-oriented modeling and design", Prentice-Hall, 1991.
5. G. **Booch**, "Object-oriented analysis and design", **Benjamin-Cummings**, 1994.
6. **ESA**, "Guide to user requirements definition phase", **ESA PSS-05**, 1994.
7. **RD44**, "**Geant4** user requirements document", **CERN**, in preparation.
8. W. B. Atwood et al., "The **Gismo** project", **C++** report, 1991.
9. L. **Lonnblad**, "**Clhep**, a class library for **HEP**", **CERN**, 1992.
10. **ISO 10303**, "Industrial integration systems and integration", **ISO TC184/SC4**, 1992.
11. J. **Shiers** et al., "A **persistant** object manager for **HEP**", **CERN/DRDC/94-30**, 1994.
12. **ODMG**, "Object data-base management group 93", 1993.
13. B. **Stroustrup** et al., "**STL**, **C++** Standard template library draft", 1994.
14. R. C. Martin, "Designing object-oriented **C++** application using the **Booch** method", Prentice-Hall **Inc**, 1995, **ISBN 0-13-203837-4**.
15. **R. Brun** et al., "Zebra user manual", 1988.