Proceedings of the Second International Workshop on EGS, 8.-12. August 2000, Tsukuba, Japan
KEK Proceedings 200-20, pp.23-30

Status of the Object-oriented EGS Interface Project

A. M. Yacout, W. L. Dunn, W. R. Nelson[1], P. Lui[1],
A. F. Bielajew[2], H. Hirayama[3] and Y. Namito[3]

Quantum Research Services, PO Box 52391, Durham, NC 27717, USA
[1]*Stanford Linear Accelerator Center, PO Box 4349, Stanford, CA 94309, USA*
[2]*The University of Michigan, 2355 Bonisteel Boulevard, Ann Arbor, MI 48109, USA*
[3]*High Energy Accelerator Research Organization (KEK), Oho, Tsukuba-shi, Ibaraki-ken 305-0801, Japan*

Abstract

The object-oriented EGS interface project seeks to simplify – using modern object-oriented and visual user interface techniques – the geometry and scoring aspects of the process of running the EGS code. The project will create an extremely user-friendly EGS package that retains and exploits the well documented physics advantages of EGS but removes the requirement that the user write HOWFAR and AUSGAB subroutines to define the geometry and scoring aspects of each new problem. In addition, several physics enhancements will be incorporated in EGS5. Although EGS5 will be able to be used in the traditional way – in a stand-alone fashion with users writing their own geometry and scoring subroutines – it is designed to be used in a completely new way – linked to a user interface through which users can manage all aspects of problem specification and code operation. This paper concentrates on the object-oriented user interface, which will dramatically simplify defining problem-specific detail for EGS. The "EGS5+VUI1" package will allow users to solve independent problems by run-time linking of the EGS5 code with class libraries that encapsulate the geometry and scoring aspects of each problem. Some simple example problems are considered in order to illustrate features of the EGS5+VUI1 package.

1 Introduction

The Electron Gamma Shower (EGS) Monte Carlo coupled electron-photon transport code has enjoyed much success and broad acceptance over a period of many years. However, the current version, EGS4[1], has now been released for over a decade without formal upgrading and re-release. Further, EGS4 requires that the user be proficient enough to program the geometry and scoring aspects of each problem in subroutines that must be compiled and linked to the static part of the code. Use of the Mortran pre-processor[2] and several "canned" geometry macros (e.g., `$PLAN1,$PLAN2P,$CYLNDR,` etc.[3]) can simplify the process, but solving different problems nevertheless requires compilation and linking of user code to EGS physics code. This places a substantial burden on the user and, not insignificantly, permits errors in logic or implementation, especially among inexperienced users.

Other general-purpose Monte Carlo radiation transport codes, such as MCNP and GEANT, deal with the geometry/scoring problem by building extensive libraries and tools. This approach has the disadvantages of creating a large code overhead and limiting the user to those geometry and scoring estimators that are built-in. EGS, on the other hand, has allowed user flexibility by allowing the user to write the geometry/scoring aspects of each problem. This leads to a leaner code but adds time and requires that the user be a competent programmer. The application of object-oriented (OO) programming techniques allows the "best of both worlds." In our OO approach, we respresent geometrical shapes, sources, and scoring estimators as classes. Objects selected from these classes are linked to the physics code at run time, without the need to compile and link user code. Use of a scripted input provides a flexible way to build objects at run time.

In the following, we discuss our efforts to significantly upgrade the EGS4 code and to develop an object-oriented EGS interface that will obviate the need for the user to write problem-specific code.

2 The EGS5 and Object-oriented EGS Interface Projects

The authors are involved in a multi-institutional effort to significantly upgrade the EGS4 code. This effort will result in an enhanced version of EGS, EGS5, and an OO user interface to the enhanced physics code. The user interface will contain a visual user interface (VUI) through which the user will communicate with the rest of the code; the combined package will be called EGS5+VUI1. It is anticipated that EGS5+VUI1 will be released before the end of the year 2001.

To date, EGS4 has been modified to conform to modern programming conventions (such as INCLUDE statements for COMMON blocks, etc.) and to incorporate low-energy photon enhancements developed at KEK. The resulting EGS4.2 version has been extensively benchmarked against EGS4. Further physics enhancements are being developed that will be incorporated into EGS. The final EGS5 physics code will include at least the following enhancements:

- Improved low-energy photon transport, e.g.[4]
- Improved electron transport modeling, e.g.[5]
- Low energy electron cross section handling[6]
- Photoelectric angular distributions[7]
- Improved bremsstrahlung photon angular distributions[8]
- K and L shell fluorescence[9]

Some of these physics enhancements that will be incorporated in EGS5 are discussed in other presentations[10, 11] at this workshop. In addition, EGS5 will incorporate cross section generation within the code, rather than requiring that the user run a pre-processor, such as PEGS, to create the necessary cross section files for different problems.

The other significant effort is to construct a powerful object-oriented user interface to EGS that will revolutionize the way EGS is run in the future. The OO paradigm is well suited to support our primary goal of producing a reusable code – reusable in the sense that modifications to one component do not require modifications to the other components of the code package. The main components in our case are the physics in EGS, the VUI, and the user code (including the geometry in HOWFAR and the scoring estimators in AUSGAB). It is noted that "modifying" can be interpreted also in the sense of "adding", so that in EGS5+VUI1 geometry and other classes can be added without having to rebuild the rest of the package.

The encapsulation property of OO programming, in which each object encapsulates (contains) its data and code (functions), provides the first means of breaking the problem into meaningful components. Each object is capable of manipulating its own data (properties) to independently provide the information required by other parts of the system. Thus, for instance, tracking a particle inside a particular geometrical shape is handled completely by the methods encapsulated in the object itself. The implementation of the tracking code is thus completely hidden from the rest of the system. The inheritance property of OO programming allows the construction of new object types (classes) based on existing types. New classes "inherit" some or all of the properties and methods of the base class. This property allows us to re-use existing code and, more importantly, provides a way to specify and establish a common interface for all object types (existing or to be added) by deriving them from one common base class. For example, geometrical objects are derived from one common ancestor; this base class defines the required methods for particle tracking (methods to determine if a specified spatial point is inside an object, to determine if a track intersects an object surface, etc.). Classes inherited from this class modify these methods according to the geometry of the shape they represent. Finally, the objects created from these classes are polymorphic: the methods in the different class types have identical names, but at run time, the correct method is called according to the type of the object being used. These properties of OO programming enable us to systematically implement the tracking and scoring aspects of Monte Carlo in general, regardless of the details of a specific problem.

3 Description of the User Interface

A schematic of the EGS5+VUI1 package is given in figure 1. The block marked “EGS5” contains

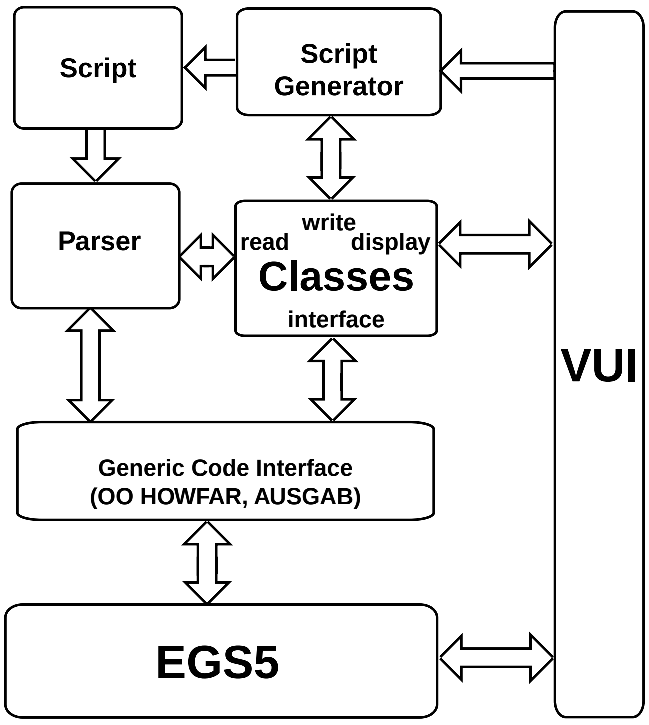

Figure 1: A schematic of the EGS5+VUI1 package.

the enhanced physics models. For the present, it will be programmed in Fortran77 and can be used in the traditional way (write problem-specific routines, prepare an input file, compile and link the routines with EGS5, and execute the resulting code for the given problem). However, EGS5 can also be used in a much simpler way, as part of the EGS5+VUI1 package. In this case, the user will set up the problem by interacting only through the VUI, which will be a full-function interface that uses icons, dialog boxes, menus, etc. to set up a specific problem. Geometric objects, materials, sources, and estimators will be selected from pre-defined libraries. Dimensions, positions, etc. will be entered into dialog boxes. Once an object has been registered, its methods will be used to draw it on the screen. Geometrical objects can contain other objects, including other geometrical objects, allowing transport in complex geometrical systems. Sessions can be saved and retrieved and the user can initiate execution of EGS merely by clicking on an appropriate icon, after the problem has been fully set up. The VUI will allow the user to construct a new problem from the ground up or by modifying an old session.

The Script Generator will generate appropriate script from the specific objects that the user selects for a given problem. The script incorporates the descriptions of all objects representing the geometry, materials, sources, and estimators in the problem using a simple syntax. Thus, no separate input data file needs to be prepared. Also, no predetermined order of records is required, as in the numerical input files used previously with EGS. Example scripts are given in the next section for three simple problems.

EGS5+VUI1 will contain a Class Library, i.e. a library of geometry, source, material, and scoring classes. The classes will contain methods that perform operations appropriate for their types. For example, each geometric class contains the following methods:

1. Method to register its name, creator method, properties (names and read/write handlers), and display methods
2. A creator method to instantiate a new object of the class
3. Property handlers (methods to read and write a value of a property)
4. Display methods to display different views of the object and an interface to query required properties
5. A method to determine if a specified point is inside an object
6. A method to determine if a track intersects an object
7. A method to determine the entry point to an object along a specified track
8. A method to determine the exit point from an object along a specified track

One of the significant features of EGS5+VUI1 is that a procedure will exist for adding new classes. Thus, if a user desires to incorporate for a specific problem a special or unusual geometrical object or scoring estimator that is not part of the existing Class Library, a procedure will exist for adding one or more classes. The new classes are added to the run-time library without rebuilding the package and will be available for all future problems. In this way the library of classes in EGS5+VUI1 can grow as needed to handle essentially arbitrary problems. It is noted, however, that the initial Class Library will be sufficient to treat a very broad range of problems.

The Parser creates objects from the script at run time, using simple lexical rules. Objects created by the Parser are instances of classes stored in the Class Library and registered at start-up by the VUI. The Parser will provide high-level functions for developers of new classes. The HOWFAR and AUSGAB routines are reduced to generic (universal) routines that implement systematic tracking and scoring algorithms and pass information between the EGS5 code and the problem-specific objects. The use of generic HOWFAR and AUSGAB routines is possible because of the OO organization. A single query is issued to a container called the "Mother Volume." The Mother Volume queries all internal geometrical objects and volumes, which in turn query their internal gometric objects and volumes. In this way, particle location and transport can be tracked with a single call at each transport step.

4 Demonstration

We consider three simple problems as a means to demonstrate the EGS5+VUI1 approach. These problems were run in an early part of the project using EGS4. Problem 1 is similar to the standard EGS demonstration problem, namely a mono-directional source emitting 1-MeV photons into a polyethylene slab along the +z direction; the slab is bounded on one side by a vacuum and on the other by air (see figure 2). We desire the total energy deposited in the polyethylene slab. The second and third problems involve a sphere of polyethylene surrounded by vacuum. A source similar to that in the first sample problem is located at the center of the sphere (see figure 3). We sought the energy deposited in the sphere, for Problem 2, and the particle flux at the outside surface of the sphere, for Problem 3. This suite of problems, though simple, allows us to demonstrate the OO approach for different geometries (slab versus sphere) as well as different estimators (energy deposition versus boundary crossing).

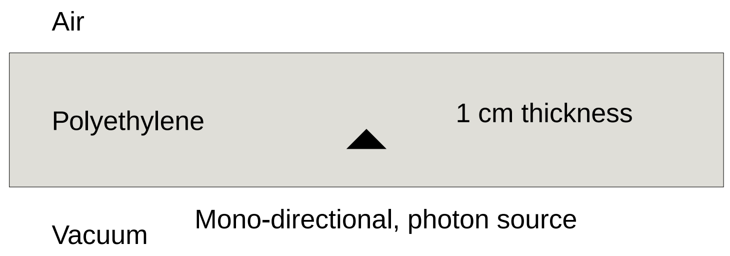

Figure 2: A mono-directional source emits 1-MeV photons into a polyethylene slab of thickness one cm. We seek the energy deposited in the polyethylene slab.

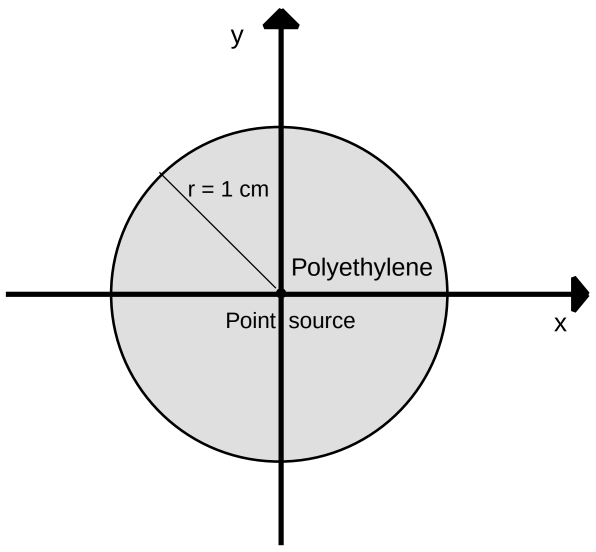

Figure 3: A mono-directional source emits 1-MeV photons into a polyethylene sphere of radius one cm. We seek in Problem 2 the energy deposited in the sphere and in Problem 3 the flux of particles at the sphere boundary.

We solved all problems using EGS in the traditional manner. This required writing two HOWFAR routines (one for the slabs and one for the sphere) and two AUSGAB routines (one for energy deposition and one for flux). The HOWFAR and AUSGAB routines were compiled and linked in various ways with the EGS4 physics code to create three executable codes. We then ran each code with 100,000 histories to obtain results in the conventional way. We also solved the three problems using the OO approach. This involved writing a script for each problem (this task eventually will be performed by the Script Generator, which has not yet been completed) and then letting the parser take each script and create the objects from a library which was linked at run time with the EGS4 object module. The scripts for the three problems are given below.

Slab Energy Deposition Script (Problem 1)

```
TMaterial:POLY{tag="POLY";id=1;};
TMaterial:AIR{tag="AIR";id=2;};
TSource:Source{energy=1.;id=0;location=(0.0,0.0,+0.0);
        number=100000;seed=5555};
TEstimator:Estimator{id=1;};
TVolume:MotherVolume{
        TSlabZ:Vaccum{thickness=100.0;TPoint=(0,0,-100.0);
        ecut=0.01;pcut=0.01;};
        TSlabZ:PolySlab{thickness=1.0;TSource=Source;
        TPoint=(0,0,0)TMaterial=POLY;TEstimator=Estimator;
        transport=1;ecut=.01;pcut=0.01;};
        TSlabZ:AirSlab{thickness=100.0;TPoint=(0,0,1.0);
        TMaterial=AIR;transport=0;ecut=.01;pcut=.01;};
};
```

Sphere Energy Deposition Script (Problem 2)

```
TMaterial:POLY{tag="POLY";id=1;};
TMaterial:AIR{tag="AIR";id=2;};
TSource:Source{energy=1.;id=0;location=(0.0,0.0,+0.0);
        number=100000;seed=5555};
TEstimator:Estimator{id=1;};
TVolume:"MotherVolume"{
        TSphere:PolySphere{radius=1.0;TSource=Source;
        center=(0,0,0);TMaterial=POLY;TEstimator=Estimator;
        transport=1;ecut=.01;pcut=0.01;};
        TSlabZ:EMPTYSlab{thickness=100.0;TPoint=(0,0,-50.0);
        transport=0;ecut=.01;pcut=.01;};
};
```

Sphere Flux Script (Problem 3)

```
TMaterial:POLY{tag="POLY";id=1;};
TMaterial:AIR{tag="AIR";id=2;};
TSource:Source{energy=1.;id=0;location=(0.0,0.0,+0.0);
        number=100000;seed=5555};
TFluxEstimator:Estimator{id=1;};
TVolume:"MotherVolume"{
        TSphere:PolySphere{radius=1.0;TSource=Source;
        center=(0,0,0);TMaterial=POLY;TEstimator=Estimator;
        transport=1;ecut=.01;pcut=0.01;};
        TSlabZ:EMPTYSlab{thickness=100.0;TPoint=(0,0,-50.0);
        transport = 0;ecut=.01;pcut=.01;};
};
```

Review of the scripts indicates the simplicity of this approach. The user interactively selects the type of source, the geometric objects, the materials, and the appropriate estimator and specifies desired values for the necessary parameters. The script generator then creates a script that embeds this information. For instance, in all three problems, there is only one source, which is located at the origin and emits mono-energetic photons at 1 MeV. The number of hisotries for each source is also embedded in the script. Further, in Problem 1, the Mother Volume consists of three slabs: a vacuum slab that extends from z = -100 cm to $z = 0$, a polyethylene slab that extends from $z = 0$ to $z = 1$ cm, and an air slab that extends from $z = 1$ cm to $z = 101$ cm; all three slabs are centered at $x = 0$ and $y = 0$. Only the polyethylene slab contains a source and an estimator. The Mother Volumes in Problems 2 and 3 each consist of a vacuum slab (arbitrarily extending from z = -50 cm to z = 50 cm) and an embedded polyethylene sphere of radius 1 cm. In fact, the scripts of Problems 2 and 3 differ only in the one line that identifies which estimator to use (TEstimator in Problem 2 and TFluxEstimator in Problem 3). The classes for TEstimator and TFluxEstimator contain the necessary methods to score the appropriate quantities.

The results obtained from the two approaches are compared in the Table. It is clear that the results are equivalent, but that in the OO case no subroutines had to be re-written and no recompilation for each problem had to be performed. Of course users of the EGS5+VUI1 package will not have to write scripts like those shown above; these will be generated by the Script Generator part of the OO user interface.

Table: Comparison of results for three sample problems.

	FORTRAN	OO VUI
Problem 1	0.0017520±0.0000158	0.0017528±0.0000158
Problem 2	0.0017054±0.0000146	0.0017063±0.0000147
Problem 3	0.230075±0.037540	0.230973±0.037547

5 Conclusions

A new version of EGS, EGS5, is being developed that will incorporate numerous physics and programming enhancements. The owner, SLAC, intends to make EGS5 available to the user community as previous versions of EGS were. However, EGS5 will also be made commercially available as a package with an object-oriented user interface in the form of EGS5+VUI1. The package will allow EGS5 to be used without the need to write problem-specific subroutines and compile and link them to the EGS5 physics code. Instead, the user will interact with the code through a visual user interface that will prepare a script that defines the problem to be solved. This script will be interpreted by the parser, which creates the problem objects for the EGS5 physics code at run time from classes stored in a dynamic link library. The OO user interface package will include numerous geometry, scoring, and source classes; however, if the library of classes is insufficient for a given problem, a direct procedure for the user to create new classes that can also be linked at run time is included. The new EGS5+VUI1 will enable users to address challenging radiation transport problems using the enhanced physics of EGS5 without the need to write problem-specific code.

Acknowledgements

This project was supported by the U.S. Department of Energy through Grant DE-FG02-97ER82465.

References

[1] W.R. Nelson, H. Hirayama and D.W.O. Rogers, "The EGS4 Code System", *SLAC-265*, Stanford Linear Accelerator Center, Stanford University, Stanford, CA (1985).

[2] A.J. Cook, "Mortran3 User's Guide", *SLAC Computational Research Group Technical Memorandum Number CGTM 209*, Stanford Linear Accelerator Center, Stanford University, Stanford, CA (1983).

[3] H. Hirayama and Y. Namito, "Lecture Notes of EGS4 Course at KEK", *KEK Internal 99-5*, KEK, Tsukuba, Japan (1999).

[4] Y. Namito, S. Ban, and H. Hirayama, "Implementation of Linearly-polarized Photon Scattering into the EGS4 Code", *Nuc. Instr. Meth.* **A322**(1993)277-283.

[5] A.F. Bielajew and Rogers, "PRESTA: The Parameter Reduced Electron-Step Transport Algorithm for Electron Monte Carlo Transport", *Nucl. Instr. Meth. Phys. Res* **B18**(1987)165-181.

[6] C.-M. Ma and A. E. Nahum, "A new algorithm for EGS4 low-energy electron transport to account for the change in discrete interaction cross-section with energy", *Nucl. Instru. Meth.* **B72**(1992)319-330.

[7] A.F. Bielajew and D. W. O. Rogers, "Photoelectron Angular Distribution in the EGS4 Code System", *National Research Council of Canada Report PIRS-0058* (1986).

[8] A.F. Bielajew, R. Mohan and C.-S. Chui, "Improved Bremsstrahlung Photon Angular Sampling in the EGS4 code system", *Med. Phys.* **17**(1990)522.

[9] M. Conti, A. Del Guerra, D. Mazzei, P. Russo, W. Bencivelli, E. Bartolucci, A. Messineo, V. Rosso, A. Stefanini, U. Bottigli, P. Randaccio and W. R. Nelson, "Use of the EGS4 Monte Carlo Code to Evaluate the Response of HgI_2 and CdTe Detectors for Photons in the Diagnostic Energy Range", *Nucl. Inst. Meth.* **A322**(1992)591-595.

[10] Y. Namito and H. Hirayama, "Improvements of Low Energy Photon Transport for EGS5", *Proc. Second International Workshop on EGS,* Aug. 8-10, 2000, KEK, Tsukuba, Japan (2000).

[11] A.F. Bielajew and S.J. Wilderman, "Innovative Electron Transport Methods in EGS5", *Proc. Second International Workshop on EGS,* Aug. 8-10, 2000, KEK, Tsukuba, Japan (2000).