Advanced Topics and Recent Topics

Koichi Murakami

KEK/CRC

Outline

More advanced features

- Supplemental about packaged physics list
- Stack management
- Shower parameterization
- Pre-assigned decay
- Python interface

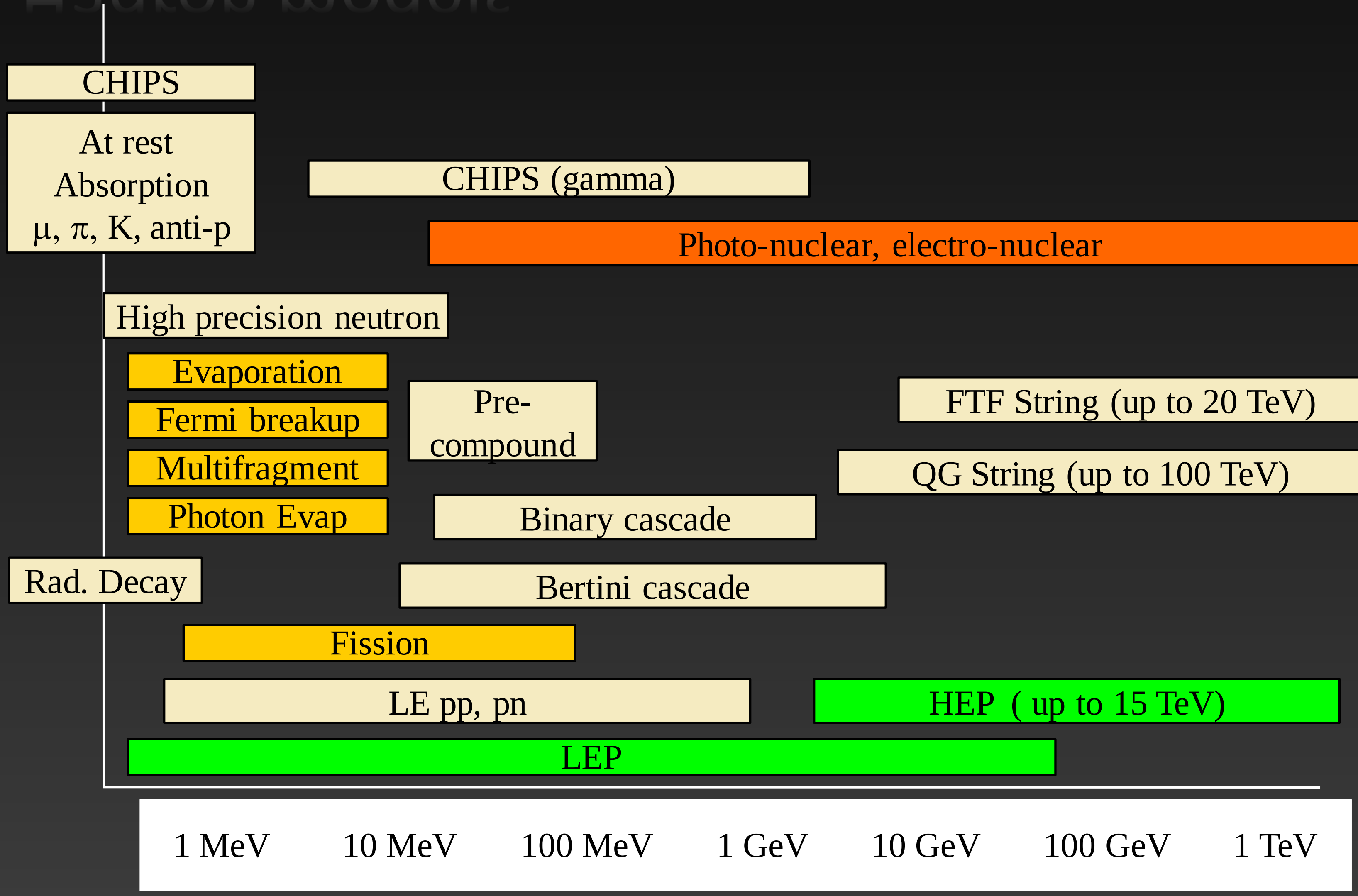
Reports from Geant4 Collaboration workshop

- Physics validation
- Performance issues
- Recent improvements

Discussion

SUPPLEMENTAL ABOUT PACKAGED PHYSICS LIST

Hadron models



Groups of lists – hadronic options

LHEP using parameterized models

QGS/FTF series replaces parameterized high energy model

- theory driven string model used for pion, protons, neutrons, kaons
- Improved cross-sections
- Better description for stopping particles using CHIPS modeling
- Revised elastic scattering

Variations for modeling at medium and low energies (e.g. *QGSP_BIC*)

- Cascade model, precompound model, CHIPS, low energy neutron transport

Inventory of Reference PL (G4 8.3)

LHEP
LHEP_EMV
LHEP_BERT
LHEP_BERT_HP
LHEP_BIC
LHEP_BIC_HP
LHEP_PRECO
LHEP_PRECO_HP
LHEP_HP
LHEP_LEAD
LHEP_LEAD_HP

QGSP
QGSP_EMV
QGSP_EMX
QGSP_NQE
QGSP_EMV_NQE
QGSP_BERT
QGSP_BERT_EMV
QGSP_BERT_HP
QGSP_BERT_NQE
QGSP_BERT_TRV
QGSP_BIC
QGSP_BIC_HP
QGSP_HP
QGSP_QEL

QGSC
QGSC_EMV
QGSC_EFLOW
QGSC_LEAD
QGSC_LEAD_HP

LBE

FTFC
FTFP_EMV
FTFP

QBBC

Lists in italic deleted in 9.0

Examples

LHEP:

- fast, for shower simulation in calorimeter
- Paratmeterized modeling for all hadronic interactions
- Standard EM physics

QGSP

- QGS for high energy interaction, using Precompound for nuclear de-excitation
- LEP for low energy (< 12-25 GeV)
- LHEP models for particles other than proton, neutron, pion, Kaon
- Elastic scattering was changing
- Standard EM physics

QGSP_BERT_HP

- Radiation background studies – good modeling for neutron production and transport
- Usage of LEP reduced by BERT for nucleons, pions, kaons below 10 GeV
- Add HP neutron transport for neutrons < 20MeV

QGSC

- Like QGSP with improved nuclear fragmentation provided by CHIPS

FTFP

- Alternative string model, under revision

QGSP_EMV, LHEP_EMV

- Alternate EM physics, using faster multiple scattering similar to 7.1

Updates in 9.0

Rename components for EM physics

- G4EMStandardPhysics, the default EM option
- G4EMStandardPhysics_option1, used by _EMV variants
  - ✓ Was G4EMStandardPhysics71
- G4EMStandardPhysics_option2, used by _EMX variants
  - ✓ Was G4EMStandardPhysics72

Removed obsolete lists

Threshold for use of FTF model lowered to 5GeV

- FTF is under development

Plans

Improve documentation

- draft pages available at *http://cern.ch/geant4/support/proc_mod_catalog/physics_lists/physicsLists.shtml*

Continue to provide new model developments

- e.g. add quasi-elastic channel for string models
- New options as new (experimental) physics lists
- Adopt mature options

Better integrate physics lists for communities like underground experiments, space users, …

STACK MANAGEMENT

Track stacks in Geant4

By default, Geant4 has three track stacks.

- "Urgent", "Waiting" and "PostponeToNextEvent"
- Each stack is a simple "last-in-first-out" stack.
- User can arbitrary increase the number of stacks.

ClassifyNewTrack() method of *UserStackingAction* decides which stack each new coming track to be stacked (or to be killed).

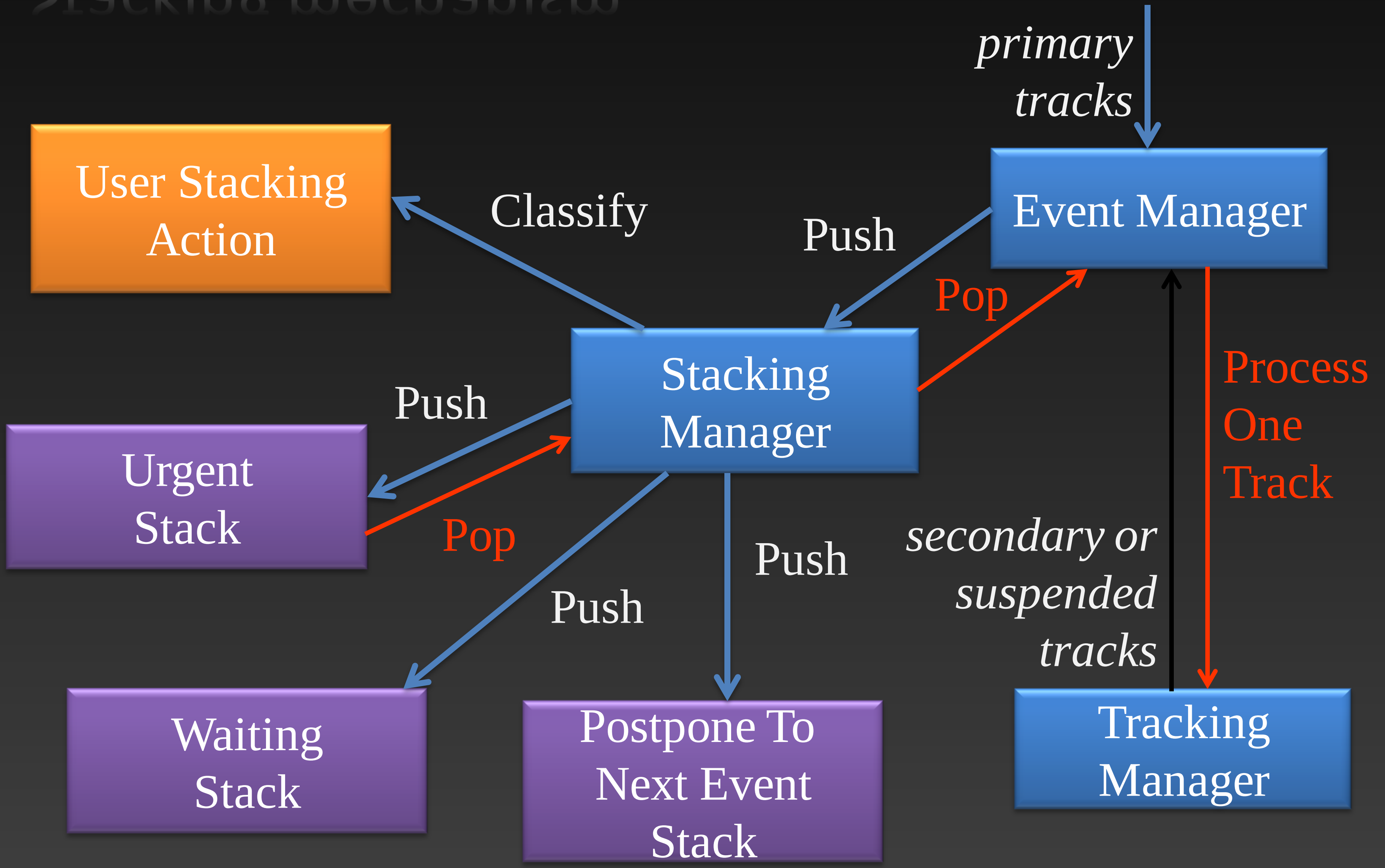
- By default, all tracks go to Urgent stack.

A Track is popped up only from Urgent stack.

Once Urgent stack becomes empty, all tracks in Waiting stack are transferred to Urgent stack.

- And NewStage() method of *UsetStackingAction* is invoked.

Stacking mechanism



G4UserStackingAction

User has to implement three methods.

G4ClassificationOfNewTrack ClassifyNewTrack(const G4Track*)

- Invoked every time a new track is pushed to G4StackManager.
- Classification
  - ✓ fUrgent - pushed into Urgent stack
  - ✓ fWaiting - pushed into Waiting stack
  - ✓ fPostpone - pushed into PostponeToNextEvent stack
  - ✓ fKill – killed

void NewStage()

- Invoked once Urgent stack becomes empty and all tracks in Waiting stack are transferred to Urgent stack
- All tracks which are transferred from Waiting stack to Urgent stack can be re-classified by invoking stackManager->ReClassify()

void PrepareNewEvent()

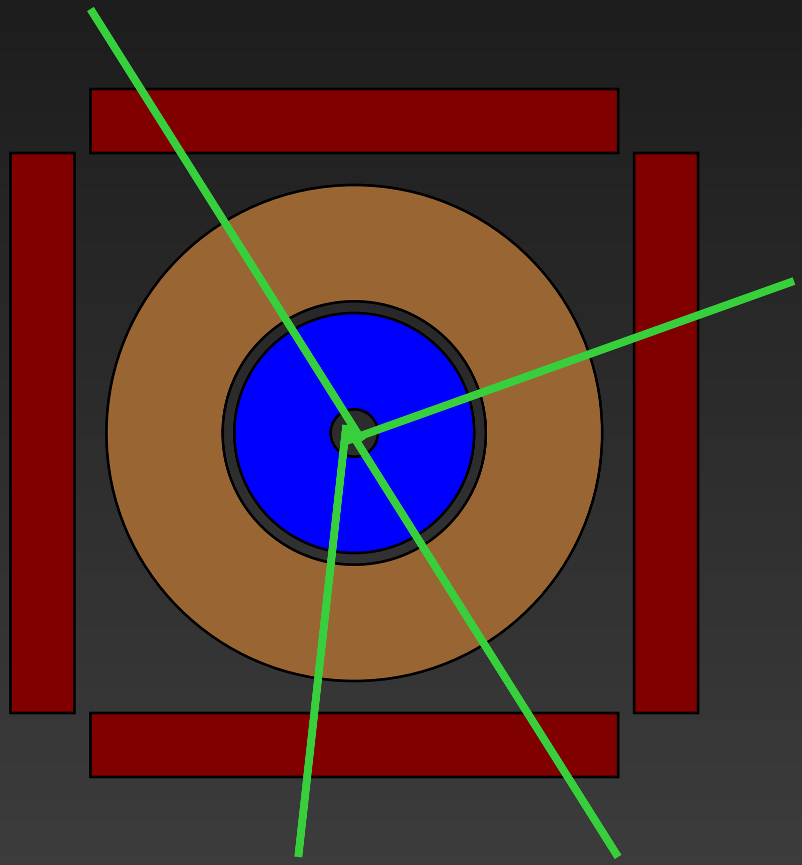
- Invoked at the beginning of each event for resetting the classification scheme.

ExN04StackingAction

ExampleN04 has simplified collider detector geometry and event samples of Higgs decays into four muons.

Stage 0

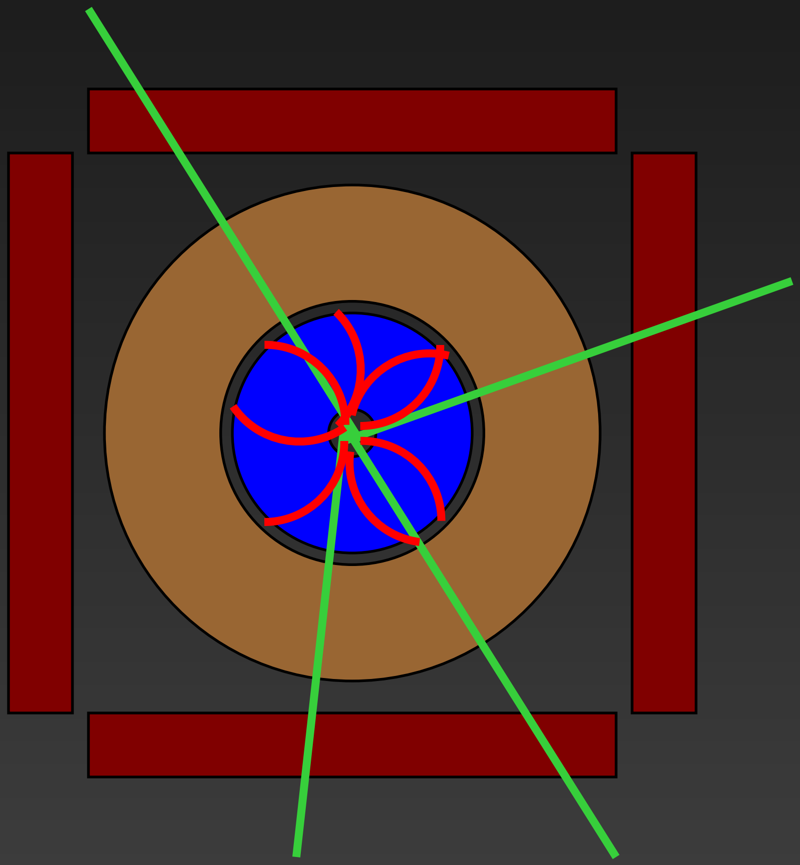
- Only primary muons are pushed into Urgent stack and all other primaries and secondaries are pushed into Waiting stack.
- All of four muons are tracked without being bothered by EM showers caused by delta-rays.
- Once Urgent stack becomes empty (i.e. end of stage 0), number of hits in muon counters are examined.
- Proceed to next stage only if sufficient number of muons passed through muon counters. Otherwise the event is aborted.



ExN04StackingAction

Stage 1

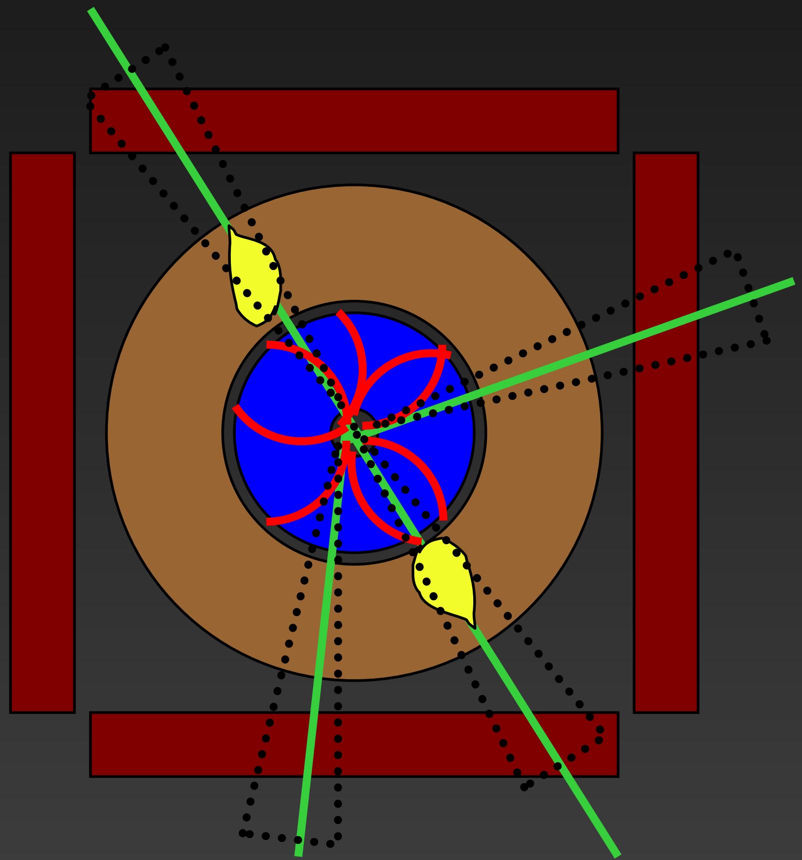
- Only primary charged particles are pushed into Urgent stack and all other primaries and secondaries are pushed into Waiting stack.
- All of primary charged particles are tracked until they reach to the surface of calorimeter. Tracks reached to the calorimeter surface are suspended and pushed back to Waiting stack.
- All charged primaries are tracked in the tracking region without being bothered by the showers in calorimeter.
- At the end of stage 1, isolation of muon tracks is examined.



ExN04StackingAction

Stage 2

- Only tracks in "region of interest" are pushed into Urgent stack and all other tracks are killed.
- Showers are calculated only inside of "region of interest".



Shower parameterization

FAST SIMULATION

Fast simulation - Generalities

Fast Simulation, also called as *shower parameterization*, is a shortcut to the "ordinary" tracking.

Fast Simulation allows you to take over the tracking and implement your own "fast" physics and detector response.

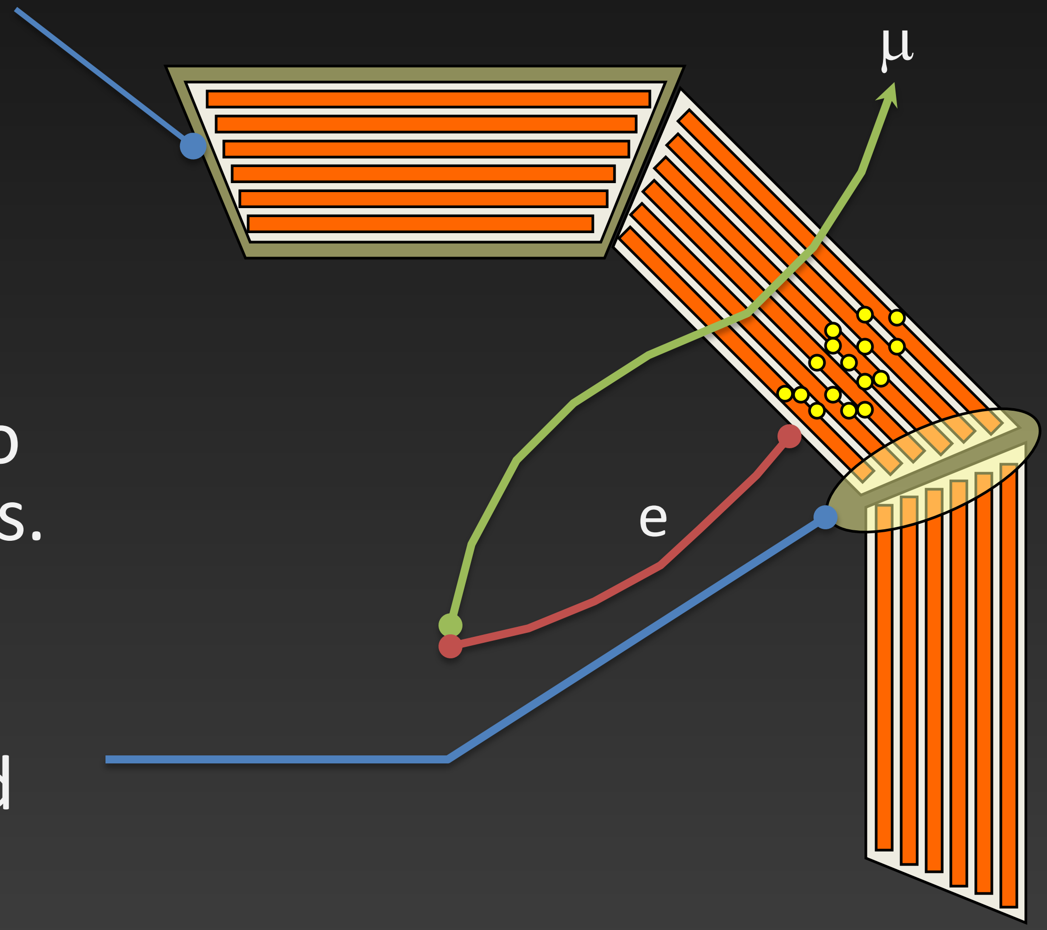
Parameterizations are generally *experiment dependent*. Geant4 provides a convenient framework.

Parameterization features

Parameterizations take place in an envelope.

Parameterizations are often dependent to particle types and/or may be applied only to some kinds of particles.

They are often not applied in complicated regions.

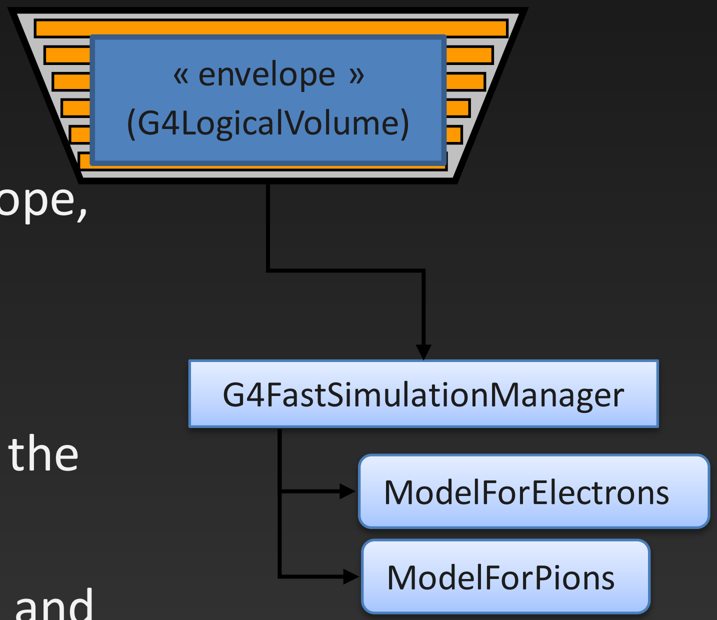


Models and envelope

Concrete models are bound to the envelope through a *G4FastSimulationManager* object.

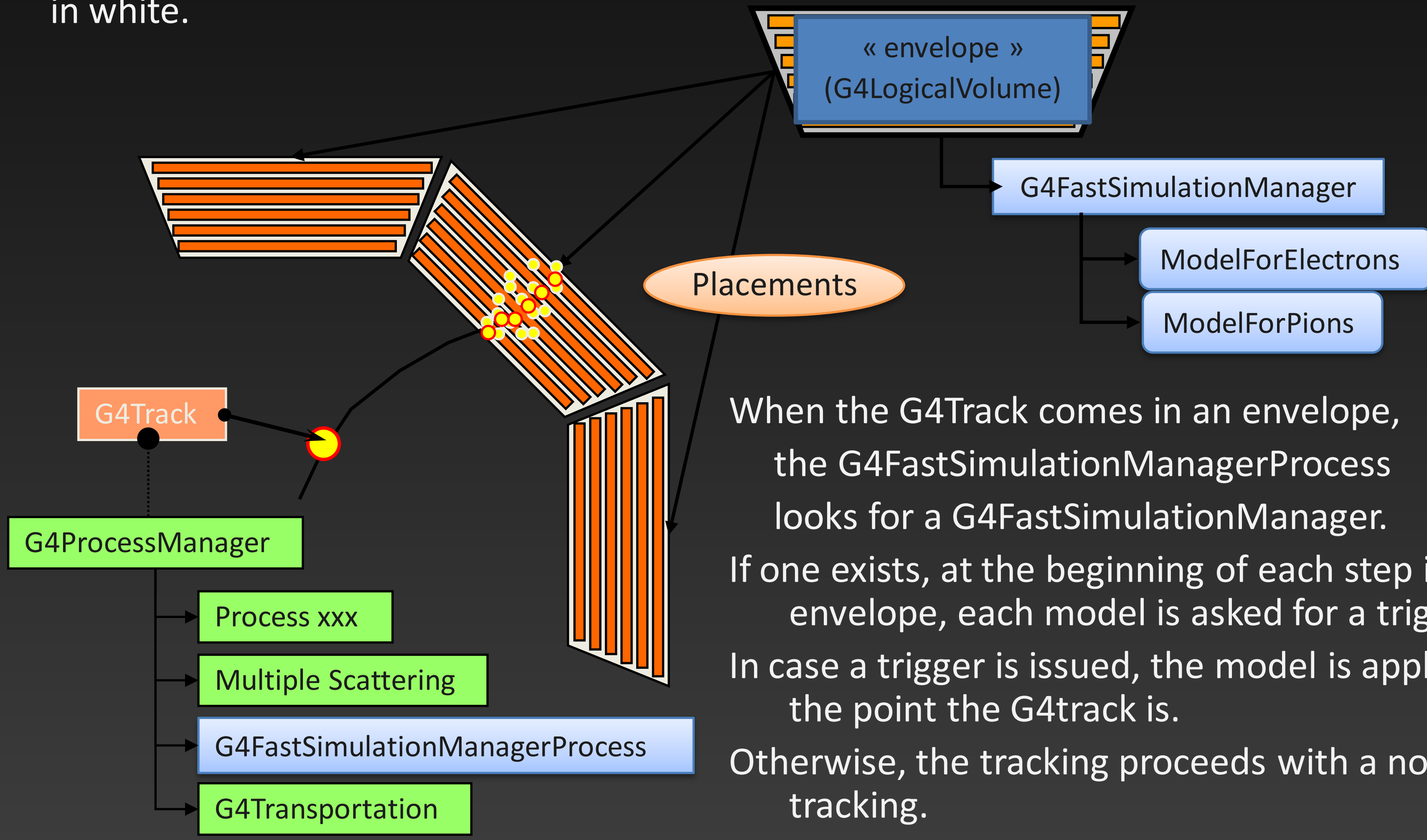
This allows several models to be bound to one envelope, i.e. G4LogicalVolume.

A model may returns back to the "ordinary" tracking the new state of G4Track after parameterization (alive/killed, new position, new momentum, etc.) and eventually adds secondaries (e.g. punch through) created by the parameterization.



Fast Simulation

The Fast Simulation components are indicated in white.



When the G4Track comes in an envelope, the G4FastSimulationManagerProcess looks for a G4FastSimulationManager.

If one exists, at the beginning of each step in the envelope, each model is asked for a trigger.

In case a trigger is issued, the model is applied at the point the G4track is.

Otherwise, the tracking proceeds with a normal tracking.

G4FastSimulationManagerProcess

The *G4FastSimulationManagerProcess* is a process providing the interface between the tracking and the fast simulation.

It has to be set to the particles to be parameterized:

- The process ordering must be the following:
  - [n-3] …
  - [n-2] Multiple Scattering
  - [n-1] G4FastSimulationManagerProcess
  - [n] G4Transportation
- It can be set *as a discrete process* or it must be set as *a continuous & discrete process* if using ghost volumes.

Ghost Volume

Ghost volumes allow to define envelopes independent to the volumes of the tracking geometry.

In addition, Ghost volumes can be sensitive to particle type, allowing to define envelops *individually to particle types*.

The *G4FastSimulationManagerProcess* provides the additional navigation inside a ghost geometry. This special navigation is done transparently to the user.

PRE-ASSIGNED DECAY TABLE

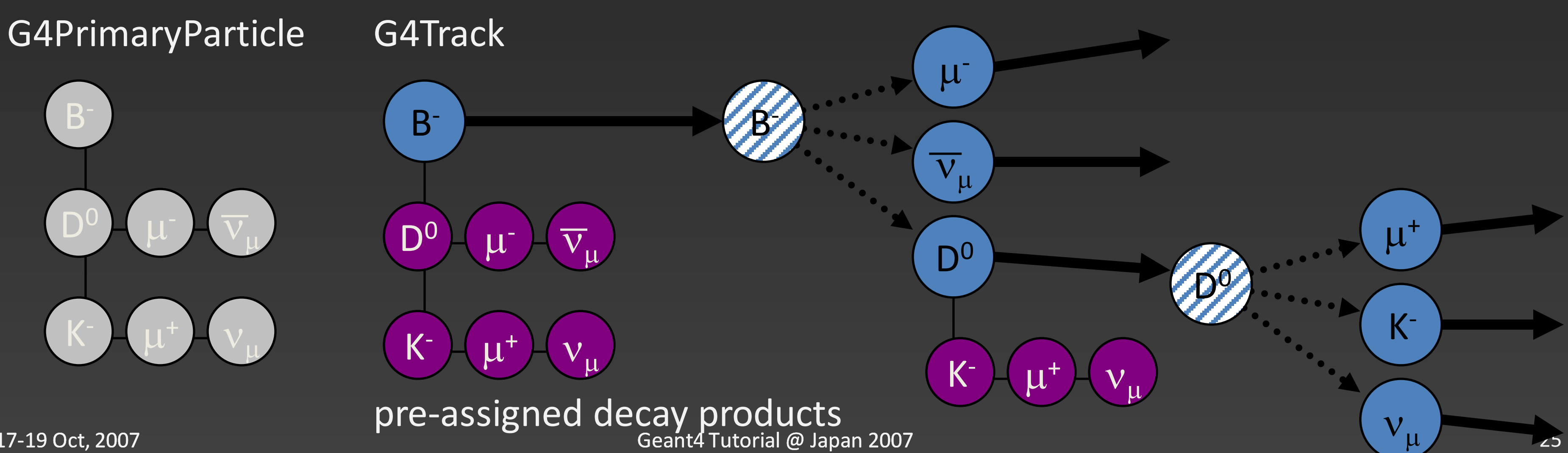
Pre-assigned decay products

Geant4 provides decay modes for long-lived particles, but decay modes for short-lived (e.g. heavy flavour) particles are not provided by Geant4

- decay process can invoke an external decay handler (G4VExtDecayer)
- Or, user must "pre-assign" proper lifetime and decay products to the parent G4PrimaryParticle.

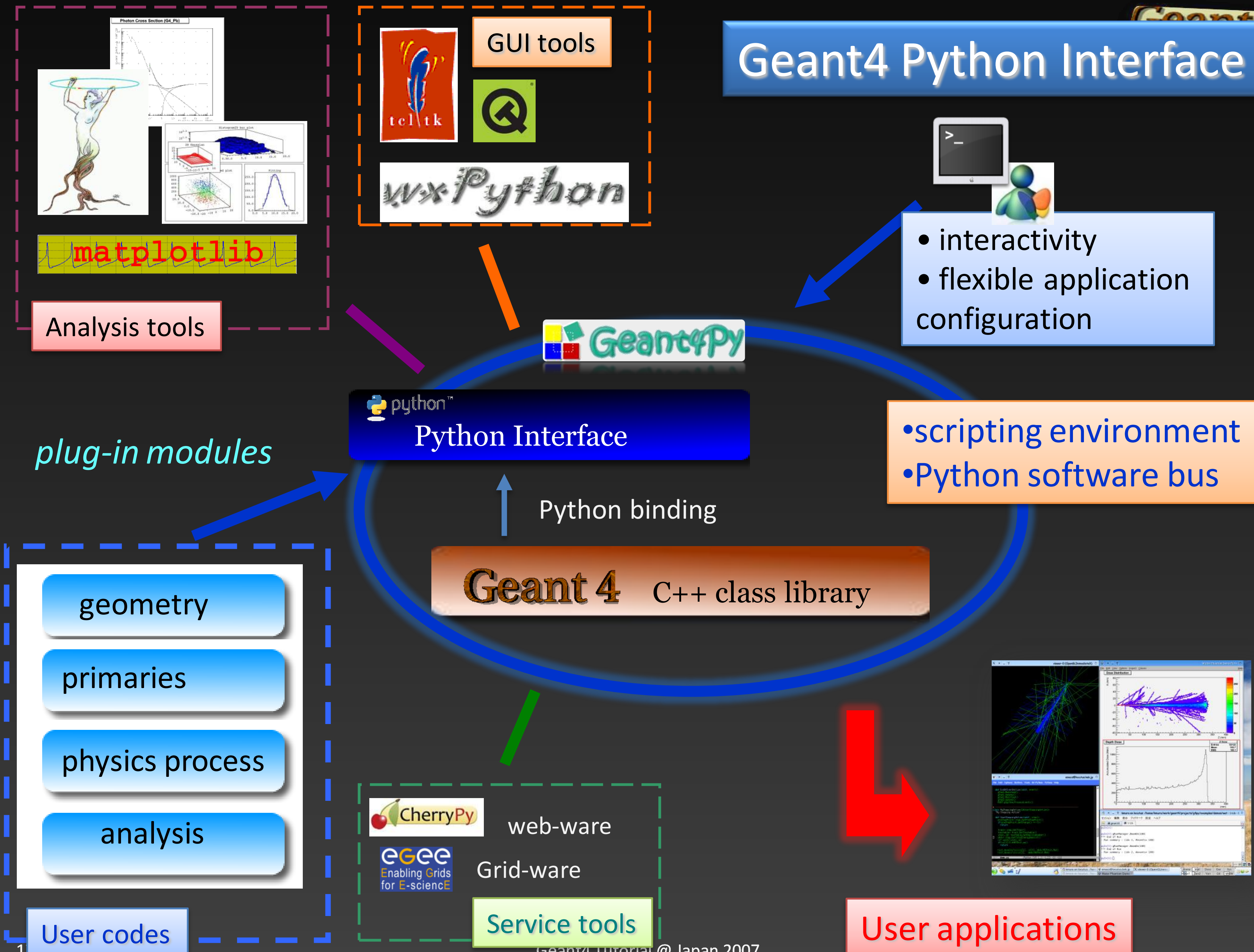
A parent particle in the form of G4Track object travels in the detector, bringing "pre-assigned" decay daughters as objects of G4DynamicParticle.

- When the parent track comes to the decay point, pre-assigned daughters become to secondary tracks, instead of randomly selecting a decay channel defined to the particle type.
- Decay time of the parent can be pre-assigned as well.



PYTHON INTERFACE

Geant4 Python Interface



Exposed Classes

Over 100 classes in different categories are exposed to Python.

Classes for Geant4 managers

- G4RunManager, G4EventManager, …
- automatically instantiated as global variables
  - ✓ gRunManager, gEventManager, …

Classes of base classes of user actions

- G4UserDetetorConstruction, G4UserPhysicsList,
- G4UserXXXAction
  - ✓ PrimaryGenerator, Run, Event, Stepping,...
- can be inherited in the Python side

Classes having information to be analyzed

- G4Step, G4Track, G4StepPoint, G4ParticleDefinition, ...
- Only safe methods are exposed.
  - ✓ Getting internal information are exposed. Some setter methods can easily break simulation results.

Classes for describing user inputs

- G4ParticleGun, G4Box, G4PVPlacement, ...
- G4String, G4ThreeVector, G4RotationMatrix,...as utility classes

An Example of C++ vs. Python

```cpp
Class MySteppingAction : public
  G4UserSteppingAction {
// My Stepping Action

void SteppingAction
   (const G4Step* step) {
  G4StepPoint* preStepPoint=
    step->GetPreStepPoint();
  G4Track* track=
     step->GetTrack();
  G4VTouchable* touchable=
     track->GetTouchable();

  if (preStepPoint->GetCharge()
      == 0) return;

  G4ThreeVector pos=
     preStepPoint->GetPosition()
  G4int id=
     touchable->GetReplicaNumber()
  G4double dedx=
     step->GetTotalEnergyDeposit();
  }
};
```

```python
class MySteppingAction
   (G4UserSteppingAction):
   "My Stepping Action"

   def SteppingAction(self, step):
     preStepPoint= step.GetPreStepPoint()
     track= step.GetTrack()
     touchable= track.GetTouchable()

     if(preStepPoint.GetCharge() == 0):
       return

     pos= preStepPoint.GetPosition()
     id= touchable.GetReplicaNumber()
     dedx= step.GetTotalEnergyDeposit()
```

- Base classes can be inherited and virtual methods can be implemented in the Python side.
- Easy to convert between C++ and Python

Practical Notes for Installation

Included in "`environments/g4py/`" directory

Boost-C++ library is required for wrapping out C++ classes/functions.

Shared libraries are required because of dynamic binding.

- Any external libraries are also required to be built in shared libraries.

Global libraries are required because Geant4Py does not know which granular libraries are used in your application.

12th Geant4 Collaboration Workshop

http://indico.cern.ch/conferenceDisplay.py?confId=10311

- Manchester, 13-19/Sep/2007
- User presentations
- Collaboration workshop

Hot/Key issues

"Geant4 Physics validation" is always a hot topic.

- At increasing accuracy for established uses
  - ✓ from HEP experiments, underground experiments, to space and medical applications
- Establishing testing suites for physics lists/processes

What is available in *Physics Lists*?

- How to find out/communicate physics performance

Improve computing performance

- performance monitor, benchmark
- profiling user program
- code review for gaining performance
- challenge in geometry and field

Physics validation activities

Validation with test beam data from LHC calorimeter (ATLAS/CMS)

Shower shapes

- hadronic shower shape study for LHC calorimeter
- CMS shower shapes
- hadronic validation at FNAL (ITEP, CMS)

CALICE rerport

Underground physics comminity (ILIAS)

Low background experiments

BESIII

Hadron validation suite

Validation suite for thin target data on hadron inelastic interaction

- Exist since 2002
- Neutron production by p, d, α, ^{12}C with E <= 3 GeV
  - p + A -> n + X
  - d + A -> n + X
  - α + A -> n + X
  - ^{12}C + A -> n + X
- Pion production by protons and pions P<13 GeV/c
  - P + A -> $\pi^{\pm}$ + X
- More 100 thin target setups
- Data versus Geant4 models
- Control on differential spectra
- Model level test
- Models under testing:
  - PreCompound
  - Binary Cascade
  - Binary Ion cascade
  - Bertini Cascade
  - Wilson-Abrasion model
  - LHEP
  - QGSP
  - QGSC
  - FTFP
- A new model can be easily included
- About 1000 comparison plots produced

V.Ivanchenko et al., CHEP'07, Victoria, Canada, Sept. 2-7, 2007

10

Improvements

Improvements/monitor on CPU performance

Improvement on tracking performance

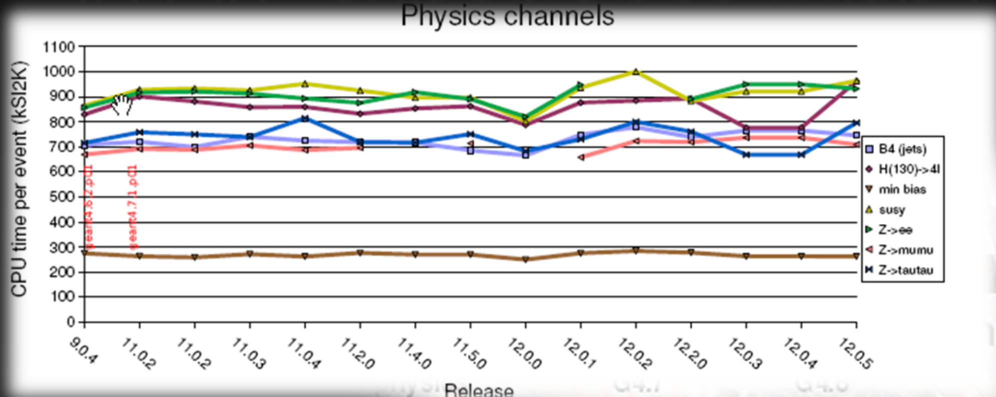
- parallel navigation and scoring
- tracking in magnetic field
- Geant4e

Improvement on geometry description

Improvements on physics performance

- new physics process/model

ATLAS experience with G4 performance



G4.8.3.p01 results

t (kSI2K)

channels	QGSP_GN	QGSP_EMV	G4.8 QGSP	G4.8 QGSP 1mm	G4.8 QGSP_BERT
susy	921,64	1123,82	1956,42	1560,52	2594,16
Zee	949,58	1107,58	1944,05	1546,41	2432,79
Ztautau	668,64	831,19	1429,71	1361,49	2129,3
H(1[illegible])4l	776,72	1067,55	1793,55	1468,79	2334,59
MB	263,35	332,66	584,2	509,29	805,98
jets	765,06	920,77	1480,34	1328,76	1957,11

Ratios

physics channels	QGSP_EMV/ QGSP_GN	QGSP/ QGSP_EMV	QGSP1mm/ QGSP_GN	QGSP_BERT/ QGSP	QGSP_BERT/ QGSP_EMV
susy	1,22	1,74	1,69	1,33	2,31
Zee	1,17	1,76	1,63	1,25	2,2
Ztautau	1,24	1,72	2,04	1,49	2,56
H(130)4l	1,37	1,68	1,89	1,3	2,19
MB	1,26	1,76	1,93	1,38	2,42
[illegible]	1,2	1,61	1,74	1,32	2,13

- No particular runtime problems found: both QGSP and QGSP_EMV ran fine
- Performance comparison with g4.7.1
- First look at performance of QGSP_BERT
- Increase of QGSP_EMV wrt g4.7.1

CPU performance monitor, an example

Tracking in Magnetic Field: QGSP_EMV Physics List

BaBar Tracker

Same Geant4 example as in the previous slide, but this time with the QGSP_EMV Physics List. 100 B-Bbar events simulated.
Local build with static libraries.

Release	sec/event	Ratios	
7.1.p01a	**3.04**	**1.00**	**(QGSP_GN)**
8.0.p01	3.78	1.24	*
8.1.p02	3.85	1.27	*
8.2	3.72	1.22	*
8.2.p01	3.84	1.26	*
8.3	3.91	1.29	*
8.3.p01	3.89	1.28	*
9.0	3.57	1.17	<--- *Code review of* ***Electromagnetic*** *physics module*
9.0.p01	3.62	1.19	

* The variations are due to tuning and adding safety checks to Urban Multiple Scattering model.

CPU optimization for Geant4 9.0

The review and optimization of interfaces have been performed

- G4VEmModel
- G4VEnergyLossProcess
- G4VEmProcess
- G4VMultipleScattering
- Modifications were provided for all derived classes

Reduction of usage of virtual methods

Reuse STL vectors - reduced calls to new and delete for intermediate vectors

Minor optimization of G4UrbanMscModel code

Summary effect *about 10%* for EM showers

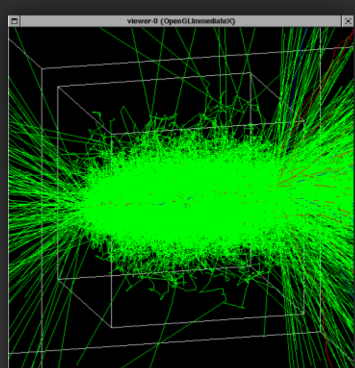
Parallel Geometries & Navigation

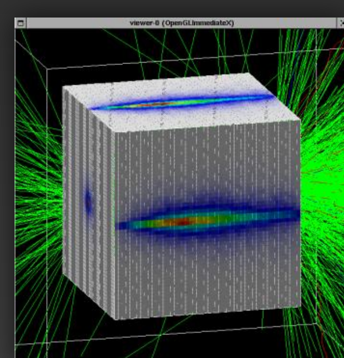
since 4.8.2, default for biasing in 4.9

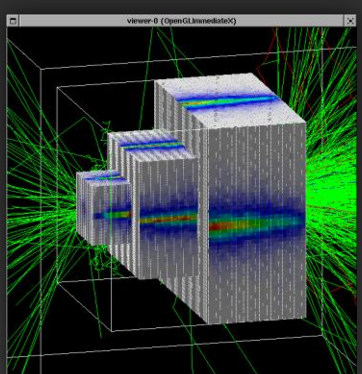
Possibility to define geometry trees which are "parallel" and overlapping to the tracking geometry

- Each assigned to a dedicated navigator object
- Navigation transparently happens in sync with the normal tracking (mass) geometry
  - ✓ Applies transparently to transport in magnetic field

Use cases: fast shower parameterisation, geometrical biasing, particle scoring, readout geometries, etc ...







Error propagation module Geant4e

Geant 4 — **What is GEANT4E** — Ciemat

- Track reconstruction needs to match signals in two detector parts
  - Propagate tracks from one detector part to another and compare with real measurement there
  - Make the average between the prediction and the real measurement

  ⇒ it needs the track parameter errors

- Many experiments have used in the past GEANE (based on GEANT3) or their 'ad hoc' solution

GEANT4e provides this functionality for the reconstruction software in the context of GEANT4

Released in geant4.9.0

Stepper performance

Table. Timing Steppers

Stepper	Order	Time for One Step Unit=1e-15 sec	Time for Accurate Advance 120 mm in Quadripole Field : Bx=Grad*x; By=-Grad*y Gradient=1 T/m , Epsilon=1e-5
RungeKutta4	4th	4.58 units	**17.4 units**
RKG3_Stepper	4th	4.33	**17.4**
CashKarp45	5th	**2.58**	**12.9**
SimpleHeum	3th	3.16	**17.0**
RungeKutta2	2th	**2.08**	23.83
ExactHelixStepper (revised)		**1.58**	Only Uniform Field
ExactHelixStepper (old)		1.7	Only Uniform Field
HelixExplicitEuler (revised)	1th	**2.7**	28.92
HelixExplcitEuler(old)	1th	3.5	Old version
HelixImplicitEuler	2th	6.9	40.66
ExplicitEuler	1th	**1.08**	673
ImplicitEuler	2th	**2.16**	25.7

Low order steppers and CashKarp45 are faster **per step**
But high order steppers are more precise and need less steps **per trajectory**

Tests on tracking in field

In progress : Tests with Geometry and not-Uniform Field

1)LarCalorimeter example with " Exact " **ATLAS** Solenoid Field or Toroid Field (not a Field Map)

- For intersection studies and accuracy
- First Test : Accuracy of propagation

  Shooting muons (Different Energies, Different Angles)

  Parameters of Propagation as in ATLAS

  (DeltaIntersection=0.00001 mm, DeltaOneStep=0.0001 mm, MaxEpsilon=0.001, MinEpsilon=0.00001)

  Difference between full and empty geometry in order of **1.e-5 mm for 3 m** of track

2)NTST test with Uniform, Quadripole Field and "Tabulated" Solenoid Field

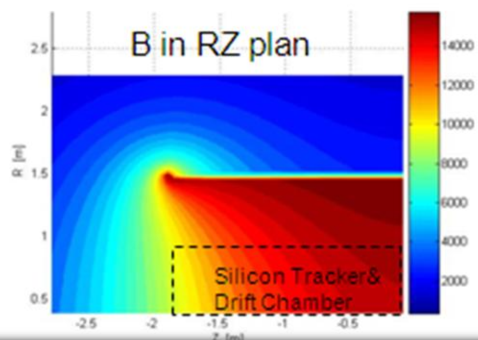
Semi Analytic formula for BaBar Solenoid : 1.5 T, Rext=1.5 m, Z half= 1.9 m

-BaBar Silicon Tracker and 40 layer Drift Chamber

-B-Bbar events

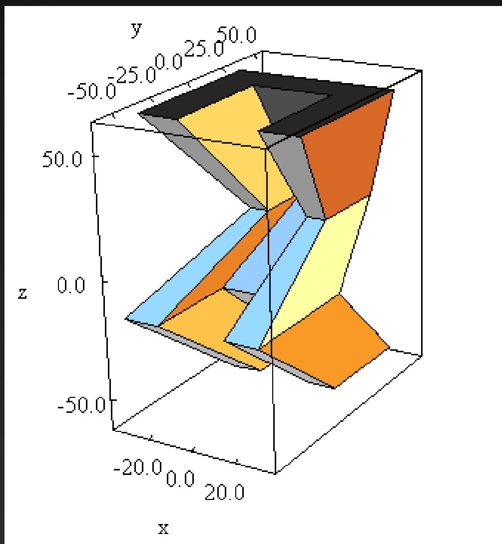
-For CPU benchmarks and accuracy

-For Testing Steppers



New solids : G4ExtrudedSolid and G4Paraboloid

Since 8.3

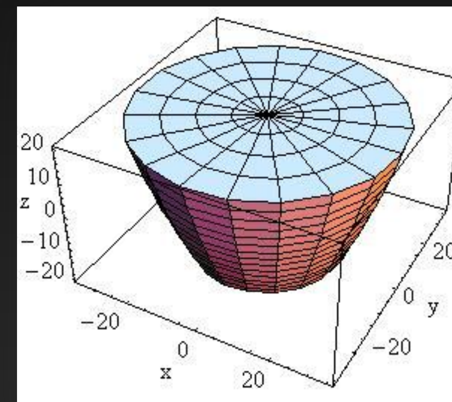


G4ExtrudedSolid by
Ivana Hrivnacova (IPN,Orsay,France)

G4ExtrudedSolid represents the extrusion Z direction of an arbitrary polygon

- For each Z section position with offset and scale factor is defined.
- The solid is implemented as a specification of G4TessellatedSolid

Scheduled for 4.9.1



G4Paraboloid by
Lukas Lindroos (Summer Student)

G4Paraboloid is a full Paraboloid with possible cut in Z

- Equation : $z = a*r^2+b$
- On the picture:
  G4Paraboloid(Name,dz,R1,R2)
  dZ = 20 and
  R1 = 20 (at z=-20)
  R2 = 35 (at z=20)

Multiple scattering options in G4 9.0

G4SafetyHelper class have been introduced

G4MscStepLimitType

- Minimal - equivalent to the algorithm of Geant4 7.1 and earlier releases (QGSP_EMV Physics Lists)

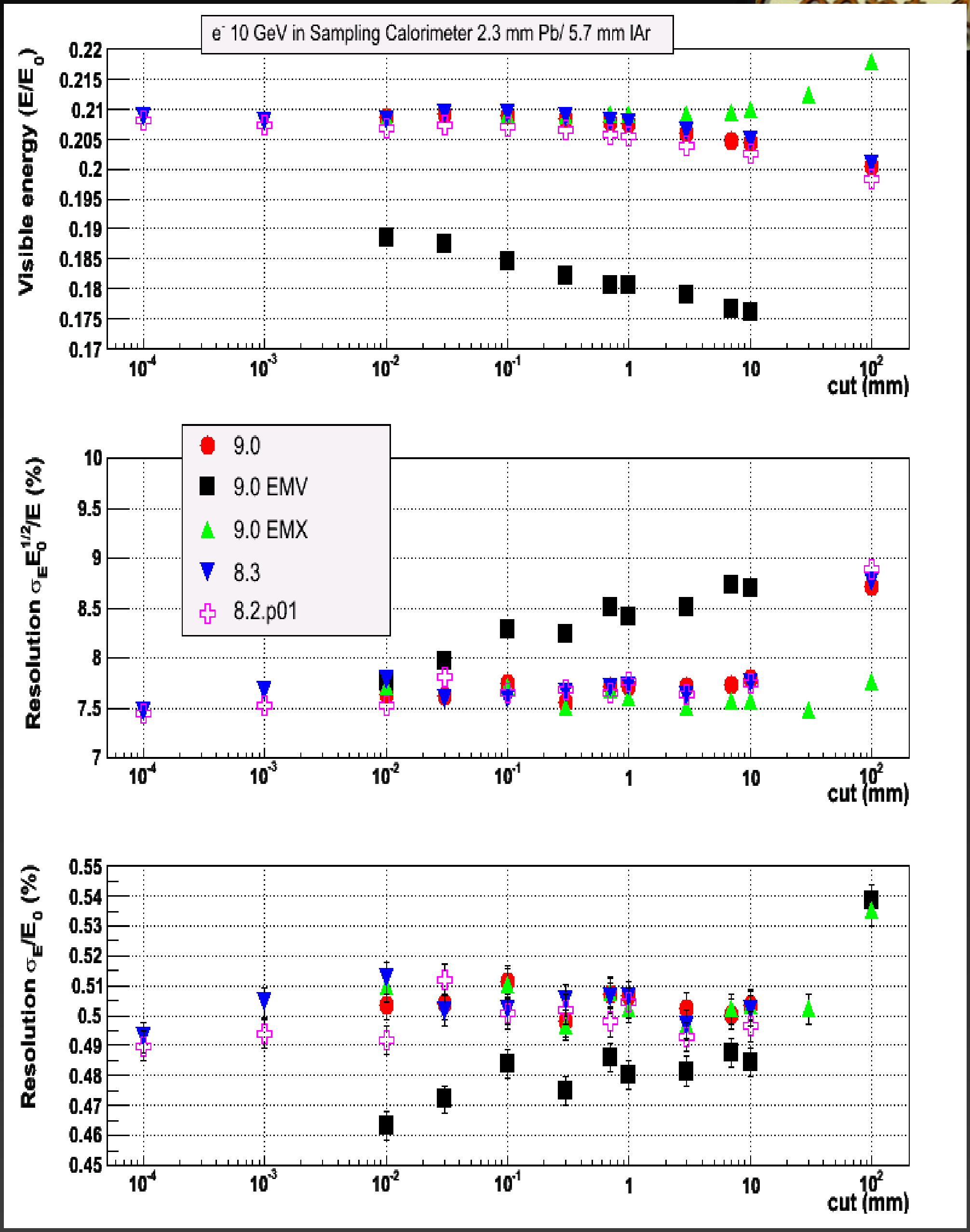
- UseSafety - the current default, uses geometrical safety (QGSP and QGSP_EMX Physics Lists)
  - ✓ *QGSP_EMX includes sub-cutoff option*

- UseDistanceToBoundary - the most advanced, recommended for accurate computations in the cases, where no magnetic field is set
  - ✓ *also option is recommended: skin = 2*

Multiple scattering options configurable via UI.

Calorimeter tests ATLAS barrel type

- Practically no difference between 8.3 and 9.0
- *EMV* results are the same as for 7.1p01
- Sub-cutoff option (EMX, cut = 7 mm) was optimized
  - G4SafetyHelper



Interface change with G4 9.0

EM base classes

- Renamed Physics Lists optional builders
- Renamed EM standard components in examples
- Renamed methods of G4EmProcessOptions
- New UI commands
  - ✓ */process/msc/*
- Updated interface to G4UrbanMscModel
  - ✓ Parameters can be changed between runs
- Use only G4SafetyHelper
  - ✓ not G4Navigator anymore
- Removed 52-type processes

EM Status Summary

Summary

- With Geant4 8.3 and 9.0 EM standard is capable to provide results on level of accuracy ~2 %
  - EMV Phys List is kept to be the same as default physics of Geant4 7.1p01
- 0.7 mm cut is the today default
  - Lower cuts not needed for LHC calorimeters
  - Lower cuts may be useful for tracking detectors
- Sub-cutoff option (EMX) provides stable results up to cut 10 mm
  - CPU performance of sub-cutoff needs to be upgraded
- There is a visible speed up for Geant4 9.0
- EM (standard) working group page:

http://cern.ch/geant4/collaboration/working_groups/electromagnetic/index.shtml

Next release, Geant4 9.1

14 Dec/2007 is the scheduled date for Geant4 9.1.