Shower Parameterisations (Fast simulation) - a shortcut to the tracking

Joanna Weng (CERN /University of Karlsruhe)

UNIVERSITÄT KARLSRUHE

Geant 4 Users' Tutorial at CERN
25-27 May 2005

Layout

II. Introduction
- Generalities
- Parameterisation Features

III. Fast Simulation Components of Geant4
- Fast Simlation Components Overview
- SimulationModel, Envelops
- SimulationManager, Ghost volumes
- Summary Picture of Fast Simulation Mechanism

IV. GFlash parametrisations
- What is Gflash, equations and parameters
- Some example results

V. Examples

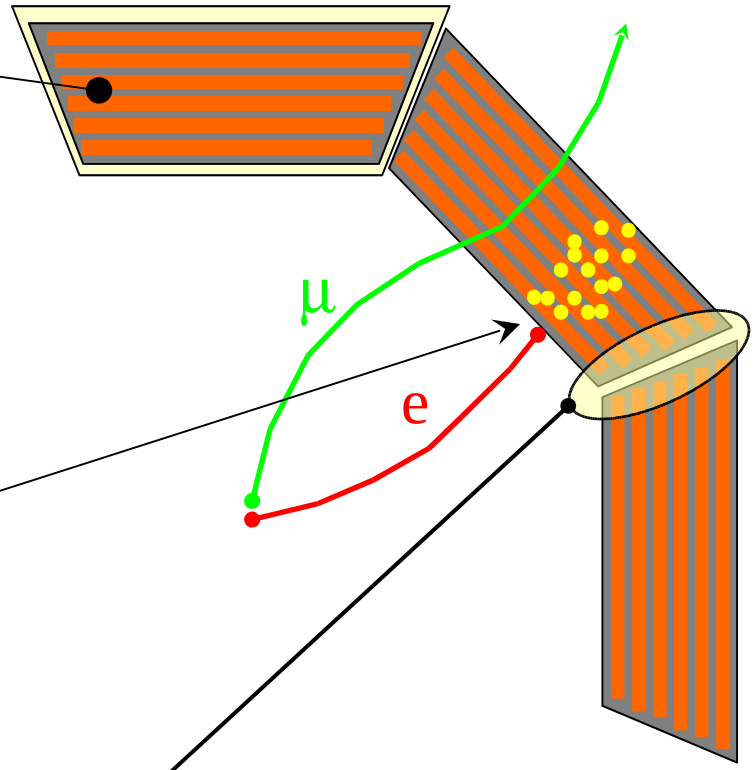
I. Introduction

Generalities

- Fast Simulation, also called parameterisation, is a shortcut to the tracking.
- Fast Simulation allows you to take over the tracking & to implement your own fast physics and detector response.
- The classical use case of fast simulation is the shower parameterisation where the typical several thousand steps per GeV computed by the tracking are replaced by a few ten of deposits per GeV.
- Parameterisations are generally experiment dependent.
- Geant4 provides a convenient framework.

Parameterisation features

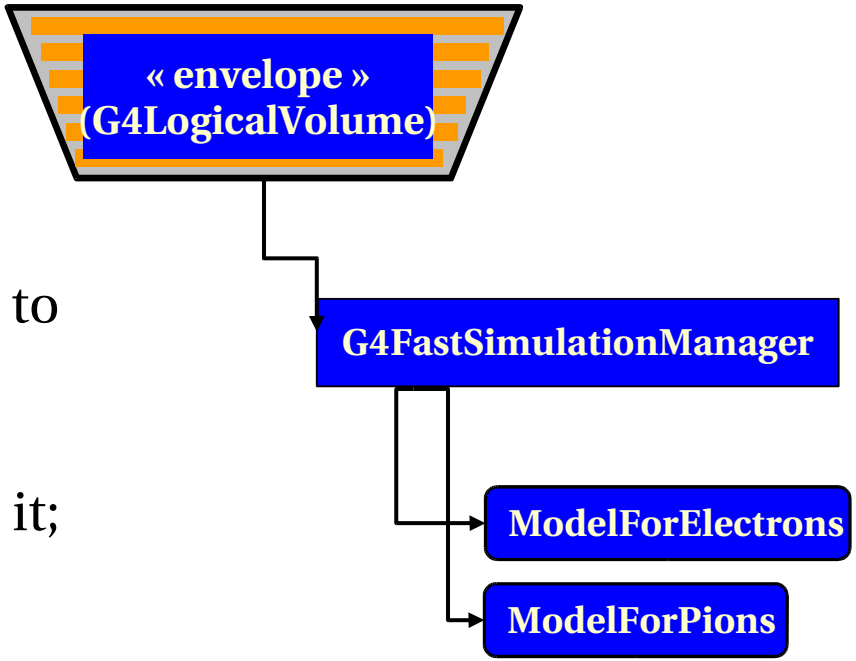
- Parameterisations take place in an *envelope*.
  This is typically the mother volume of a sub-system or of a large module of such a sub-system.
- Parameterisations are often *particle type* dependent and/or may apply only to some.
- They are often not applied in complicated regions.



II. Fast Simulation Components of GEANT4

Fast Simulation Components Overview

- In GEANT4 you implement parameterisations as models inheriting from the base class: ***G4VFastSimulationModel***
- Those models are bound to the envelope by a ***G4FastSimulationManager*** object, allowing several models to be bound to one envelope.
- The ***envelope*** is simply a G4LogicalVolume with a G4FastSimulationManager bound to it;
- All its [grand[…]]daughters will be sensitive to the parameterizations
- To activate the parameterisation you will have to add the ***G4FastSimulationManagerProcess*** to the list of processes of the particles.



G4VFastSimulationModel

Override the three following methods to create your model:

- **G4bool IsApplicable(const G4ParticleDefinition *)**
  - Must return 'true' for if your model applies to the given particle.
- **G4bool ModelTrigger(const G4FastTrack &)**
  - Must return 'true' if you want your model to take over the tracking at the current point.
  - The trigger decision is helped using the ***G4FastTrack*** which agregates the G4Track and envelope related informations.
- **void DoIt(const G4FastTrack &, G4FastStep &)**
  - This is your parameterisation properly said, where you will develop your parameterised shower for example. DoIt() is invoked if ModelTrigger() has previously returned 'true'.
  - To tell the tracking that you moved/killed/… the primary G4Track and/or produced secondaries during your parameterisation you have to fill the ***G4FastStep*** accordingly.

Envelope

- The envelope mechanism is completely automated.
- Envelope is not a special class but simply a ***G4LogicalVolume***.
  - Only an internal switch is turned on to make the volume being an envelope.
  - The ***G4FastSimulationManager*** makes this switch to be turned on.
- When a volume is an envelope all its daughters are recursively bound the to same ***G4FastSimulationManager*** object.
- The parameterisation code is thus accessible anywhere inside the whole envelope

G4FastSimulationManagerProcess

- The ***G4FastSimulationManagerProcess*** provides the ***interface*** between the tracking and the fast simulation

- It has to be set to all the particles to be parameterised by ***G4VFastSimulationModel***'s to make those models effective.
  - The process ordering is the following:

    [n-3] …
    [n-2] Multiple Scattering
    [n-1] G4FastSimulationManagerProcess
    [n] G4Transportation

  - It can be set as a discrete process

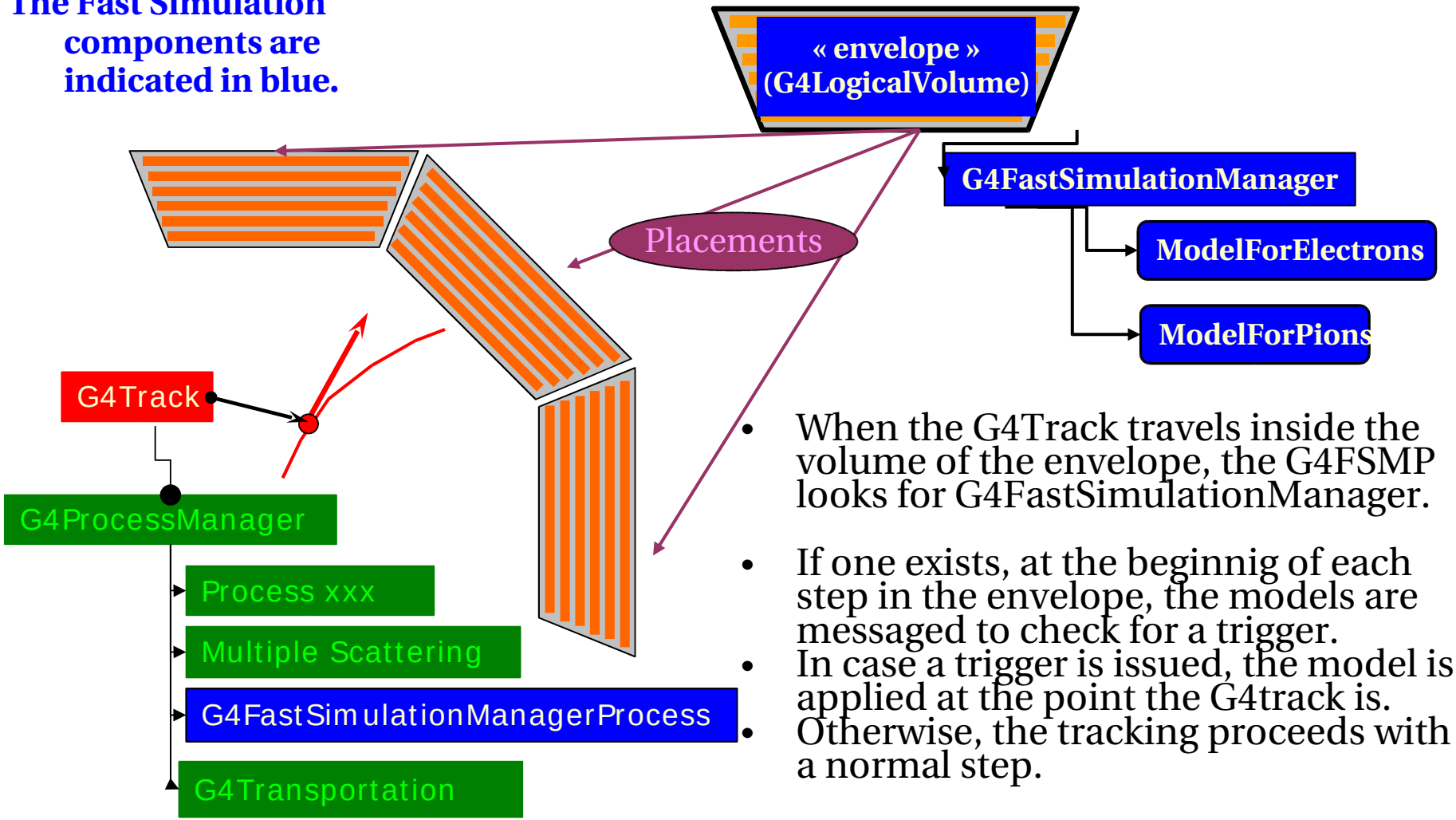
Ghost Volumes

- **Ghost volumes** allow to define envelopes independent to the volumes of the tracking geometry.
  - > For example, this allows to group together electromagnetic and hadronic calorimeters for hadron parameterization or to define envelopes for imported geometries which do not have a hierarchical structure.

- In addition, **Ghost volumes** can be sensitive to particle type, allowing to define envelops individually to particle types.
- **Ghost Volumes** of a given particle type are placed as a clone of the world volume for tracking (done automatically by ***G4GlobalFastSimulationManager)***
- The ***G4FastSimulationManagerProcess*** provides the additional navigation inside a ghost geometry. This special navigation is done transparently to the user.

Fast Simulation Components Overview

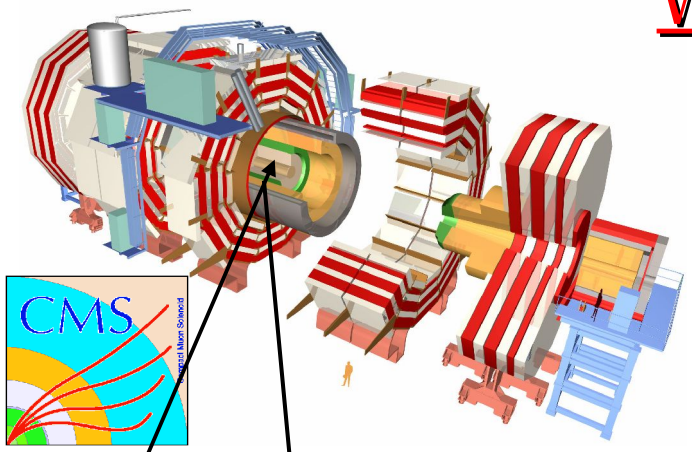
The Fast Simulation components are indicated in blue.



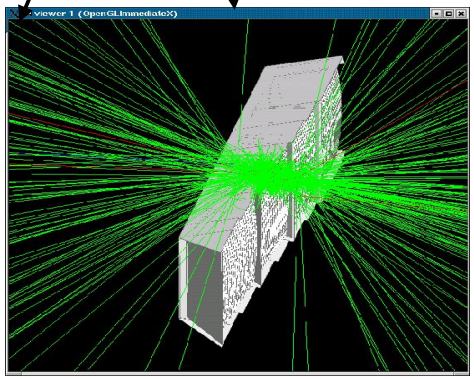
- When the G4Track travels inside the volume of the envelope, the G4FSMP looks for G4FastSimulationManager.
- If one exists, at the beginnig of each step in the envelope, the models are messaged to check for a trigger.
- In case a trigger is issued, the model is applied at the point the G4track is.
- Otherwise, the tracking proceeds with a normal step.

III. Gflash parametrisation in GEANT4

Why Gflash ?



Example:
CMS detector



Geant4 simulation of 50 GeV shower in CMS ECAL super module (PbWO4 crystals)

Problem:

Full simulation of LHC events is very time consuming (O (minutes/event)), especially because of simulating electromagnetic shower

Aim: Speed up full simulation in electromagnetic calorimeter (ECAL) without to sacrifice too much precision

A Solution:

GFlash package from H1, written in Fortran and working within the framework of Geant3 (Fortran)

•Provides a set of equations & parameters to describe electromagnetic shower profiles

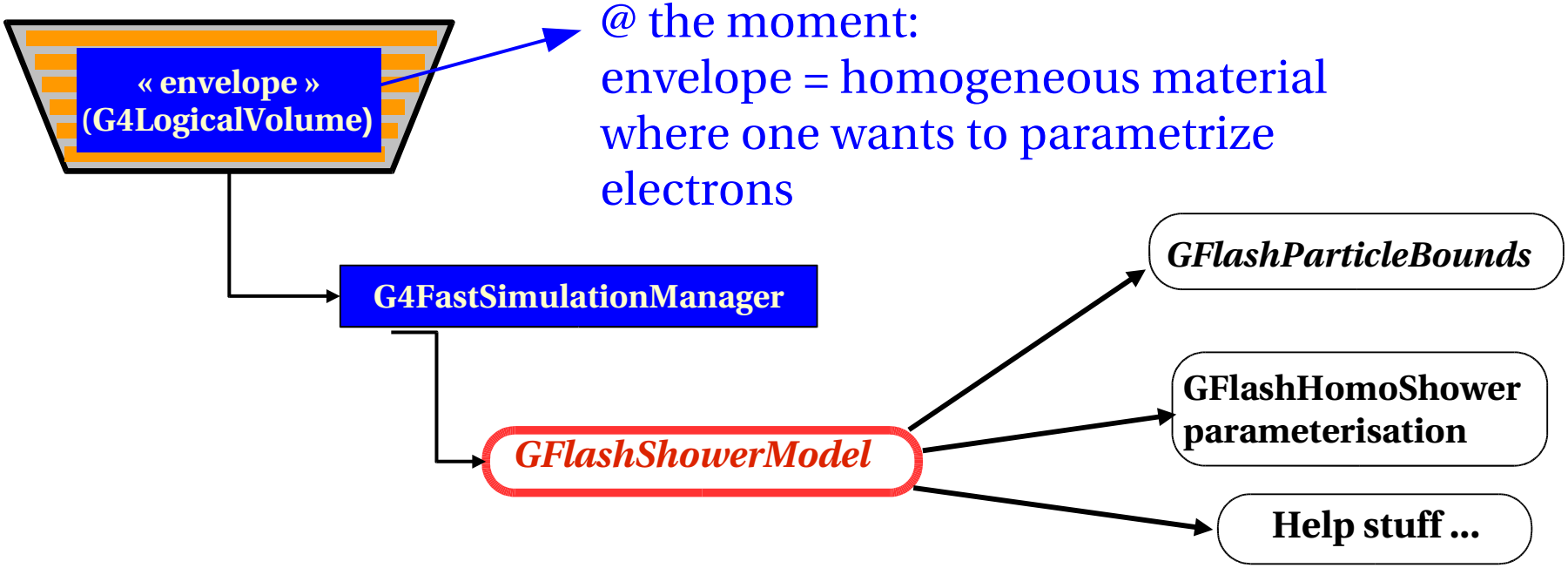
Gflash in Geant4

- **Implementation of GFlash (with original parameter)**

now available in Geant4.7.0 (within the fast shower framework)
(geant4/parametrisations/gflash in g4 repository)

- Example showing how to use gflash library included

(geant4/examples/advanced/gflash in g4 repository)



Gflash parametrisation – longitudinal profile

Spatial energy distribution of electromagnetic showers is given by three probability density functions:
(hep-ex/0001020, Grindhammer & Peters)

$$dE(\vec{r}) = E\, f(t)\, dt\, f(r)\, dr\, f(\phi)\, d\phi$$

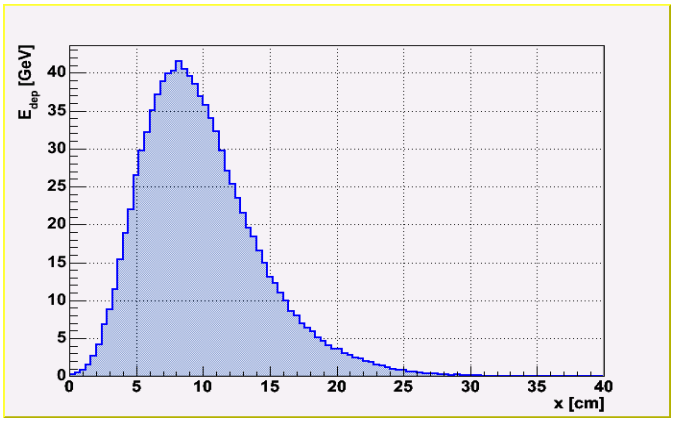
In f the energy is assumed to be distributed uniformly:

$$f(\phi) = 1/2\pi$$

Longitudinal Profile parametrized by gamma function with parameters and :

$$< \frac{1}{E}\frac{dE}{dt} > = f(t) = \frac{(\beta t)^{(\alpha-1)}\, \beta\, e^{(-\beta t)}}{\Gamma(\alpha)}$$

t[Xo]:
longitudinal coordinate



Gflash parametrisation – radial profile

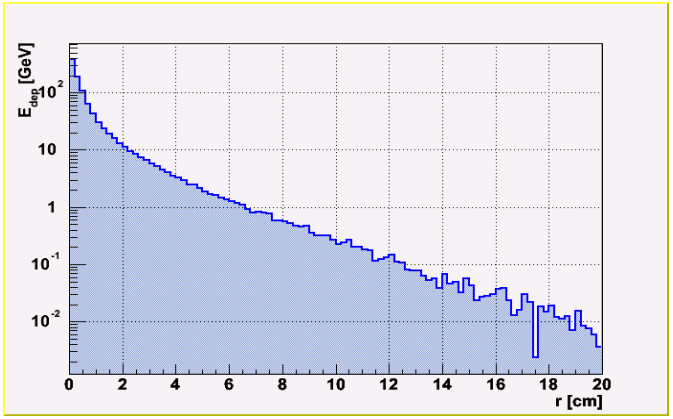
The average radial energy profile can be described by (with $0 \le p \le 1$) :

$$< \frac{1}{dE(t)} \frac{dE(r,t)}{dr} > = f(r) = p \frac{2 r R_{C(ore)}^2}{(r^2 + R_{C(ore)}^2)^2} + (1-p) \frac{2 r R_{T(ail)}^2}{(r^2 + R_{T(ail)}^2)^2}$$

r := Transverse size of shower, measured in Moliere radius (R_M)

R_C (R_T) is the median of the core (tail) component

R_C (R_T) depends on the shower depth



Gflash speed up

CPU time of full Geant 4.7.0 simulation and GFLASH shower parameterisation for a single electron (Pentium III @ 1Ghz) in an PbWO4 cube:

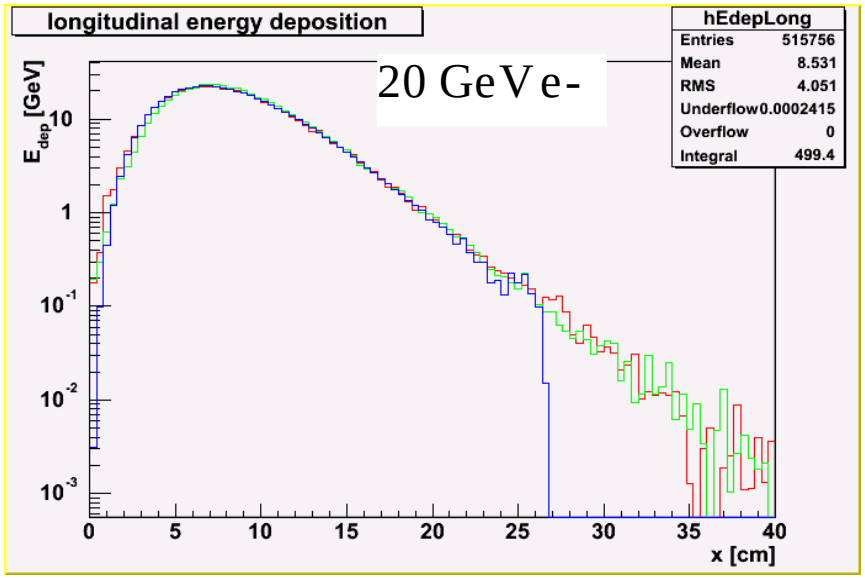
Electron Energy	Time / event full simulaton	Time / event GFLASH	Speed-up Factor
1 GeV	0.10	0.01	16
5 GeV	0.46	0.01	48
10 GeV	0.92	0.01	67
50 GeV	4.60	0.04	103
100 Gev	9.37	0.08	117
500 GeV	46.50	0.31	149
1000 GeV	91.75	0.57	162

Results quite promising :-);

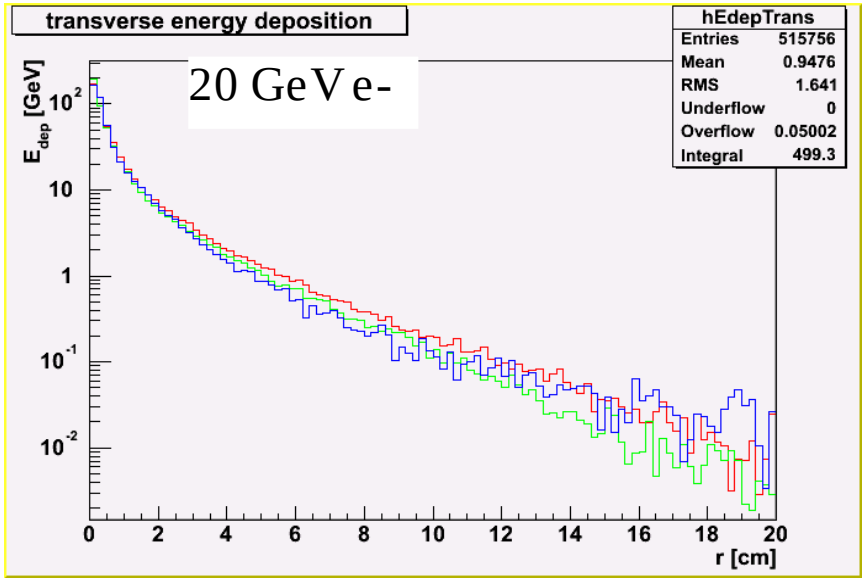
Gflash tuning

-> For G(4)flash retuning can be necessary (radial profile).
For experiment specific geometries retuning often necessary anyway, therefore possible to pass user-tuned parameters to Gflash:

*GFlashHomoShowerParamterisation(G4Material * aMat,*
*GVFlashHomoShowerTuning * aPar)*



Geant 4.7.0 / Geant3 / Gflash



Geant 4.7.0 / Geant3 / Gflash

IV. Examples

How to use the fast simulation

- Example of use of Fast Simulation can be found in **example/novice/N05**
- This includes examples with non-ghost and ghost envelopes

How to use Gflash

Example of Use (geant4/examples/advanced/gflash)

In DetectorConstruction: Assign Gflash parameterisation to envelope

```
theFastShowerModel = new GflashShowerModel("GflashShowerModel",
                                           calo_log);
theParametrisation = new GflashHomoShowerParamterisation
                         (matManager->getMaterial(mat));
theFastShowerModel->SetParametrisation(*theParametrisation);
theFastShowerModel->SetParticleBounds(*theParticleBounds);
theFastShowerModel->SetHitMaker(*theHMaker);
```

- In Physics List: Add FastSimulationManagerProcess

```
G4FastSimulationManagerProcess* theFastSimulationManagerProcess
= new G4FastSimulationManagerProcess();
{
(Loop over particles and add to Process Manager)
}
```

How to use Gflash(2)

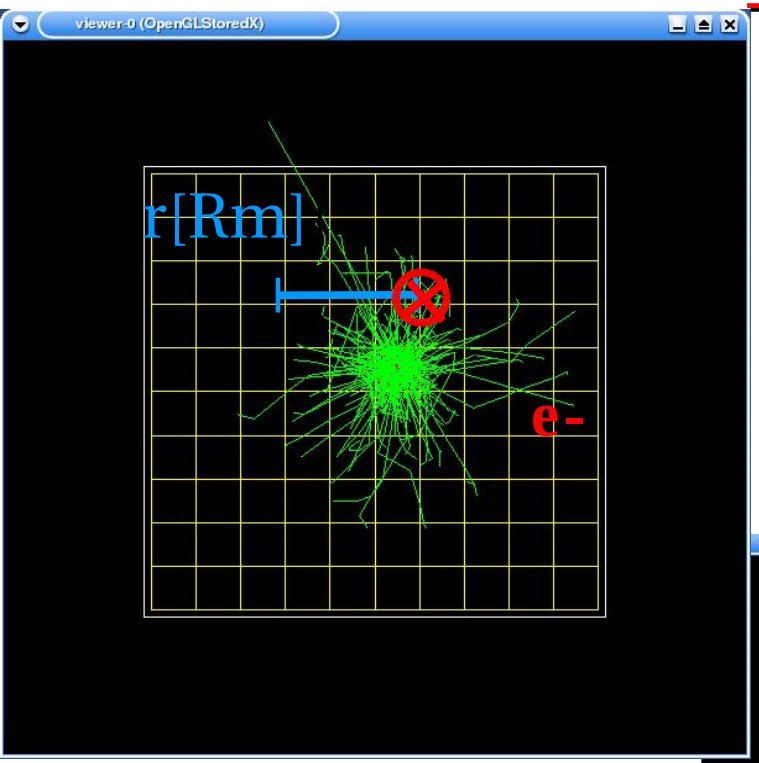
•In the interface to your sensitive detector: inherit from ***G4VGFlashSensitiveDetector***

class YourSensitiveDetector: public G4VSensitiveDetector ,
public G4VGFlashSensitiveDetector {
...
}

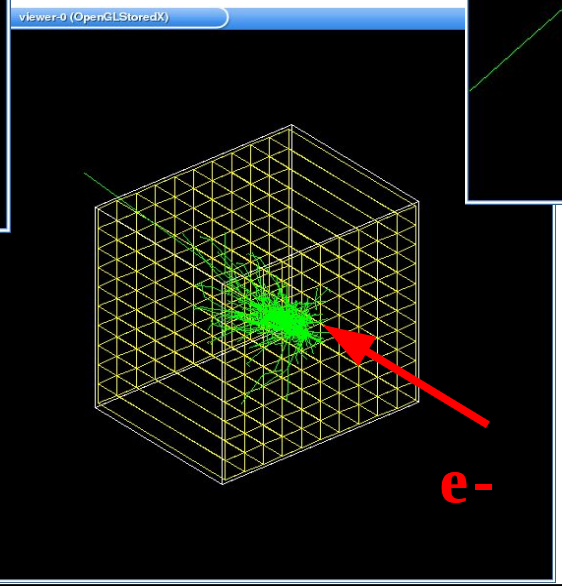
• In the interface to your sensitive detector:
implement seperate interface for Gflash hit processing:

G4bool ProcessHits(G4Step,G4TouchableHistory*);*
*G4bool ProcessHits(G4GFlashSpot*aSpot,G4TouchableHistory*);*

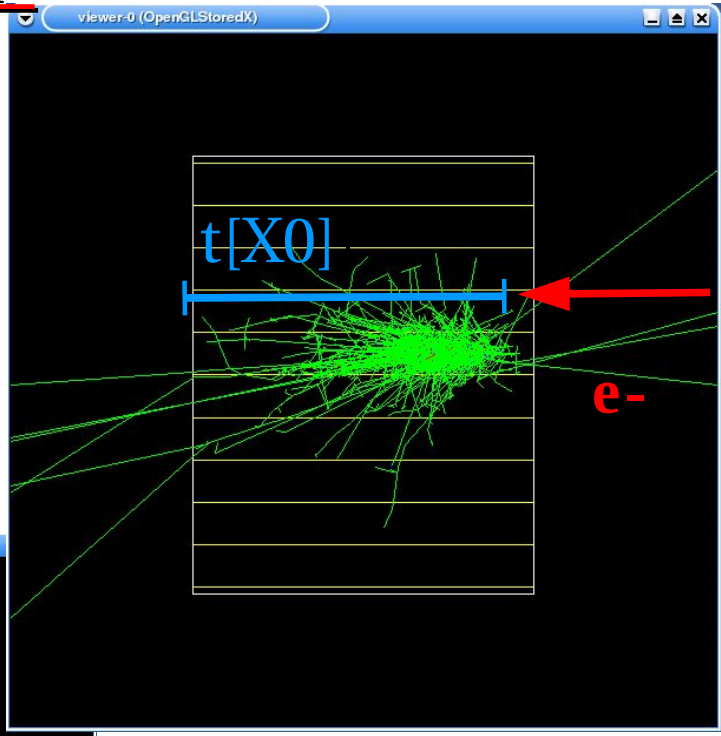
Visualisation



Radial profile



CMS ECAL model
(G4 example)



Longitudinal profile