Changes required in users code along with Geant4 v8.0

Makoto Asai
SLAC/SCCS

Geant4 version 8.0

- Released on December 16th, 2005
- In this release, some design changes and/or enhanced features require some migrations being necessary for some uses in order to upgrade from 7.1 to 8.0.
  - “Non-static” particle definition
  - New multiple scattering process
  - Signature change in G4VProcess
  - Merging “envelope” into G4Region
  - Migration to <sstream>
- We will likely release a patch in a few weeks.
  - Announcement will be posted on our Home Page and sent to Geant4 announcement mailing list.

“Non-static” particle definition

- In v8.0 all particle definition class objects are instantiated dynamically at the moment of actual invocation of G4VUserPhysicsList::ConstructParticle() method.
- This method is invoked at the moment the user’s physics list is instantiated and set to G4RunManager (not at the moment of G4RunManager::Initialize()).
- This new scheme allows the user to set alternative physics quantities, e.g. their masses, of elementary particles before they are instantiated with these quantities hard-coded in Geant4.
- This new scheme causes some restrictions which might affect on the user’s already-existing code.

"Non-static" particle definition Restriction - 1

- Physics processes and models must be instantiated in ConstructProcess() method of the user's physics list. In other words, physics processes or models must not be data members of the physics list.
  - Some physics processes and models require that particles used in these processes/models have already been instantiated.
  - If a process is a data member of the physics list, it is instantiated along with the instantiation of the physics list itself, i.e. before the invocation of ConstructParticle().
  - Most of "educated-guess" physics lists distributed with previous Geant4 releases have had this problem.
- Thus we argue the user to use the new physics lists distributed with this new release 8.0.
  - In case you have your customized physics list modified from old "educated-guess" physics list, you might need to reiterate yours.

“Non-static” particle definition Restriction - 1

- If the user uses his/her own physics list and finds a G4Exception with the following message happens at the instantiation of the physics list, fix the physics list.

Error message :

”Access to G4ParticleTable for finding a particle or equivalent operation occurs before G4VUserPhysicsList is instantiated and assigned to G4RunManager. Such an access is prohibited by Geant4 version 8.0. To fix this problem, please make sure that your main() instantiates G4VUserPhysicsList and set it to G4RunManager before instantiating other user classes such as G4VUserPrimaryParticleGeneratorAction.”

Fix :

All physics processes and models defined as data members of the physics list must be moved to ConstructProcess() method, and must be explicitly instantiated by “new” operator.

"Non-static" particle definition Restriction - 2

- User action classes (derived from G4VUserPrimaryGeneratorAction, G4UserRunAction, G4UserEventAction, G4UserStackingAction, G4UserTrackingAction and G4UserSteppingAction) must be instantiated after the physics list is instantiated and set to G4RunManager.
  - Please note that user's detector construction class is a user initialization class and thus is not affected by this restriction.

"Non-static" particle definition Restriction - 2

- If the user finds a G4Exception with the following message, fix the main().

Error message (for G4VUserPrimaryGeneratorAction) :

"You are instantiating G4VUserPrimaryGeneratorAction BEFORE your G4VUserPhysicsList is instantiated and assigned to G4RunManager. Such an instantiation is prohibited by Geant4 version 8.0. To fix this problem, please make sure that your main() instantiates G4VUserPhysicsList AND set it to G4RunManager before instantiating other user action classes such as G4VUserPrimaryParticleGeneratorAction."

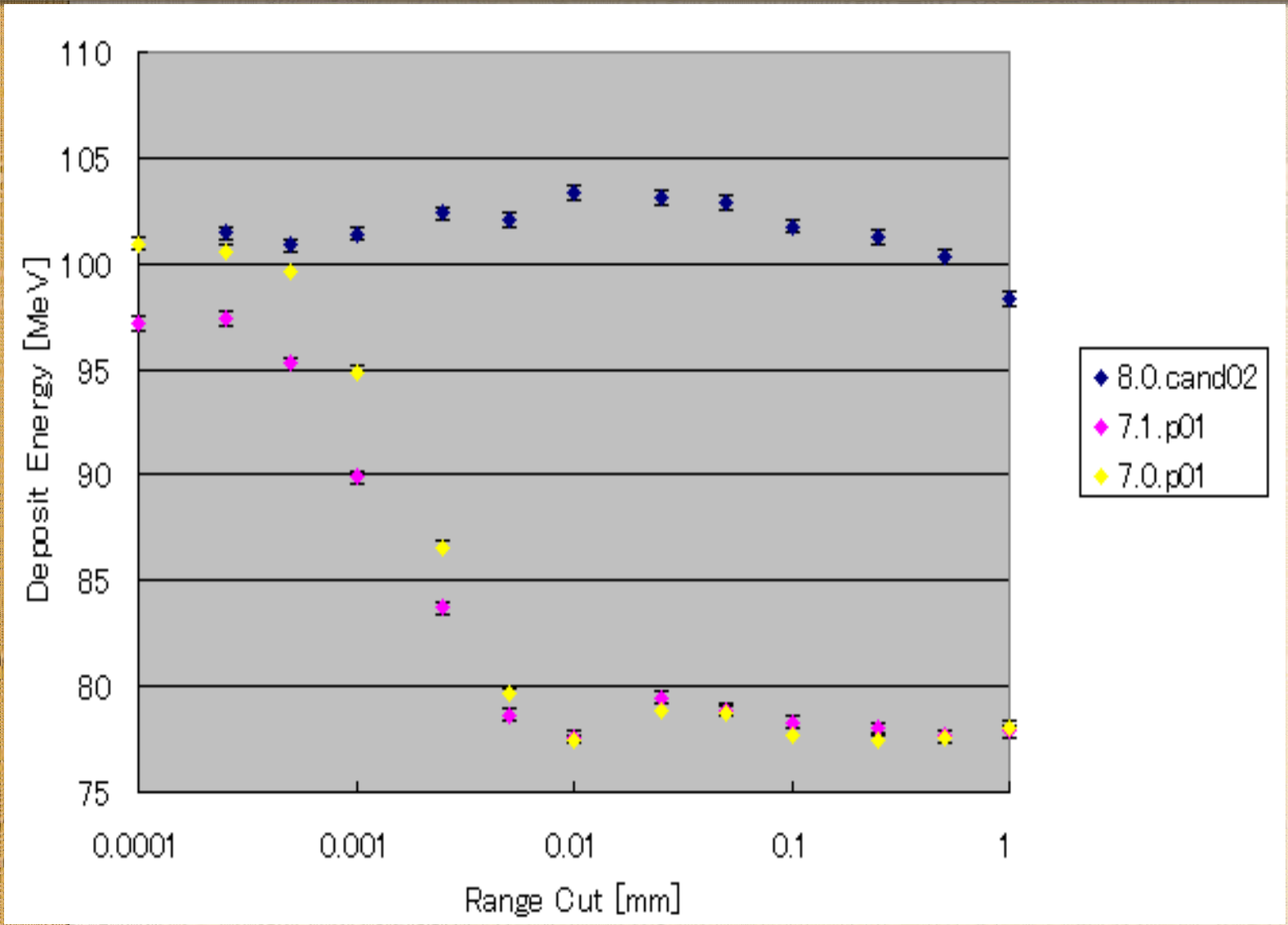
Fix :

Edit your main() to make sure all user action classes are instantiated after your physics list is instantiated and set to G4RunManager.

New multiple scattering process

- In the electromagnetic standard package major changes have been introduced concerning the Multiple Scattering process.
- To improve the behavior of low energy particles (electrons in particular, but affecting also hadrons), the Multiple Scattering now limits the step size for the particles.
  - This restriction is undertaken using several criteria, and is applied systematically, in all volumes and materials. In addition, a model of the correlation between lateral displacement and final direction has been implemented (see the Physics Reference Manual for further information).
- As a result, most physical observables become more stable when varying production cuts (i.e. less "cut dependent").

New multiple scattering process



New multiple scattering process

- There is a corresponding cost, a CPU-time penalty, when utilizing the same value of the production thresholds.
  - This penalty can be significant, depending on the user's setup and the cut values.
  - For many use cases the increased stability will allow the choice of higher production thresholds, recovering computing performance while maintaining physics performance.
- To enable the user to investigate its benefits, a mechanism is provided to deactivate this step limitation.
  - The new method MscStepLimitation(bool) of G4MultipleScattering disables these new step limitations.
- In addition, in order to help the transition, the old version is available for this release 'frozen' in the class G4MultipleScattering71.
  - Several examples (in extended/electromagnetic) provide sample physics lists which use this older version.
  - Please confirm your physics list.

Signature change in G4VProcess

- A signature change is implemented in the G4VProcess base class.
- The virtual method StartTracking() defined in G4VProcess now takes const G4Track* as an argument in its signature.
  - This enhancement enables a process to alternate its initialization for a track with respect to the nature of the track.
  - For example, the shower parameterization process can alternate the granularity of the geometry inside the envelope according to the particle type of the track.
- Users implementing their own physics process and making use of this virtual method will have to modify the signature accordingly.

Merging "envelope" into G4Region

- "Envelope" was a logical volume which has a valid pointer to G4FastSimulationManager object.
- In v8.0, G4FastSimulationManager must be set to G4Region instead of G4LogicalVolume.
  - The concept of "envelope" and "ghost volume" for fast parameterization has been merged with the G4Region in this release. Therefore, a logical volume no longer acts directly as an "envelope", since this attribute now assigned to G4Region.
  - In case you are using a shower parameterization (e.g. Gflash), your geometry description must be fixed accordingly.
  - Please refer to the User Guide for Application Developers for the detailed information. The relevant modules have migrated accordingly. See exampleN05 which has been revised to demonstrate this new mechanism.

Merging "envelope" into G4Region

- A design iteration has taken place for the GFlash fast shower parameterization model and some classes were renamed to follow the new implementation and design for envelopes. User code making use of the GFlash tool should migrate accordingly.
  - TRD models for cluster parameterization have been retired. The functionality is now provided directly as physics processes and included in the electromagnetic packages.

Migration to <sstream>

- <strstream> types are no longer in use by the Geant4 kernel in this release. Also any protection from compilation warnings by the compiler for deprecation of such header are no longer implicitly included.
- User code therefore will be exposed to implement such migration to <sstream>, if not already done.