# Geant4 release 5.1 summary

Gabriele Cosmo
EP/SFT

# Release contents

- Introduction of *region* as concept for defining *production cuts*
  - First introduced in *beta* release of February
  - Modified strategy for handling cuts by physics processes
  - Reviewed initialization mechanism of physics tables
  - Backward-compatibility with previous release and defined migration path for future releases
- Bug fixes

# Brief introduction to cuts & regions - 1

- Cut = production threshold
  - Distance below which no secondary particles are produced (internally converted to energy for each used material)
  - NOT tracking cut
- Geant4 was originally designed to allow a *unique* cut in range
  - **Unique cut in range for each particle**
    - The default is the same cut for all particles
  - **Allows consistency of simulated physics**
    - Guaranties that a volume with high cuts (I.e. with poor physics quality) will not "pollute" the simulation of a neighboring volume with low cuts
- Requirements from ATLAS, BABAR, CMS, LHCb, … to allow several cuts
  - Globally or per particle

# Brief introduction to cuts & regions - 2

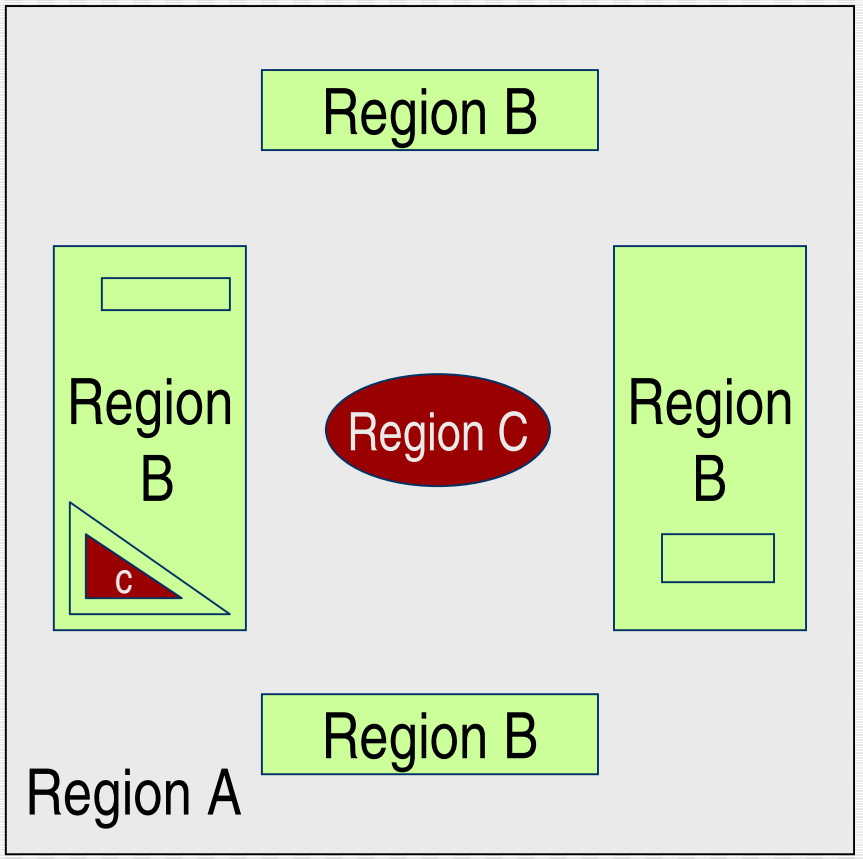
- Motivations for having several cuts
  - A unique cut may be source of run-time performance penalties
    - Detector's parts with lower cuts fix the cut for the whole simulation
    - A cut can be far too low than necessary in some detector's parts
  - A cut is typically required at the level of a detector subsystem

# Brief introduction to cuts & regions - 3

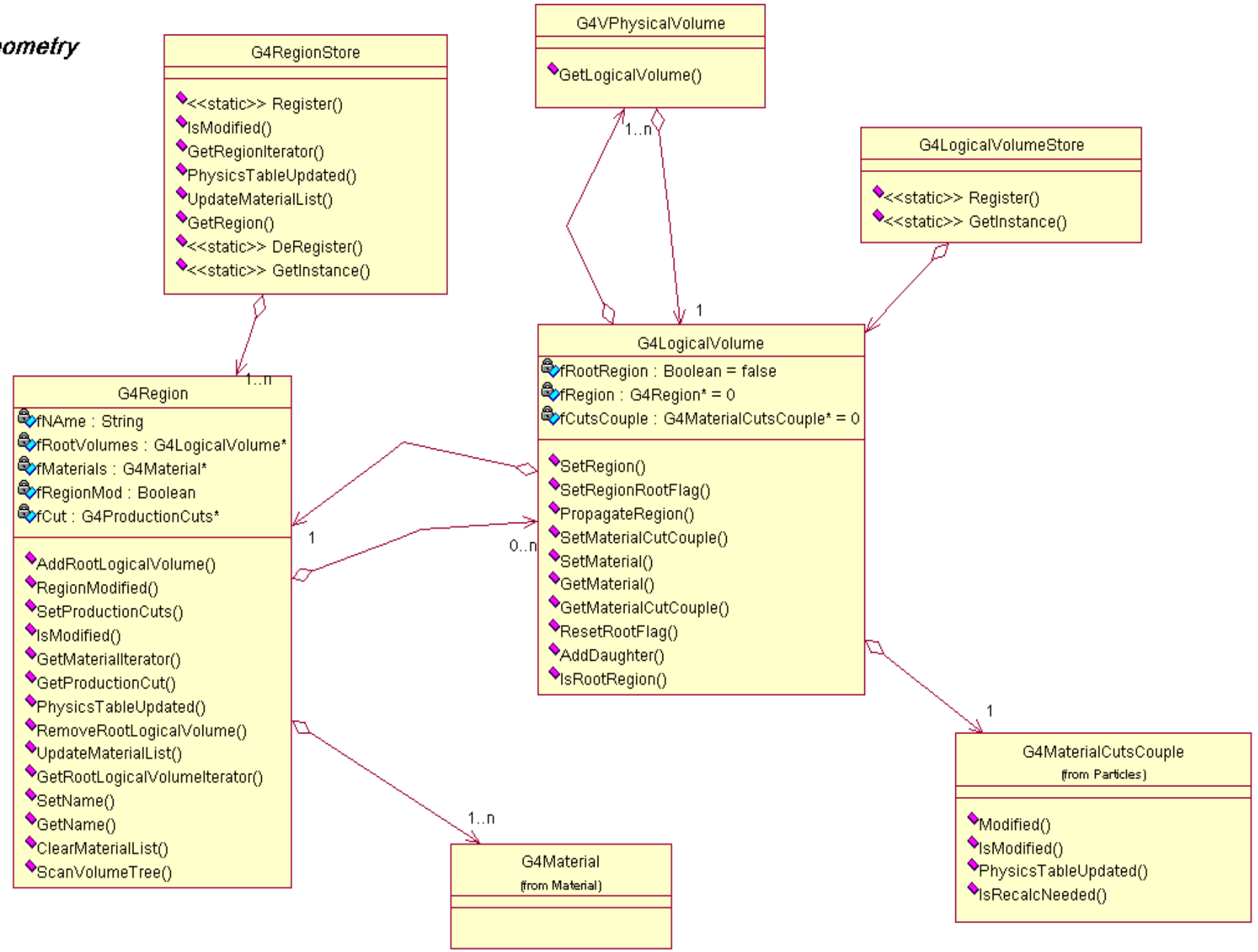
- Introduce the concept of *G4Region*
  - Large geometrical area, typically the root logical volume of a subsystem
    - Can also be a group of root logical volumes (e.g. barrel + end-caps of the calorimeter)
  - A cut in range *G4ProductionCut* is associated to a region. A default one is defined
    - Eventually, a range cut per particle is allowed
  - One or more root logical volumes can be added to a region

# Brief introduction to cuts & regions - 4

- A region can have several root volumes
- The world is in a predefined *default* region



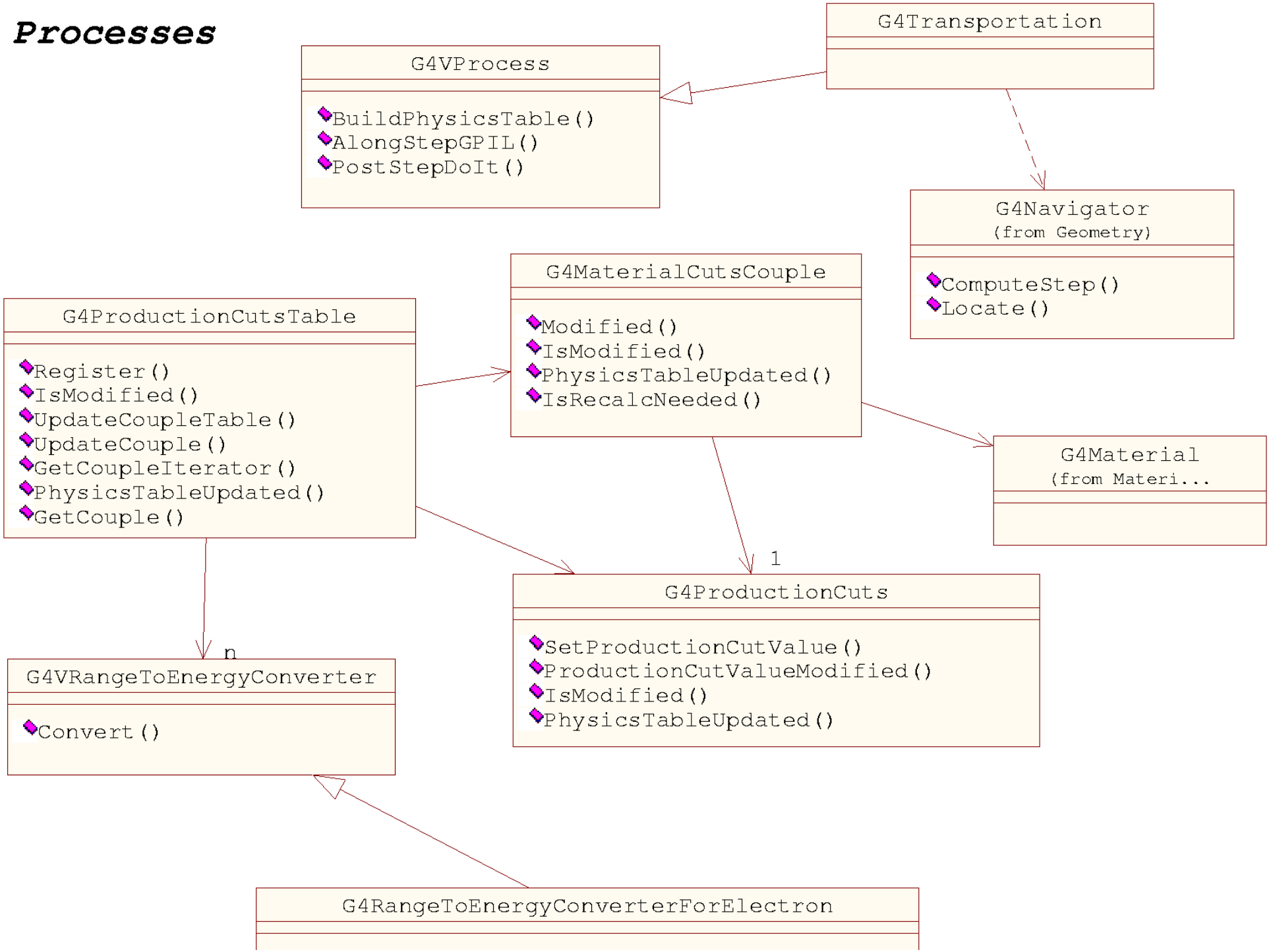
***Geometry***


G4RegionStore
<<static>> Register()
IsModified()
GetRegionIterator()
PhysicsTableUpdated()
UpdateMaterialList()
GetRegion()
<<static>> DeRegister()
<<static>> GetInstance()
G4VPhysicalVolume
GetLogicalVolume()
1..n
1
G4LogicalVolumeStore
<<static>> Register()
<<static>> GetInstance()
G4LogicalVolume
fRootRegion : Boolean = false
fRegion : G4Region* = 0
fCutsCouple : G4MaterialCutsCouple* = 0
SetRegion()
SetRegionRootFlag()
PropagateRegion()
SetMaterialCutCouple()
SetMaterial()
GetMaterial()
GetMaterialCutCouple()
ResetRootFlag()
AddDaughter()
IsRootRegion()
1..n
G4Region
fNAme : String
fRootVolumes : G4LogicalVolume*
fMaterials : G4Material*
fRegionMod : Boolean
fCut : G4ProductionCuts*
AddRootLogicalVolume()
RegionModified()
SetProductionCuts()
IsModified()
GetMaterialIterator()
GetProductionCut()
PhysicsTableUpdated()
RemoveRootLogicalVolume()
UpdateMaterialList()
GetRootLogicalVolumeIterator()
SetName()
GetName()
ClearMaterialList()
ScanVolumeTree()
1
0..n
1..n
G4Material
(from Material)
1
G4MaterialCutsCouple
(from Particles)
Modified()
IsModified()
PhysicsTableUpdated()
IsRecalcNeeded()

# Physics processes & cuts

- Only processes dealing with cuts are affected
- Main issue is to know which cross-section table to use in the current volume at tracking time
- In the old scheme, for a given process, there was a one-to-one relation between a material and a cross-section table:
  - This was used to retrieve the physics table using:
  - *« index of material » == « index of physics table »*
- Now, since the same material may appear in several regions above relation is replaced by:
  - *« index of {material, region} couple » == « index of physics table »*

# Processes


G4Transportation
G4VProcess
BuildPhysicsTable()
AlongStepGPIL()
PostStepDoIt()
G4Navigator
(from Geometry)
ComputeStep()
Locate()
G4MaterialCutsCouple
Modified()
IsModified()
PhysicsTableUpdated()
IsRecalcNeeded()
G4ProductionCutsTable
Register()
IsModified()
UpdateCoupleTable()
UpdateCouple()
GetCoupleIterator()
PhysicsTableUpdated()
GetCouple()
G4Material
(from Materi...
1
n
G4ProductionCuts
SetProductionCutValue()
ProductionCutValueModified()
IsModified()
PhysicsTableUpdated()
G4VRangeToEnergyConverter
Convert()
G4RangeToEnergyConverterForElectron

# Some features and known limitations

- *G4Region* implemented for use in cuts
  - But could be of more general usage in future
- Above mechanism requires a ***partition*** of the logical volumes
  - A same logical volume cannot belong to two different regions
  - Understood as being a (very) weak limitation in practice
- The propagation of the region's attributes is triggered at initialization time and whenever a change to the detector-description setup is applied
- *G4ProductionCut* defines a set of cuts for all particles
  - The same cut value appears for, say e-, in two different cut objects and same materials appear in the related regions
    - the relevant cross-section table entries will be calculated twice
  - Cross-section tables will be built only for *used* materials

# More features …

- The calculation of the cross-section tables takes place after the *BeamOn* command is issued.
  - The initialization of the run-manager is noticeably faster
  - On the other hand, more time is required to get the first event loop starting
- Production thresholds are now adopted only for *gammas*, *electrons* and *positrons*
  - Other particles do not use the concept of production threshold

# Items for migration of the user code to release 5.1 - 1

- User applications having user-defined production cuts will work with no required transition to the new interface, as long as regions are not used
- The new interface for setting production cuts will be required for using the 'cuts-by-region' feature (see the guides)
- The old interface for setting/controlling production cuts (methods and UI commands) will be made obsolete in the next public release and later-on removed
- The file format of storing physics tables has been changed. Thus all the stored physics tables must be rebuilt

# Items for migration of the user code to release 5.1 - 2

- Advanced applications making use of a sub-classed run-manager will have to migrate to the new initialization scheme and interface defined in *G4RunManager*
  - Similarly for user-defined physics processes
- Particularly, users must be aware of the change in the sequence of building the physics tables
  - *G4RunManager::BuildPhysicsTable()* must be invoked from *G4RunManager::RunInitialization()*
  - *G4VUserPhysicsList::SetCuts()* does not trigger the construction of the physics tables, it simply sets the production thresholds (cuts)

# What next ?

- Release 5.2, end of June
  - *Consolidation* release
    - Performance optimisation
    - Some minor developments
    - Bug fixes, code cleanup
- Release 6.0, December
  - New features
  - See:

    http://cern.ch/geant4/source/planned_features.html