CERN, 18 February 2010

Geant4 Physics Lists

Alberto Ribon

CERN PH/SFT

Outline

- Introduction
  - What is a physics list and why we need one?
- The **G4VUserPhysicsList** class
  - The simplest approach
- Modular physics lists: **G4VModularPhysicsList**
  - A more sophisticated approach
- Pre-packaged reference physics lists
  - The recommended approach
  - **QGSP_BERT**

What is a Physics List?

- A class that collects all the particles, physics processes, and production thresholds needed by your application
- One and only one physics list should be present in each Geant4 application
- There is no default physics list: it should always be explicitly specified
- It is a very flexible way to build a physics environment
  - User can pick only the particle he needs
  - User can assign to each selected particle only the processes he is interested
- But, user must have a good understanding of the physics required in his application
  - Omission of particles or physics processes will cause errors or poor simulation

Why do we need a Physics List? (1/2)

Nature has just one "physics": so why Geant4 does not provide a complete and unique set of particles and physics processes that everyone can use?

- There are many different physics models, corresponding to a variety of approximations of the real phenomena
  - very much the case for hadronic physics
  - but also for electromagnetic physics

  According to the application, one can be better than another
  Comparing them can give an idea of systematic errors

- Computation speed is important for simulations
  - A user may want a less detailed but faster approximation

- No application requires all the physics and particles
  - e.g. most high-energy applications do not need detailed transportation of low-energy neutrons

Why do we need a Physics List? (2/2)

For this reason, Geant4 takes a component-based approach to physics

- Provide many physics components (processes) which are de-coupled from one another
- User selects these components in custom-designed physics lists in much the same way as a detector geometry is built
- Exceptions:
  - A few electromagnetic processes must be used together
  - Future processes involving intereference of electromagnetic and strong interactions may require coupling as well

Physics processes in Geant4

- Electromagnetic physics
  - "standard" processes valid from ≈ 1 keV to ≈ PeV
  - "low-energy" processes valid from ≈ 250 eV to ≈ PeV
  - optical photons
- Weak physics
  - decay of subatomic particles
  - radioactive decay of nuclei
- Hadronic physics
  - pure hadronic processes valid from 0 to ≈ TeV
  - γ - , e - , μ – nuclear valid from ≈ 10 MeV to ≈ TeV
- Parameterized or "fast simulation" physics

G4VUserPhysicsList

- All physics lists must derive from this class
  - and then be registered with the Run Manager
- Example:

```
class MyPhysicsList: public G4VUserPhysicsList {
  public:
    MyPhysicsList();
   ~MyPhysicsList();

    void ConstructParticle();
    void ConstructProcess();
    void SetCuts();
}
```

- User must implement the methods:
  ConstructParticle , ConstructProcess , and SetCuts

ConstructParticle()

- Choose the particles you need in your simulation and define all of them here. Example:

```
void MyPhysicsList::ConstructParticle() {
  G4Electron::ElectronDefinition();
  G4Positron::PositronDefinition();
  G4Gamma::GammaDefinition();
  G4Proton::ProtonDefinition();
  G4Neutron::NeutronDefinition();
  ...
}
```

It is possible to use Geant4 classes that create group of particles (instead of individual ones):

G4BosonConstructor , G4LeptonConstructor ,
G4MesonConstructor , G4BaryonConstructor , G4IonConstructor

ConstructProcess()

- For each particle defined in **ConstructParticle()** assign all the physics processes that you want to consider in your simulation

```
void MyPhysicsList::ConstructProcess() {
    AddTransportation();
    // method provided by G4VUserPhysicsList , assign transportation process
    //  to all particles defined in ConstructParticle()

    ConstructEM();
    // convenience method that user may define; put electromagnetic physics here

    ConstructGeneral();
    // convenience method that user may define;
}
```

ConstructEM()

```
void MyPhysicsList::ConstructEM() {
  theParticleIterator->reset();
  while ( (*theParticleIterator)() ) {
    G4ParticleDefinition* particle = theParticleIterator->value();
    G4ProcessManager* pmanager = particle->GetProcessManager();
    G4String particleName = particle->GetParticleName();
    if ( particleName == "gamma" ) {
      pmanager->AddDiscreteProcess( new G4GammaConversion() );
      …
    }
    …
}
```

ConstructGeneral()

```
void MyPhysicsList::ConstructGeneral() {

   G4Decay* theDecayProcess = new G4Decay();
   theParticleIterator->reset();

   while ( (*theParticleIterator)() ) {

      G4ParticleDefinition* particle = theParticleIterator->value();
      G4ProcessManager* pmanager = particle->GetProcessManager();

      if ( theDecayProcess->IsApplicable( *particle ) ) {

         pmanager->AddProcess( theDecayProcess );
         pmanager->SetProcessOrdering( theDecayProcess, idxPostStep );
         pmanager->SetProcessOrdering( theDecayProcess, idxAtRest );
      }
}
```

SetCuts()

```
void  MyPhysicsList::SetCuts() {

   defaultCutValue = 1.0*mm;

   SetCutValue( defaultCutValue, "gamma" );
   SetCutValue( defaultCutValue, "e-" );
   SetCutValue( defaultCutValue, "e+" );
   // These are all the production cut values you need to set
   //  not required for any other particle
}
```

G4VModularPhysicsList

Use **G4ModularPhysicsList** to build a realistic physics lists which would be too long, complicated and hard to maintain with the previous approach.
Its features are:

- Derived from **G4VUserPhysicsList**
- **AddTransportation()** automatically called for all registered particles
- Allows to define "physics modules":
  - electromagnetic physics
  - hadronic physics
  - decay physics
  - ion physics
  - radioactive physics
  - optical physics
  - etc.

A simple G4VModularPhysicsList

- Constructor

```
MyModPhysList::MyModPhysList() : G4VModularPhysicsList() {
   defaultCutValue = 1.0*mm;
   RegisterPhysics( new ProtonPhysics() );
   // all physics processes having to do with protons
   RegisterPhysics( new ElectronPhysics() );
   // all physics processes having to do with electrons
  RegisterPhysics( new DecayPhysics() );
  // decay of unstable particles
}
```

- SetCuts

```
 void MyModPhysList::SetCuts() {
   SetCutsWithDefault();
 }
```

Physics Constructors

Allow you to group particle and process construction according to physics domains

```
class ProtonPhysics : public G4VPhysicsConstructor {
  public:
    ProtonPhysics( const G4String& name = "proton" );
    virtual ~ProtonPhysics();

    virtual void ConstructParticle();
    // easy - only one particle to build in this case

    virtual void ConstructProcess();
    // put here all the processes a proton can have
}
```

Reference Physics Lists

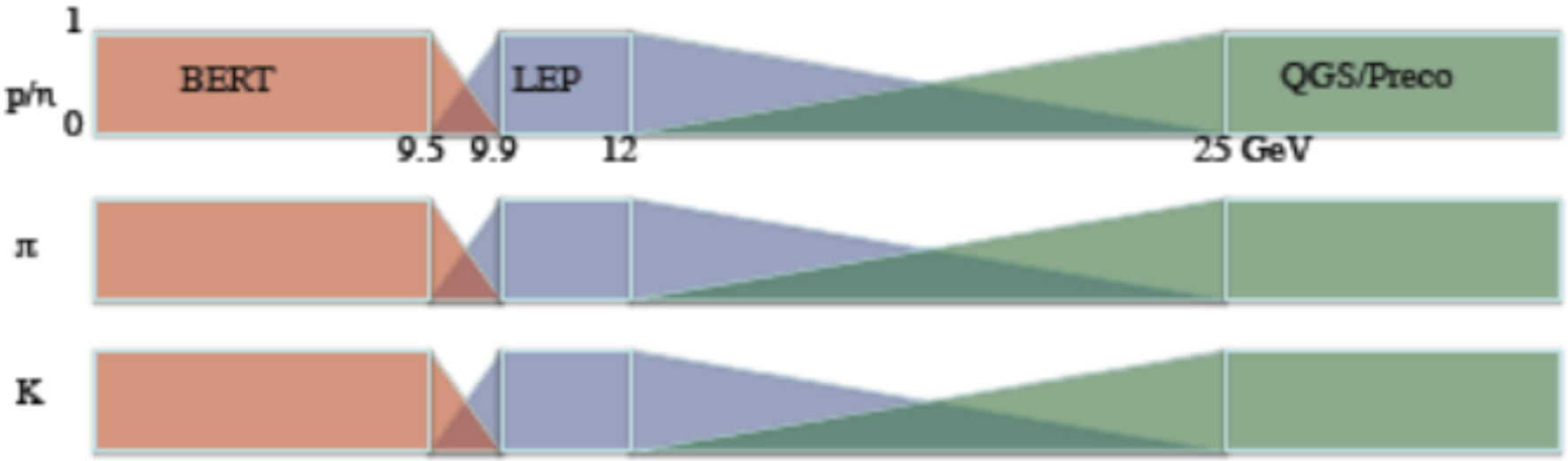
- Writing a complete and realistic physics list for EM physics and even more for hadronic physics is involved, and it depends on the application. To make things easier, pre-packaged ***reference physics lists*** are now provided by Geant4, according to some reference use cases.

- Few choices are available for EM physics (different production cuts and/or multiple scattering); instead, several possibilities are available for hadronics physics:
  LHEP,
  QGSP_BERT, **QGSP_BERT_EMV**, **QGSP_BERT_EMX**, **QGSP_BERT_HP**,
  QGSP_BIC, **QGSP_BIC_EMY**,
  FTFP_BERT, **FTF_BIC**,
  QGSC_BERT, **CHIPS**, ...

- These lists are "best guess" of the physics needed in a given case; they are intended as starting point (and their builders can be re-used); the user is responsible of validating them for his application.

QGSP_BERT (1/2)

- Let's give a look to the reference physics list QGSP_BERT, which is the most used one in high-energy physics
  - used in production by ATLAS and CMS
  - very similar also to the one used by BaBar
- Various hadronic models are used for different particles and in certain kinetic energy ranges:



QGSP_BERT (2/2)

- QGS (Quark Gluon String) model **[12 GeV, 10 TeV]** for pions, kaons, protons and neutrons
- LEP (Low Energy Parameterized) model **[9.5 GeV, 25 GeV]**
- BERT (Bertini cascade) model **[0, 9.9 GeV]** for pions, kaons, protons and neutrons
- Nucleus de-excitation: Precompound + evaporation
- Neutron capture and fission: parameterized models
- Other hadrons (hyperons, anti-baryons, light ions): parameterized model
- CHIPS: capture of negative hadrons; elastic for protons and neutrons; quasi-elastic, gamma-nuclear, electron-nuclear
- Standard electromagnetic

Summary

- All the particles, physics processes, and production cuts needed for an application must go into a physics list
- Two kinds of physics list classes are available for users to derive from:
  - **G4VUserPhysicsList** : for relatively simple physics lists
  - **G4VModularPhysicsList** : for detailed physics lists
- Some pre-packaged reference physics lists are provided by Geant4 as starting point for users
- Care is required by the user to compose or choose the right physics list to use for his application