Version 10.7

Particles and Processes in Geant4

Vladimir Ivantchenko

CERN, Geneva, Switzerland & Tomsk State University, Russia

Geant4 Advanced Course

4 October 2021

Outline

- Geant4 basic interfaces to physics
- Geant4 particles
- Geant4 processes
- Physics Lists
- Ions and exotic particles
- Geant4 cuts
- User interface to physics

Geant4 basic interface to physics

- The interface of Geant4 kernel to physics is abstract
- Base physics abstract classes are following:
  - The **`G4ParticleDefinition`** objects shared between threads
  - The **`G4VProcess`** thread local objects
  - The **`G4ProcessManager`** thread local interface class
- Configuration of physics is prepared in the G4VUserPhysicsList mandatory user class
- These interfaces are stable for ~20 years allowing users to work with different Geant4 versions and providing a basis for new developments
  - Concrete physics is implemented in physics models and cross section classes
  - Alternative models and cross sections are provided in Geant4 libraries
  - A user may be also a developer of a custom particle, process, physics model, or cross section

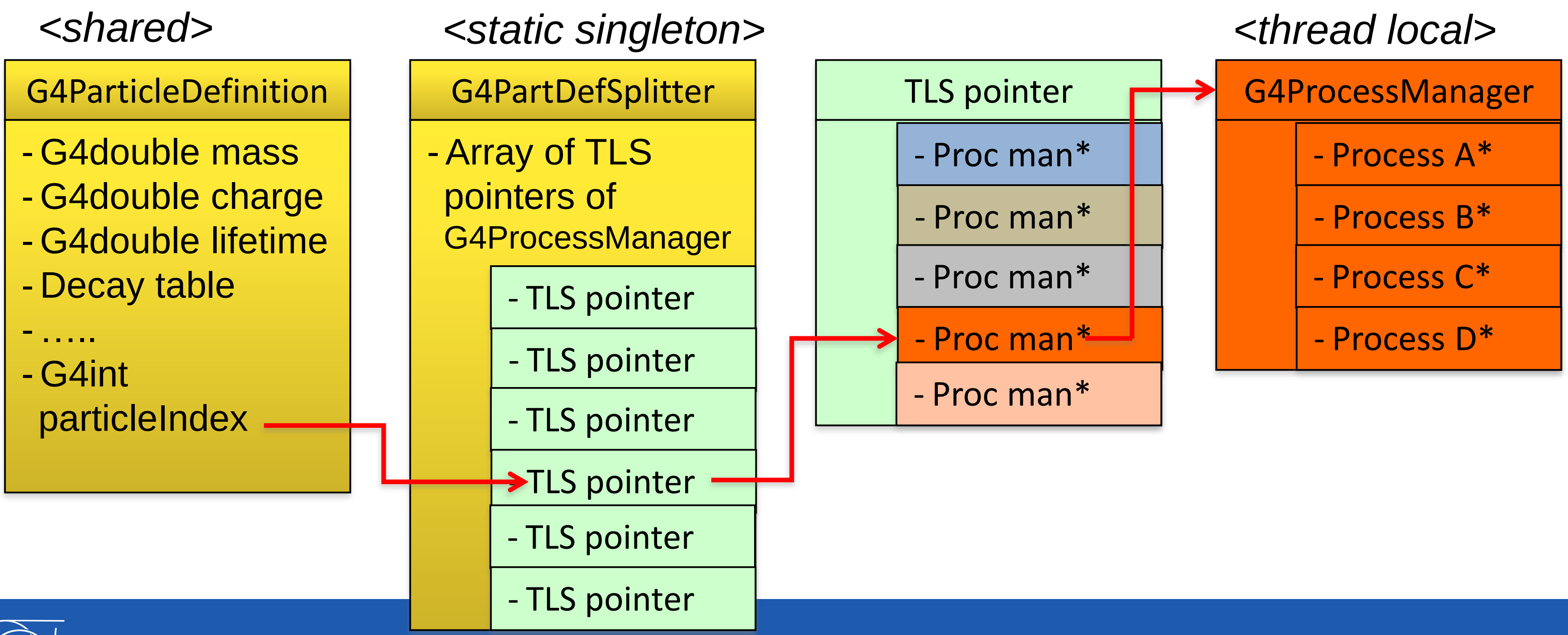
GEANT4 PARTICLES

Geant4 particles

- G4ParticleDefinition is the main object keeping static information about particles
  - Name, mass, charge, quantum numbers, decay table….
- "Stable" particles
  - Leptons: $e^{\pm}$, $\mu^{\pm}$, ….
  - Bosons: G4Gamma, G4OpticalPhoton, ….
  - Geantino is a particle without any interaction
  - "Stable" hadrons: $\pi^{\pm}$, $K^{\pm}$, ….
  - Light ions: d, t, ^{3}He, ^{4}He
  - G4GenericIon is used to define physics for all other ions
- "ShortLived" hadrons normally do not tracked by Geant4 but used internally by hadronic models
  - Quarks, di-quarks, ρ(770), ω(783)…

Split class – case of particle definition

- In Geant4, each particle type has its own dedicated object of G4ParticleDefinition class.
  - Static quantities : mass, charge, lifetime, decay channels, etc.,
    - Are shared by all threads
  - Dedicated object of G4ProcessManager : list of physics processes which this particle undertakes.
    - Physics process object must be thread-local
    - Thread local storage is used (TLS)

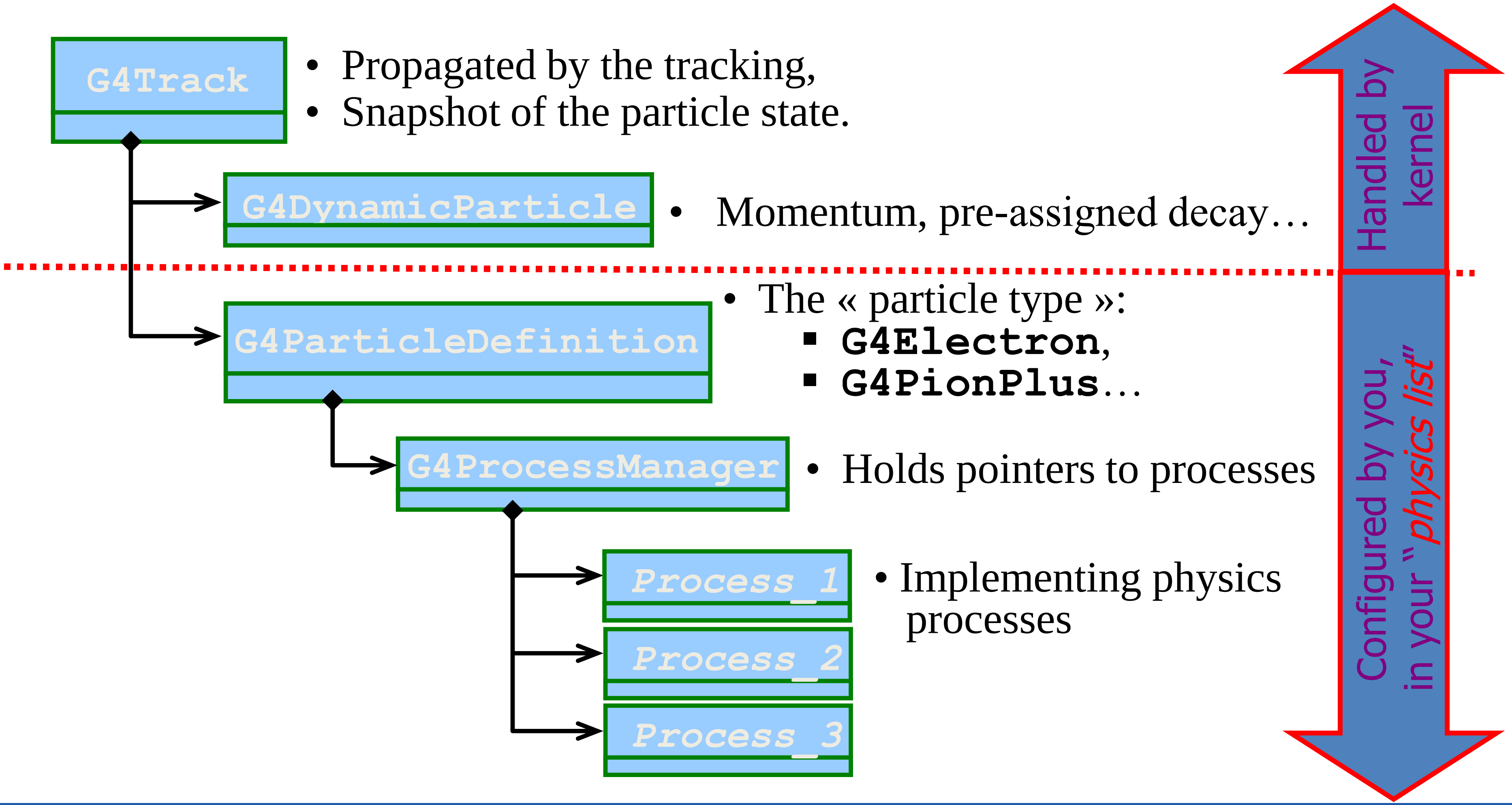


GEANT4 PROCESSES

Geant4 process

- Processes are classified as:
  - Electromagnetic
  - Hadronic
  - Decay
  - Parameterized
  - Transportation
  - ……..
- Any process has process has type and sub-type
  - const G4String& G4VProcess::GetProcessType();
  - G4int G4VProcess::GetProcessSubType();
    - This method is recommended to be used for MC truth
    - The list of sub-types is stable since introduced and only updated with new processes
- Any process may be initialized using virtual methods:
  - **G4bool IsApplicable(const G4ParticleDefinition &);**
    - Used to check if a process can handle the given particle type
  - **void PreparePhysicsTable**(const G4ParticleDefinition&);
  - **void BuildPhysicsTable**(const G4ParticleDefinition&);
    - Used for initialization of internal data of the process before run

From G4Track to processes



- Propagated by the tracking,
- Snapshot of the particle state.
- Momentum, pre-assigned decay…
- The « particle type »:
  - **G4Electron**,
  - **G4PionPlus**…
- Holds pointers to processes
- Implementing physics processes

Geant4 Physics: Electromagnetic

- The standard EM part: provides a complete set of EM interactions (processes) of charged particles and gammas from 1 keV to ~PeV
  - used practically in all kind of Geant4 applications
- The low energy EM part: includes special treatments for low energy e-/+, gammas and charged hadrons:
  - more sophisticated approximations valid down to lower energies e.g. more atomic shell structure details
  - some of these models will be valid down to ~10 eV but cannot be used above upper limits, which vary from 1 MeV to few GeV
- Optical photons: interactions special only for long wavelength photons
  - processes for reflection/refraction, absorption, wavelength shifting, (special) Rayleigh scattering
  - G4OpticalPhoton is the particle type
- Phonon physics is also implemented within Geant4

Geant4 Physics: Hadronic

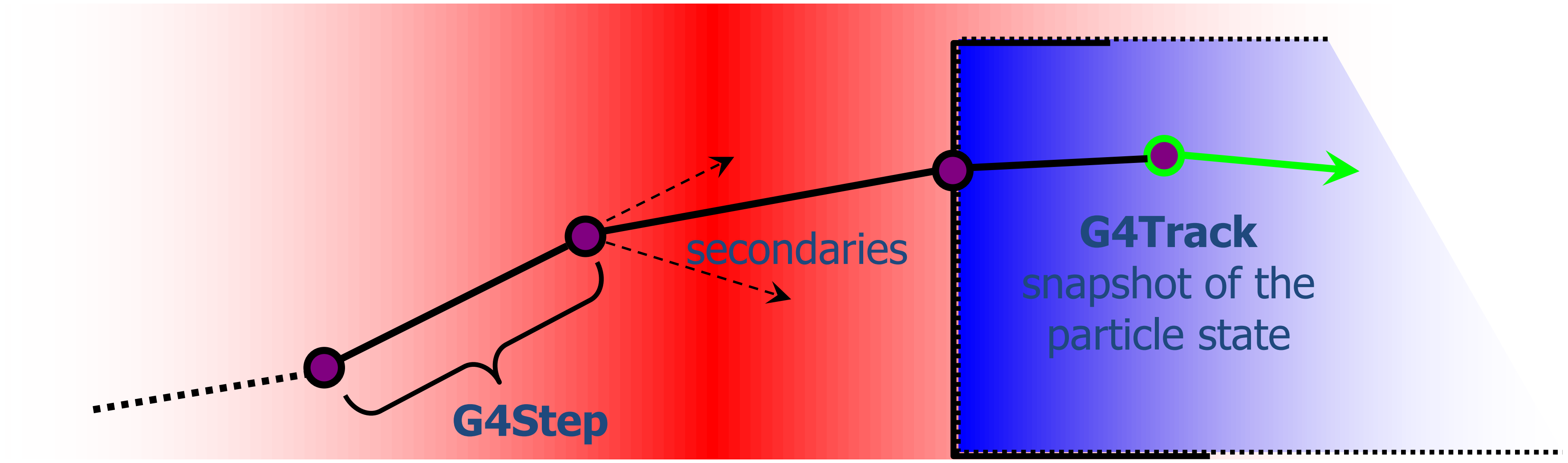
- Pure hadronic interactions for 0 to 100 TeV
  - elastic, inelastic, capture, fission
- Radioactive decay:
  - both at-rest and in-flight
- Photo-nuclear interaction from ~1 MeV up to 100 TeV
- Lepto-nuclear interaction from ~100 MeV up to 100 TeV
  - e+ and e- induced nuclear reactions
  - muon induced nuclear reactions
- Recently introduced processes of neutrino-nuclear interactions

Geant4 Physics: Decay, Parameterized and Transportation

- Decay processes includes:
  - weak decay (leptonic, semi-leptonic decay, radioactive decay of nuclei)
  - electromagnetic decay (π^0, Σ^0, etc.)
  - strong decay not included by default
    - they are part of hadronic models
    - may be assigned by a user to a particle
- Parameterized process:
  - assigned to G4LogicalVolume
  - instead of step-by-step simulation provides hits in the logical volume and list of particles living the volume
  - for example, EM shower generation in a calorimeter based on parameters obtained from detailed simulation of the calorimeter response
- Transportation process:
  - responsible for propagating a particle through the geometry in electromagnetic or gravitational field
  - needs to be assigned to each “stable” particle

Geant4 tracking

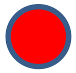
- **`G4Track`** is the object "pushed" step by step by the tracking :

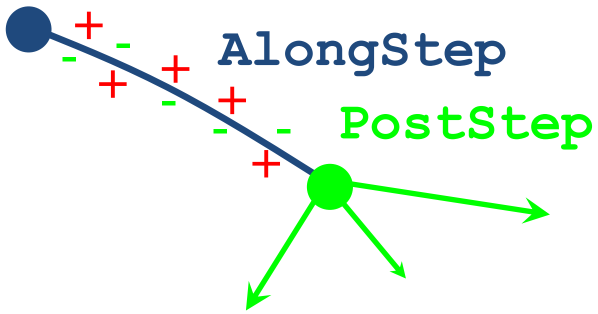


- Moving by one step is the responsibility of the "stepping"
  - Which is the core engine of the "tracking" machinery
- These moves/steps are defined by physics or by geometry
  - Step length limit is a result of competition of processes
  - Processes involved at a step may change the **`G4Track`**
  - G4Transportation stops track at the volume boundary

G4VProcess: 3 kind of actions

- **G4VProcess** is an abstract class defining the common interface of all processes in Geant4:
  - Used by all processes including G4Transportation
  - Defined in `source/processes/management`
- Three kinds of actions:
  - **`AtRest`** actions:
    - Decay, e^+ annihilation …
  - **`AlongStep`** actions:
    - To describe continuous (inter)actions, occurring along the path of the particle, like ionisation;
  - **`PostStep`** actions:
    - For describing point-like (inter)actions, like decay in flight





G4VProcess : actions summary

- The virtual «action» methods are following:
  - `AtRestGetPhysicalInteractionLength(), AtRestDoIt();`
  - `AlongStepGetPhysicalInteractionLength(), AlongStepDoIt();`
  - `PostStepGetPhysicalInteractionLength(), PostStepDoIt();`
- Optional run time virtual methods:
  - **StartTracking(G4Track*);**
    - Allowing the process preparation for a new G4Track
  - **EndTracking();**
    - End of given G4Track

G4VProcess: extensions

- A process can implement any combination of the three **`AtRest`**, **`AlongStep`**, and **`PostStep`** actions:
  - for example, decay = **`AtRest`** + **`PostStep`**

- If you plan to implement your own process:
  - A set on intermediate classes exist implementing various combinations of actions, for example:
    - **`G4VDiscreteProcess`**: only **`PostStep`** actions
    - **`G4VContinuousDiscreteProcess`**: **`AlongStep + PostStep`** actions

G4ProcessManager

- It is a Geant4 kernel class
  - A user should not change it
- **`G4ProcessManager`** maintains three vectors of actions :
  - One for the **`AtRest`** methods of the particle;
  - One for the **`AlongStep`** ones;
  - And one for the **`PostStep`** actions.
- Note, that the ordering of processes provided by/to the **`G4ProcessManager`** vectors is relevant and used by the stepping
  - There are few critical points you should be aware of
    - Multiple scattering can shift end point of a step
    - Scintillation, Cerenkov and some other processes assuming that a step and energy deposition at the step are defined

PHYSICS LISTS

Physics Lists

- Physics List is an object that is responsible to:
  - specify all the particles that will be used in the simulation application
  - together with the list of physics processes assigned to each individual particles
- One out of the 3 mandatory objects that the user needs to provide to the G4RunManager in case of all Geant4 applications:
  - it provides the information when, how and what set of physics needs to be invoked
- Provides a very flexible way to set up the physics environment:
  - the user can chose and specify the particles that they want to be used
  - the user can chose the physics (processes) to assign to each particle
- Geant4 distribution includes the “physics_list” sub-library with many components and many predefined Physics Lists

Modular Physics Lists

- Current recommendation to use Physics List via inheritance from G4VModularPhysicsList which derives from G4VUserPhysicsList
- Main public methods:
  - **G4VModularPhysicsList::RegisterPhysics**(G4VPhysicsConstructor*)
    - Addition of physics constructor
  - **G4VModularPhysicsList::ReplacePhysics**(G4VPhysicsConstructor*)
    - Replacement of the same type of physics constructor
- Constructor types:
  - Electromagnetic, EM extra (lepton-nuclear)
  - Decay, Radioactive Decay
  - Hadron elastic, hadron inelastic
  - Ion elastic and inelastic
  - Stopping of negatively charged particles
  - Step limiters (tracking cuts)
  - Optical
  - User may add custom constructor
- Physics List and its components are unique objects, which called in each thread two methods
  - G4VPhysicsConstructor::ConstructParticle()
  - G4VPhysicsConstructor::ConstructProcess()
  - Only const class members are allowed

Cuts definition

- In past cuts were defined in SetCuts() method of physics list
  - After migration to the MT mode, we recommend not doing this
  - Cuts may be defined via UI commands
  - Details on Geant4 cuts will be described below
- Using UI interface Geant4 kernel change cuts and try to count number of steps in the same run
  - /run/setCut 0.01 mm
  - /run/beamOn 100
- Define cuts only for electrons
  - /run/setCutForAGivenParticle e- 10 um
  - /run/setCutForRegion GasDetector 0.1 mm
  - /run/dumpCouples
- How to change low-energy limit of production threshold
  - /cuts/setLowEdge 0.1 keV

Instantiation and ownership of physics objects

- G4PhysicsListHelper provides correct ordering for all processes from Geant4 libraries
  - **G4PhysicsListHelper*** helper = **G4PhysicsListHelper::GetPhysicsListHelper();**
  - helper->**RegisterProcess**(G4VProcess*, G4ParticleDefinition*);
- Custom process should be instantiated with defined ordering
  - **G4ParticleDefinition*** particle;
  - **G4ProcessManager*** man = particle->**GetProcessManager();**
  - man->**AddDiscreteProcess**(G4VDiscreteProcess*); // added to the end
  - man->**AddProcess**(G4VProcess*, idxAtRest, idxAlongStep, idxPostStep);
- Ownership of classes is not belonging to the Physics List class
  - **G4ParticleDefinition** classes are static shared between threads
  - **G4VProcess** classes are registered in process thread local store
  - Model classes for EM and hadronic physics are also registered in thread local store
  - Hadronic cross sections are registered in another thread local store
  - All registrations are done automatically
- All processes, models, and cross section classes should be instantiated via "new"
  - Should not be included by object in any class

IONS AND EXOTIC PARTICLES

Geant4 Approach for Ions

- Light ions are individual Geant4 particles:
  - **G4Deuteron**
  - **G4Triton**
  - **G4He3**
  - **G4Alpha**
- Generic ion serves all other ions:
  - **G4GenericIon** - only one particle
  - Not a real particle (charge = +1, mass = Mp)
  - Serving for any kind of ion with Z>2
  - All concrete ions peak up processes and cross sections of the G4GenericIon
- Ion names
  - "**C12**" means that the carbon ion is in the ground state
  - "**Co60**[58.590]" is the first excitation state of Co60
  - Extra information about atomic shell may be filled to any ion

Exotic particles

- Not discovered particles are not part of Geant4 particle library
- To search exotics users should introduce non-existing particles in the user code
  - Such particles should be instantiated in **ConstructParticle**() method of one of custom G4VPhysicsConstructor, which is user responsibility
  - User should take care attaching processes to exotic particles in **ConstractProcess**() method
- Geant4 offers two extended examples
  - $G4INSTALL/examples/extended/exoticphysics/monopole
  - $G4INSTALL/examples/extended/exoticphysics/dmparticle
  - These examples demonstrate different variants of addition of extra particles and interactions
- In the monopole example additional classes are available for tracking of the magnetic monopole in magnetic field
  - **G4MonopoleTransportation**, **G4MonopoleEquation**

GEANT4 CUTS

How tracks are created and killed

- G4Track can be created
  - By **G4VUserPrimaryGeneratorAction**
  - By any **G4VProcess**
- Geant4 particle is tracked until it is killed by one of Geant4 processes:
  - Transport out of the world volume
  - Inelastic interaction
  - Decay
  - **G4NeutronKiller** or **G4UserLimits**
  - If during tracking kinetic energy become zero and there is no processes **AtRest** the particle is killed by the stepping manager
- Any particle may be also killed by user action classes
- Geant4 introduced conception of "**cut in range**"
  - Physically this means required spatial accuracy of simulation
  - At initialization for each material a production threshold for kinetic energy of secondary particles is computed
    - This means different production thresholds for different materials
  - This is the main difference between Geant4 and other simulation tools, which implement only tracking cuts

Cut and production thresholds for energy loss processes

- User defines cut in range expressed in units of length
- Using this range Geant4 kernel compute production threshold T_{cut} for each material during initialization
- For a typical process (G4hIonisation, G4eIonisation, …), the production threshold T_{cut} subdivides the continuous and discrete parts of energy loss:
  - Mean rate of energy lost due to soft energy transfers
  - Cross section for discrete δ-electron production
  - above T_{cut}
  - *Both energy loss and cross sections are restricted*

$$\frac{dE(E,T_{cut})}{dx} = n_{at} \int_0^{T_{cut}} T \frac{d\sigma(Z,E,T)}{dT} dT$$

$$\sigma(Z,E,T_{cut}) = \int_{T_{cut}}^{T_{\max}} \frac{d\sigma(Z,E,T)}{dT} dT$$

- At each step energy deposition is sampled by a fluctuation model using the computed mean energy loss
- Optionally, energy loss may be modified :
  - for the sampling of fluorescence and Auger–electrons emission
- 4-momentum balance is provided in all cases

What particles have cut in range?

- Cuts in range are defined for
  - Gamma
  - Electron
  - Positron
  - Proton
- Cut for proton is used for all hadrons and ions by elastic scattering processes
- By default, cut in range is defined globally
  - It is possible to have different cut in range for particle type
  - It is possible to define specific cut in range per G4Region

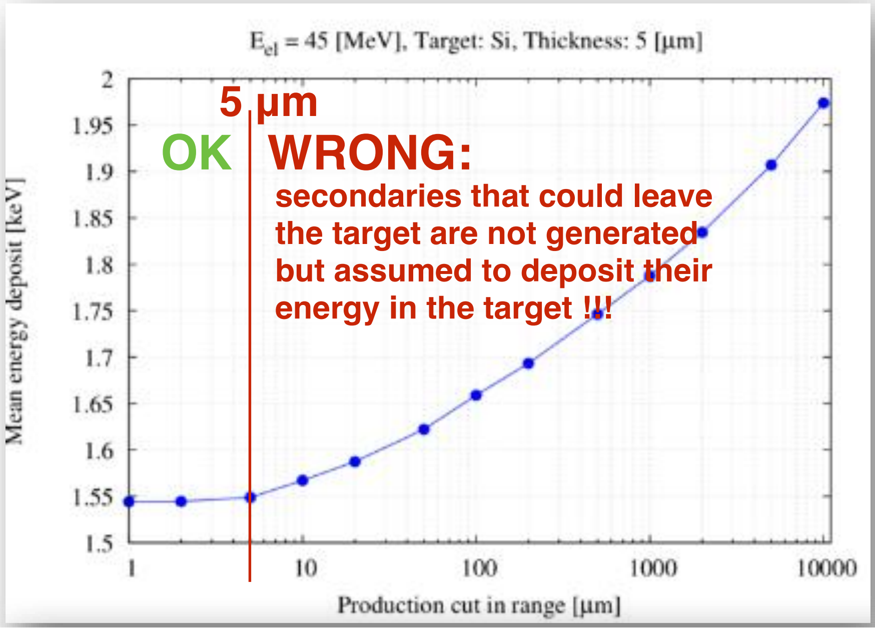
Which processes use cut in range ?

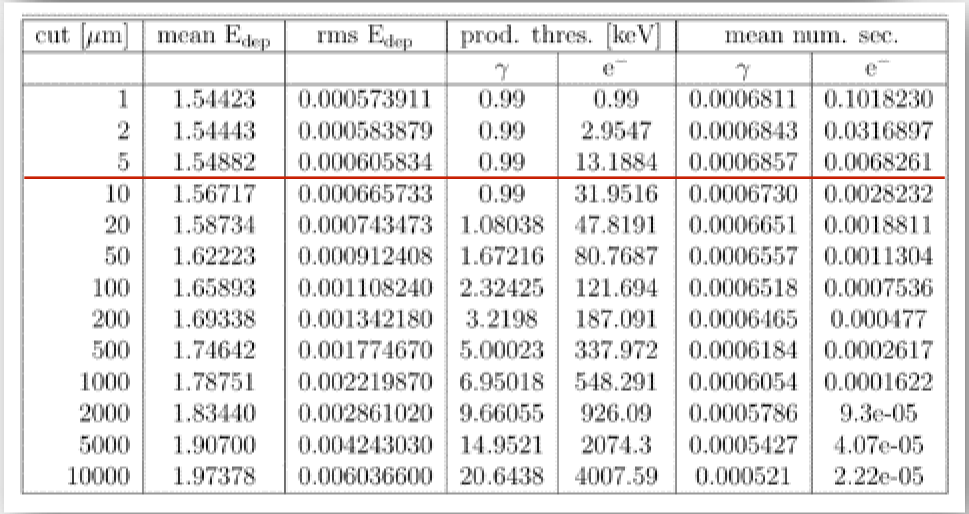
- It is not mandatory to use cuts
  - They are needed to secure CPU performance of simulation
- Energy thresholds (derived from cut in range) are used
  - for gamma are used in Bremsstrahlung
  - for electrons are used in ionisation and e+e- pair production processes
  - for positrons is used in the e+e- pair production process
  - for gamma and electrons are used optionally (“ApplyCuts” options) in some discrete processes
    - Photoelectric effect, Compton, gamma conversion
- Production threshold for gamma and e^{+-} obtained from range cut cannot be whatever
  - The default low energy limit is 1 keV
  - May be changed via UI command:
    - /cuts/setLowEdge 100 keV
- Energy threshold for protons are used to define the threshold for kinetic energy of a nuclear recoil
  - EM single scattering process
  - Hadron elastic scattering

Tracking cuts

- Additionally, to cut in range it is possible to use various tracking cuts
  - Unwanted particles may be killed after the step if corresponded flag is proposed
- In the default physics configurations two types of tracking cuts are applied:
  - Low-energy thresholds for charged particles by ionization 1 keV
  - Time cut for neutron transport 10000 ns
- Tracking cuts values are customizable and can be changed via UI commands
  - Energy cut for protons may be set to zero
- User may easily setup extra tracking cut or step limiter
  - The best is to add extra G4VProcess
  - **G4NeutronKiller** is the example

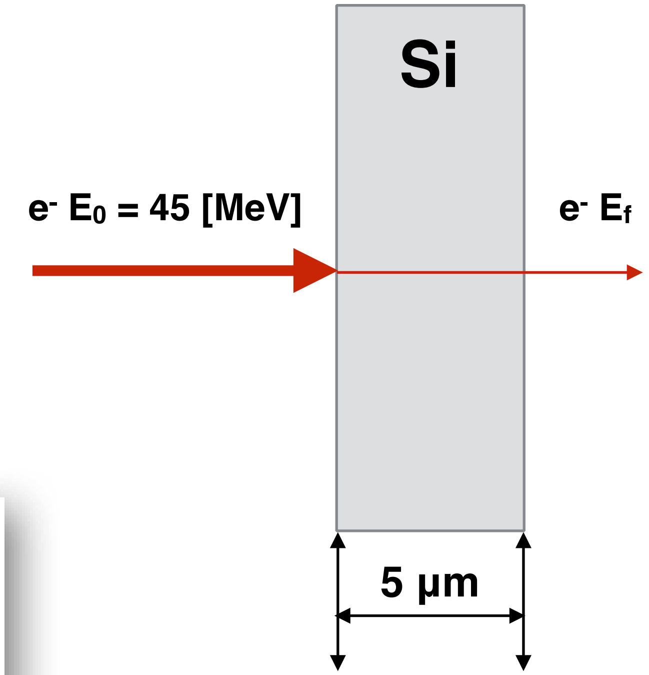
Example demonstrating importance of cuts





cut [μm]	mean E_{dep}	rms E_{dep}	prod. thres. [keV]		mean num. sec.	
			γ	e^-	γ	e^-
1	1.54423	0.000573911	0.99	0.99	0.0006811	0.1018230
2	1.54443	0.000583879	0.99	2.9547	0.0006843	0.0316897
5	1.54882	0.000605834	0.99	13.1884	0.0006857	0.0068261
10	1.56717	0.000665733	0.99	31.9516	0.0006730	0.0028232
20	1.58734	0.000743473	1.08038	47.8191	0.0006651	0.0018811
50	1.62223	0.000912408	1.67216	80.7687	0.0006557	0.0011304
100	1.65893	0.001108240	2.32425	121.694	0.0006518	0.0007536
200	1.69338	0.001342180	3.2198	187.091	0.0006465	0.000477
500	1.74642	0.001774670	5.00023	337.972	0.0006184	0.0002617
1000	1.78751	0.002219870	6.95018	548.291	0.0006054	0.0001622
2000	1.83440	0.002861020	9.66055	926.09	0.0005786	9.3e-05
5000	1.90700	0.004243030	14.9521	2074.3	0.0005427	4.07e-05
10000	1.97378	0.006036600	20.6438	4007.59	0.000521	2.22e-05

Compute the mean of the energy deposit in the target: E_0 - primary, E_f - final energy



USER INTERFACE TO PHYSICS

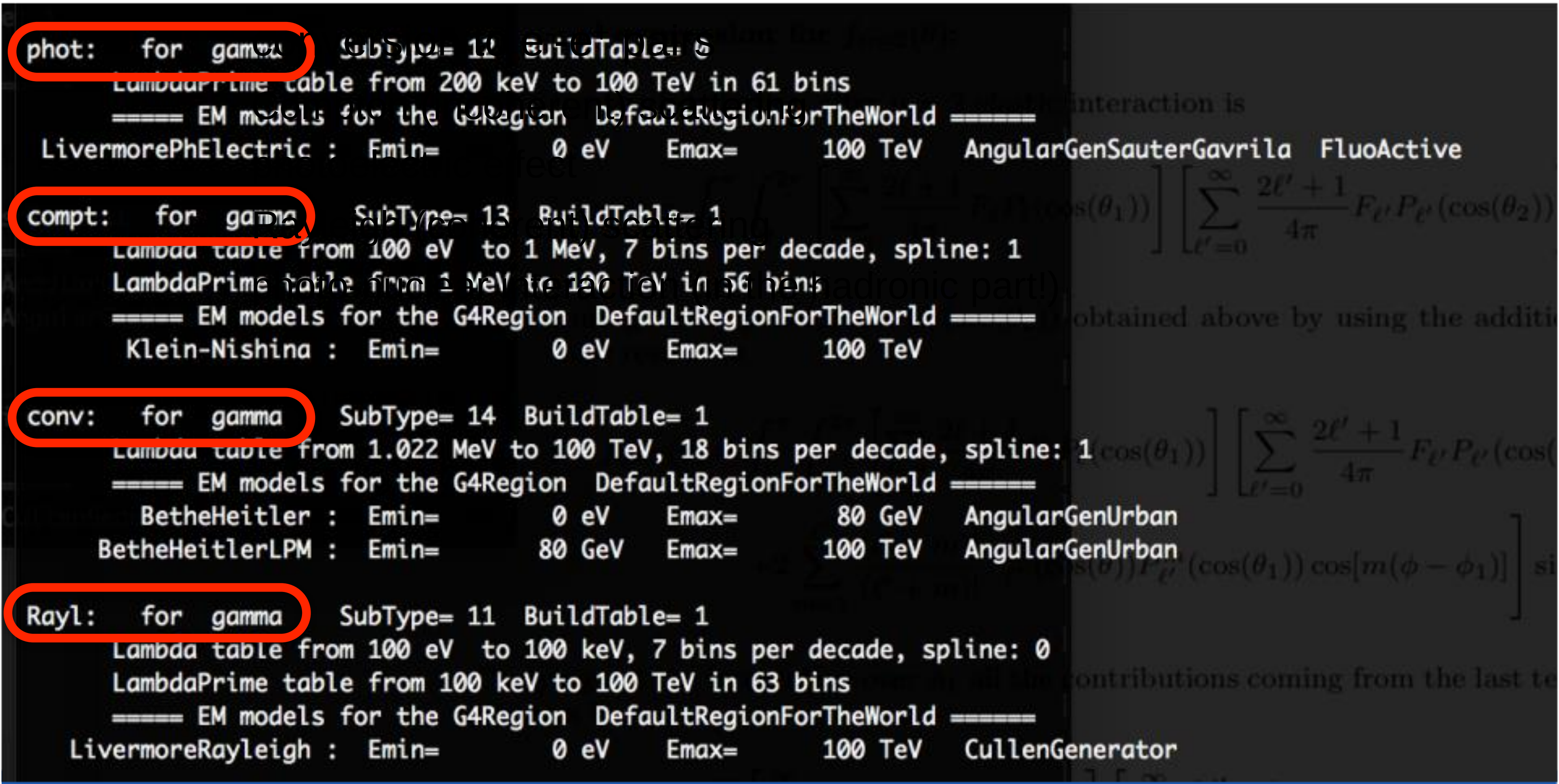
Displaying processes and particles

- When your application has started and when the run manager has been initialized, you can:

- Check the physics processes attached and their ordering:
  - `/particle/select e-`
  - `/particle/processes/dump`
- Check what particles exist:
  - `/particle/list`
- Check a particle property:
  - `/particle/select e-`
  - `/particle/property/dump`
- Please type "help" to get the full set of commands for particle category

Gamma EM processes in the standard output

- Geant4 standard EM interactions for:
  - photon (γ) interactions (example):



```
phot:    for  gamma    SubType= 12  BuildTable= 0
      LambdaPrime table from 200 keV to 100 TeV in 61 bins
      ===== EM models for the G4Region  DefaultRegionForTheWorld ======
 LivermorePhElectric :  Emin=        0 eV     Emax=      100 TeV   AngularGenSauterGavrila  FluoActive

compt:   for  gamma    SubType= 13  BuildTable= 1
      Lambda table from 100 eV  to 1 MeV, 7 bins per decade, spline: 1
      LambdaPrime table from 1 MeV to 100 TeV in 56 bins
      ===== EM models for the G4Region  DefaultRegionForTheWorld ======
        Klein-Nishina :  Emin=        0 eV     Emax=      100 TeV

conv:    for  gamma    SubType= 14  BuildTable= 1
      Lambda table from 1.022 MeV to 100 TeV, 18 bins per decade, spline: 1
      ===== EM models for the G4Region  DefaultRegionForTheWorld ======
         BetheHeitler :  Emin=        0 eV     Emax=       80 GeV   AngularGenUrban
      BetheHeitlerLPM :  Emin=       80 GeV    Emax=      100 TeV   AngularGenUrban

Rayl:    for  gamma    SubType= 11  BuildTable= 1
      Lambda table from 100 eV  to 100 keV, 7 bins per decade, spline: 0
      LambdaPrime table from 100 keV to 100 TeV in 63 bins
      ===== EM models for the G4Region  DefaultRegionForTheWorld ======
    LivermoreRayleigh :  Emin=        0 eV     Emax=      100 TeV   CullenGenerator
```

THANK YOU