Introduction to Geant4

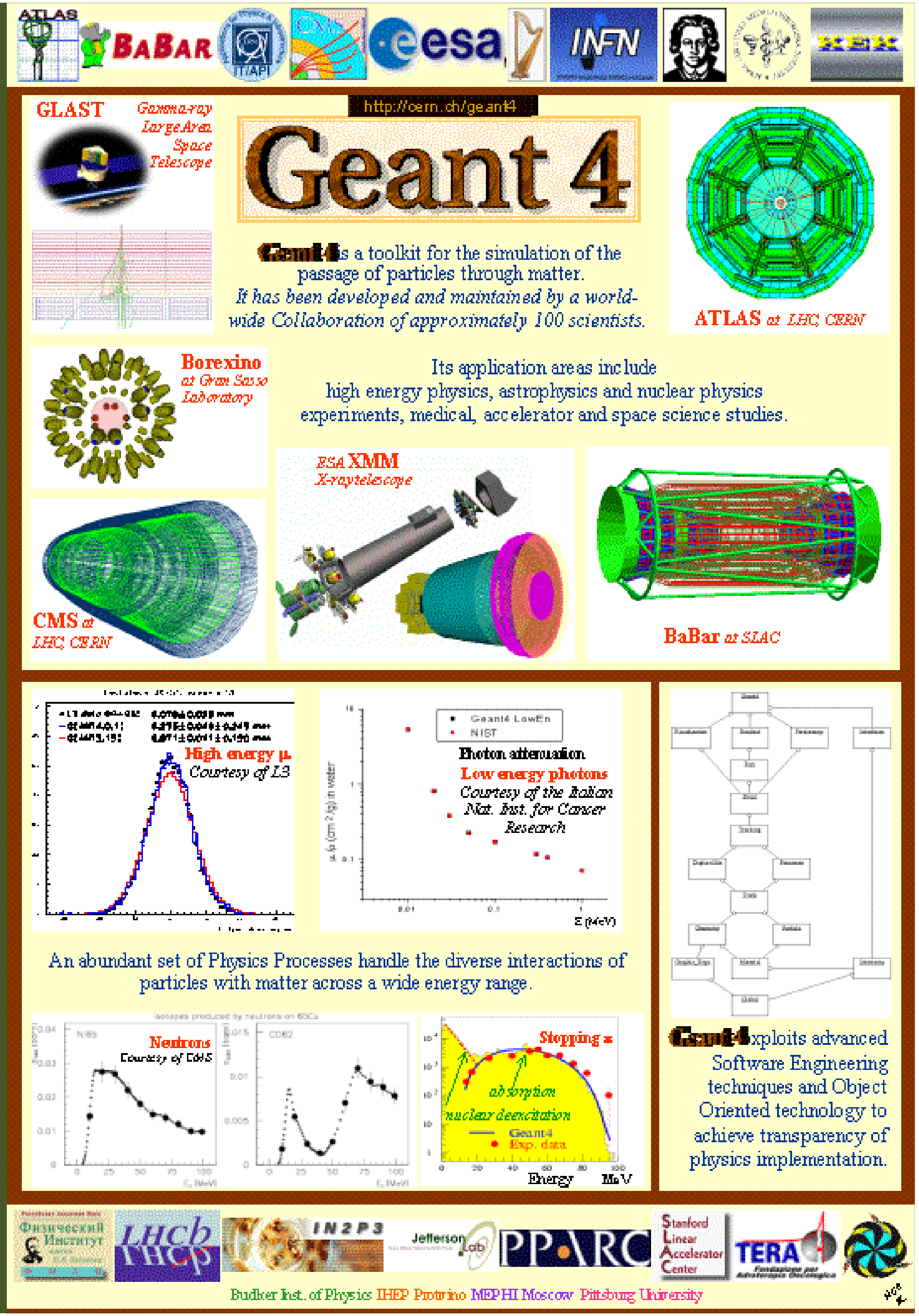
John Apostolakis (CERN)

Borrowing especially from presentations of M. Asai (SLAC)

Geant4 Tutorial Course
At 10th Geant4 Conference and Workshop
November 3-5 2005, Bordeaux

Contents

- General introduction and brief history
- A few user applications
- Geant4 kernel
  - Basic concepts and kernel structure
  - User classes



General introduction and brief history

What is Geant4?

- Geant4 grew from the needs for a complete toolkit to simulate the interaction of particles with matter in setups of arbitrary complexity
  - In HEP it is the successor of GEANT3, the leading toolkit for detector simulation
    - designed and engineered to address detailed simulation for the next decade(s)
  - A variety of additional requirements came from heavy ion physics, cosmic ray physics, astrophysics, space science and medical applications.
- Configurability and flexibility are key attributes, designed-in from the start
  - Choice in physics modeling, in building setups, in
- Geant4 has been of the first successful attempt to re-design a major package of HEP software for the next generation of experiments using an Object-Oriented environment.
  - G4 is not only for HEP but goes well beyond that.

Flexibility of Geant4

- In order to meet wide variety of requirements from various application fields, a large degree of functionality and flexibility are provided.
- Choice of physics models, cross-sections and physics-list ‘configurations’
  - Models with different strengths for most interactions
  - With suggested physics lists for use cases, which a user can utilize or tailor
- Geant4 offers many types of geometrical descriptions to describe realistic geometries, small or large
  - A variety of shapes: simple (CSG), BREP, or combinatorial (Boolean)
  - Possibilities to place individual volume copies, group, parameterize, or reflect volumes
- Everything is open to the user
  - From the choice of physics to the choice of visualization and analysis solutions

Physics in Geant4

- It is rather unrealistic to develop a uniform physics model to cover wide variety of particles and/or wide energy range.
- Much wider coverage of physics comes from mixture of theory-driven, parameterized, and empirical formulae. Thanks to polymorphism mechanism, both cross-sections and models (final state generation) can be combined in arbitrary manners into one particular process.
- Geant4 offers
  - Electromagnetic (EM) processes
  - Hadronic processes
  - Photo-hadron and lepto-hadron processes
  - Optical photon processes
  - Decay processes
  - Shower parameterization
  - Event biasing techniques
  - And you can plug-in more

Physics in Geant4

- Each cross-section table or physics model (final state generation) has its own applicable energy range. Combining more than one tables / models, one physics process can have enough coverage of energy range for wide variety of simulation applications.
- Geant4 provides sets of alternative physics models so that the user can freely choose appropriate models according to the type of his/her application.
- Several individual universities / physicists groups are contributing their physics models to Geant4. Given the modular structure of Geant4, developers of each physics model are well recognized and credited.

Optimization techniques

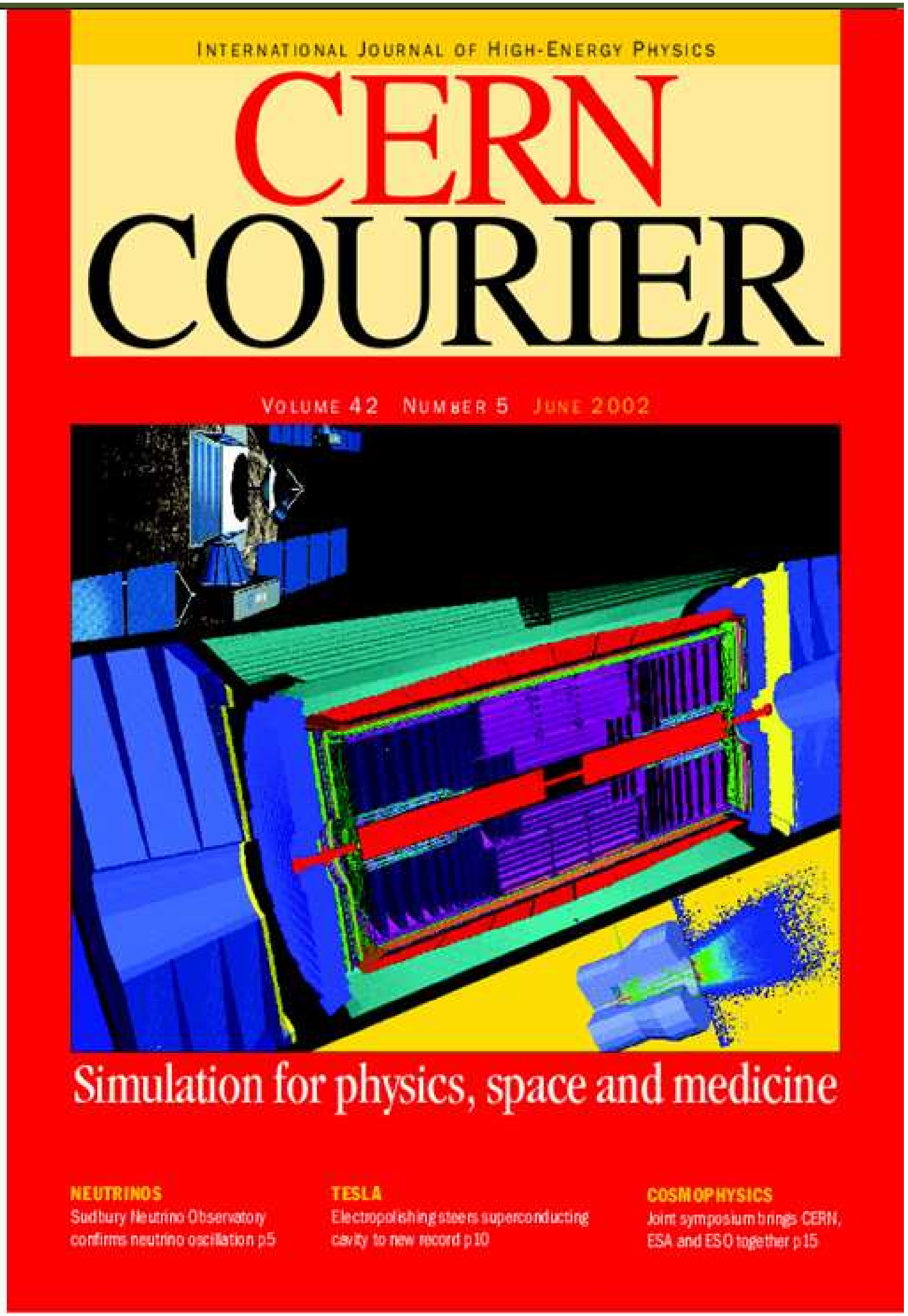
- Production thresholds
  - Typically for processes with infrared divergences
  - Facility for enforcing uniform thresholds across all processes
- Using the process interface additional capabilities
  - Fast simulation
    - Framework for user parameterisations
    - New parameterizations of showers
      - in homogeneous and (soon) sampling calorimeters
  - Event biasing techniques
    - Geometrical (importance) biasing
    - Simple leading particle
  - And you can plug-in more

Technology transfer

Particle physics software aids space and medicine

Geant4 is a showcase example of technology transfer from particle physics to other fields such as space and medical science [...].

CERN Courier, June 2002



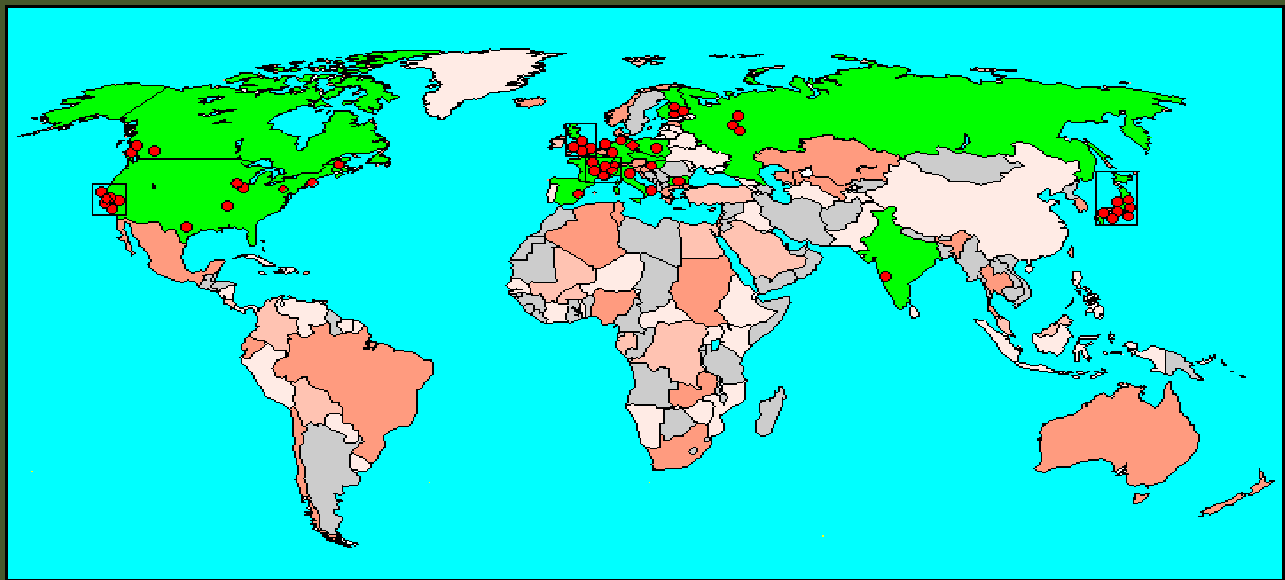
Geant 4

Geant4 – Early and latest history

- Dec ’94 - Project start
- Jul ’98 - First beta release
- Dec ’98 - First Geant4 public release
- …
- Dec ’03 - Geant4 6.0 release
- Jun ’04 - Geant4 6.2 release
- Dec 17th, 2004 - Geant4 7.0 release
    - Feb 26th, ’05 - Geant4 7.0-patch01 release
- Jun 30th, 2005 – Geant4 7.1 release
- We currently provide two to three public releases, and beta releases every year.

Geant4 Collaboration

HARP



Lebedev

Univ. Barcelona

PPARC

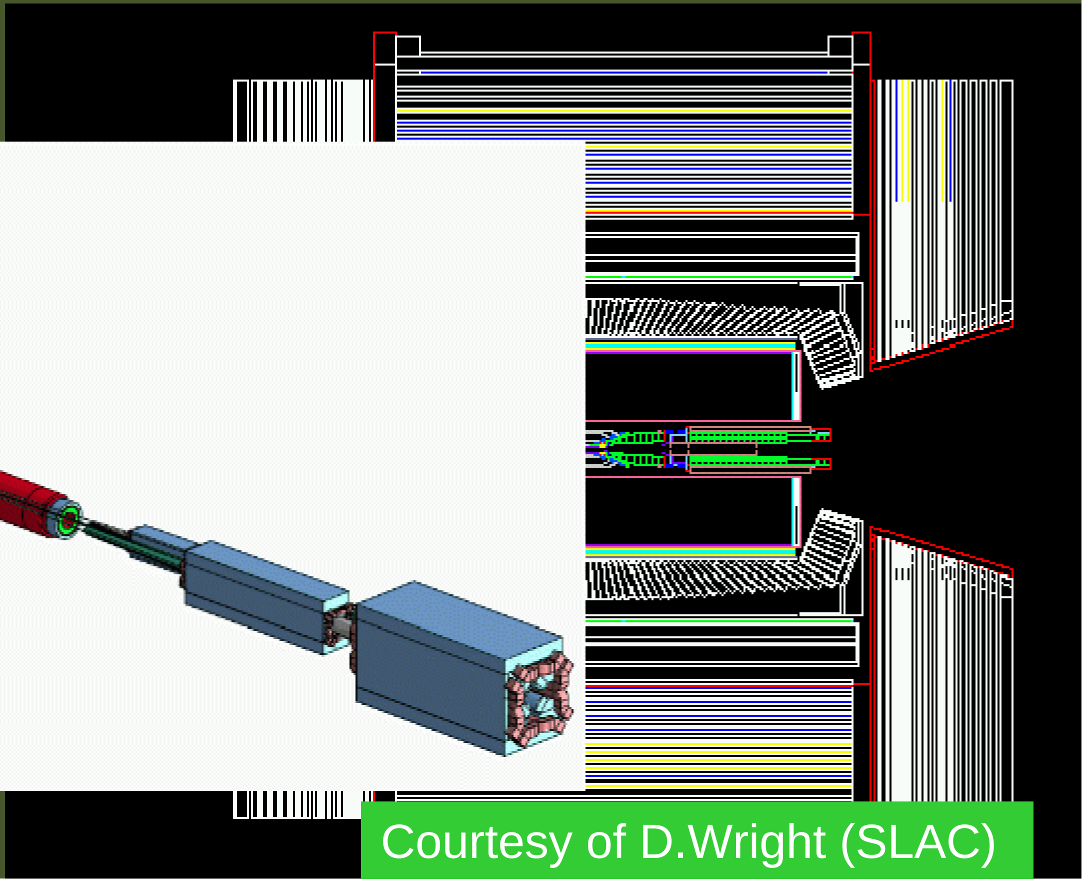
Helsinki Inst. Ph.

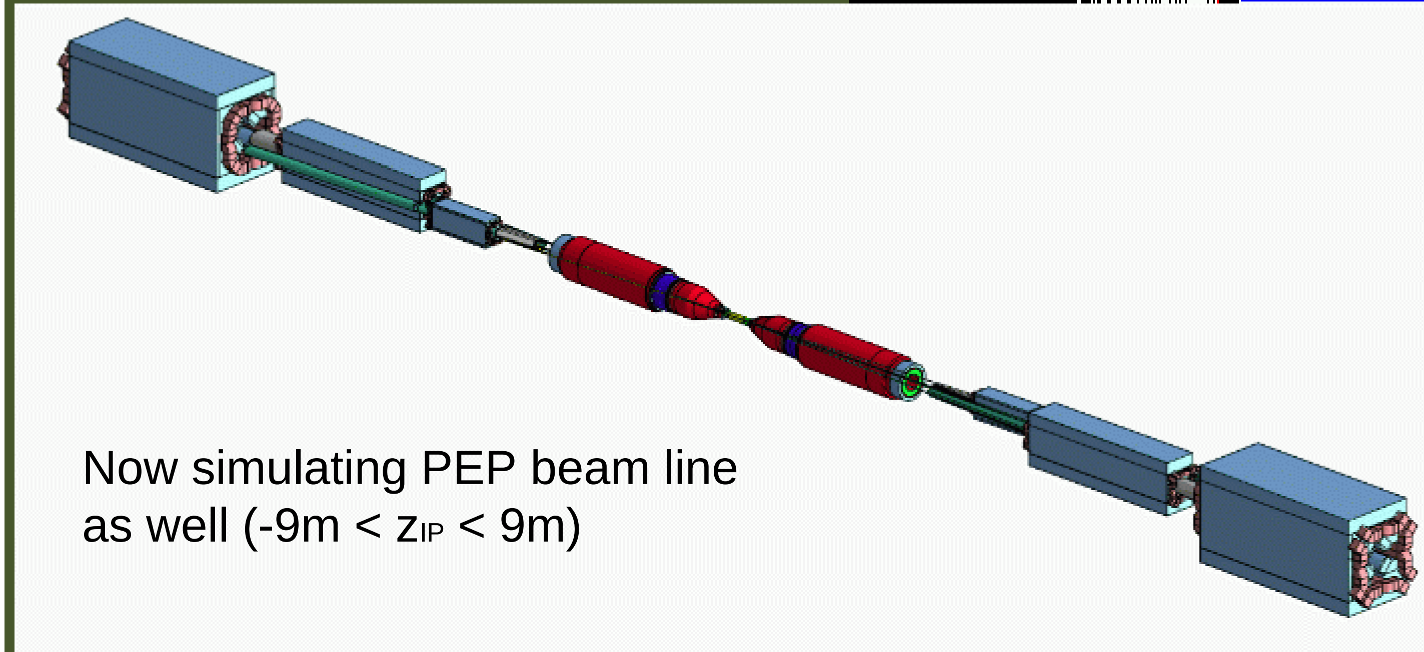
Collaborators also from non-member institutions, including
Budker Inst. of Physics
IHEP Protvino
MEPHI Moscow
Pittsburg University

Highlights of Users Applications

BaBar

- BaBar at SLAC is the pioneer experiment in HEP in use of Geant4
  - Started in 2000
  - Simulated $5*10^9$ events so far
  - Produced at 20 sites in North America and Europe
  - Current average production rate 6.1×10^7 events/week





Now simulating PEP beam line as well ($-9m < z_{IP} < 9m$)

Courtesy of D.Wright (SLAC)

Geant 4

Geant4 at the LHC Today

Now Geant4 has become the standard simulation for ATLAS, LHCB, and CMS

	ATLAS	CMS	LHCb
Transition to Geant4 (G3 stopped)	DC02 '04	Nov '03	May '04
Produced # of events in DC	12 M	40 M	80 M
CPU time (sec)/ event (2.8 Ghz)	600 ($pp \rightarrow Z \rightarrow ee$) 700 (SUSY)	200 (QCD jets) 60 (min bias)	22-65
Memory used	400 Mb	220 Mb	220 Mb
# of placed volumes	5 M	1.2 M	18 M

No memory leaks!!

$\Rightarrow$ Observations:

- Geant4 in production is running now very stable/very few problems (~ 10^{-5})
- Transition to Geant4 has been a very smooth process for all experiments

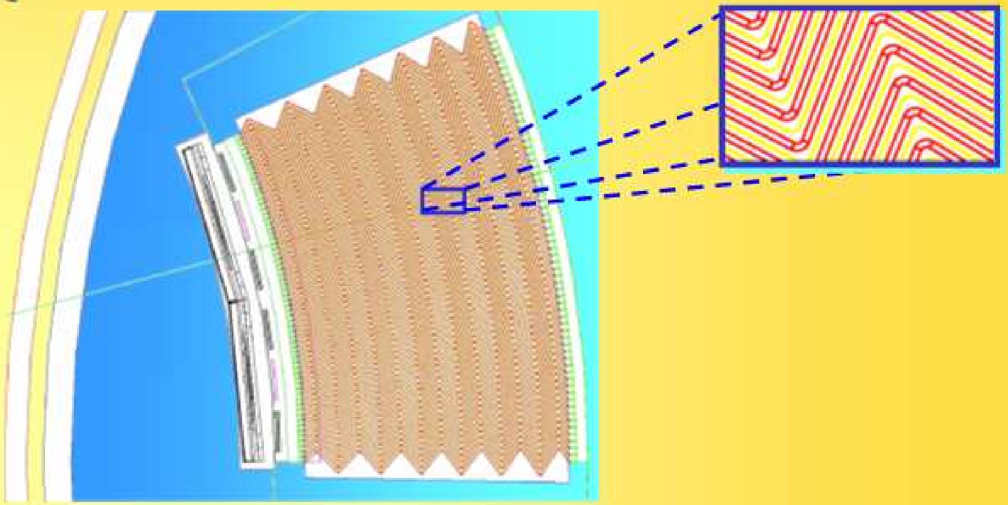
As of December 2004

Peter Loch
University of Arizona
Tucson, Arizona 85721

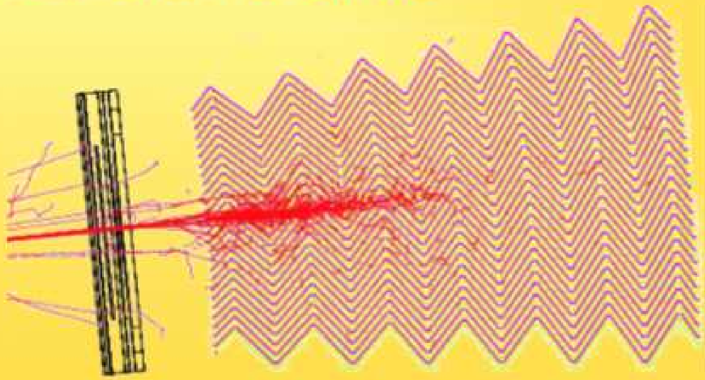
Geant4 Setups (2)

- Forward Calorimeter (FCal) Testbeam Setup

- Electromagnetic Barrel Accordion Calorimeter

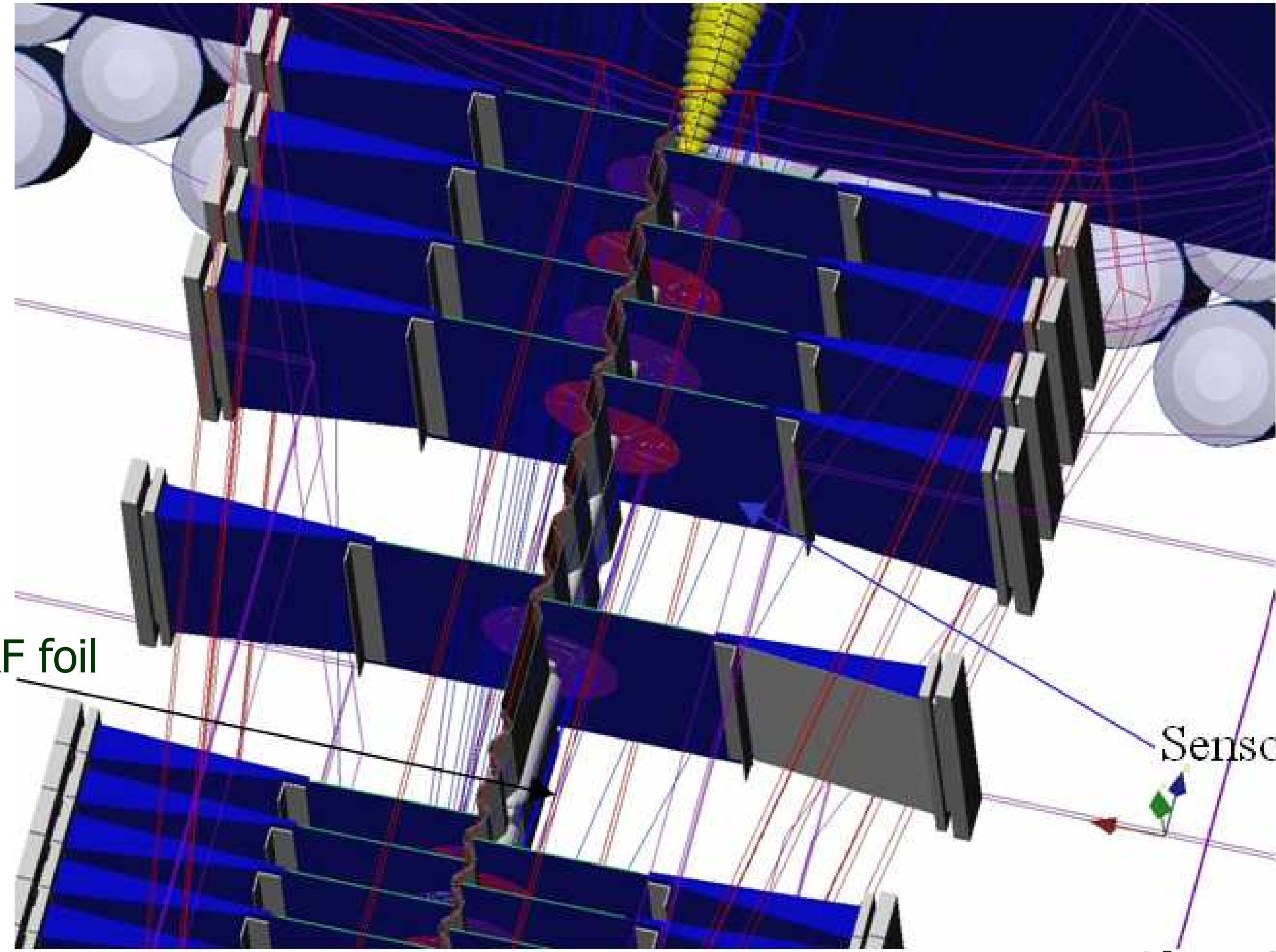


- 10 GeV Electron Shower



5 million Volumes
Geant4 in production: early 2004

Geant4 at the LHC Today



LHCb Vertex Locator description

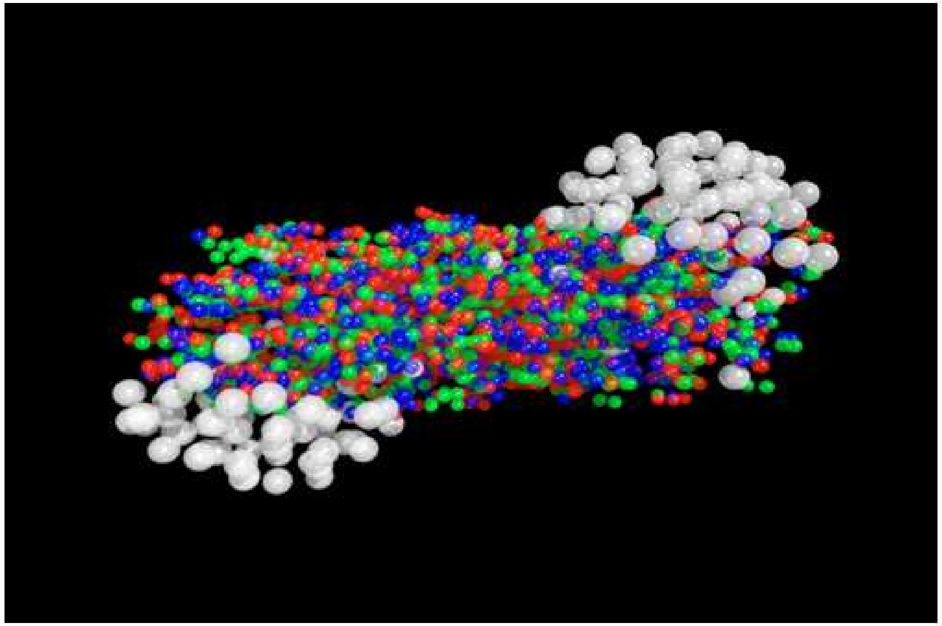
Complicated geometry
Details are very important

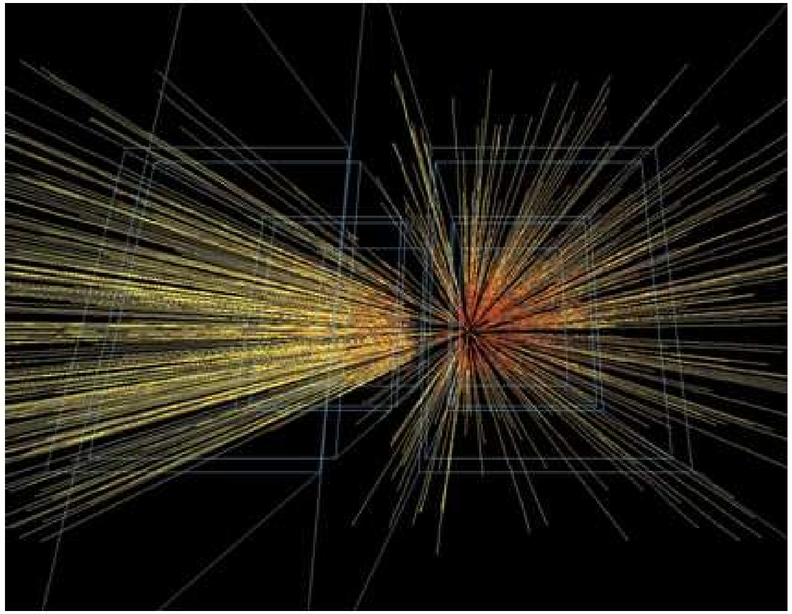
Geant4 can handle it!!

LHCb: 18 Million volumes
G4 in production: May 2004

Pushing G4 to the limits: Heavy Ions

Courtesy CMS Collaboration





Events with > 50000 particles/event in detector acceptance

1.2 million volumes

Production use starting Nov 2003

Geant 4

Heavy-Ion Collisions

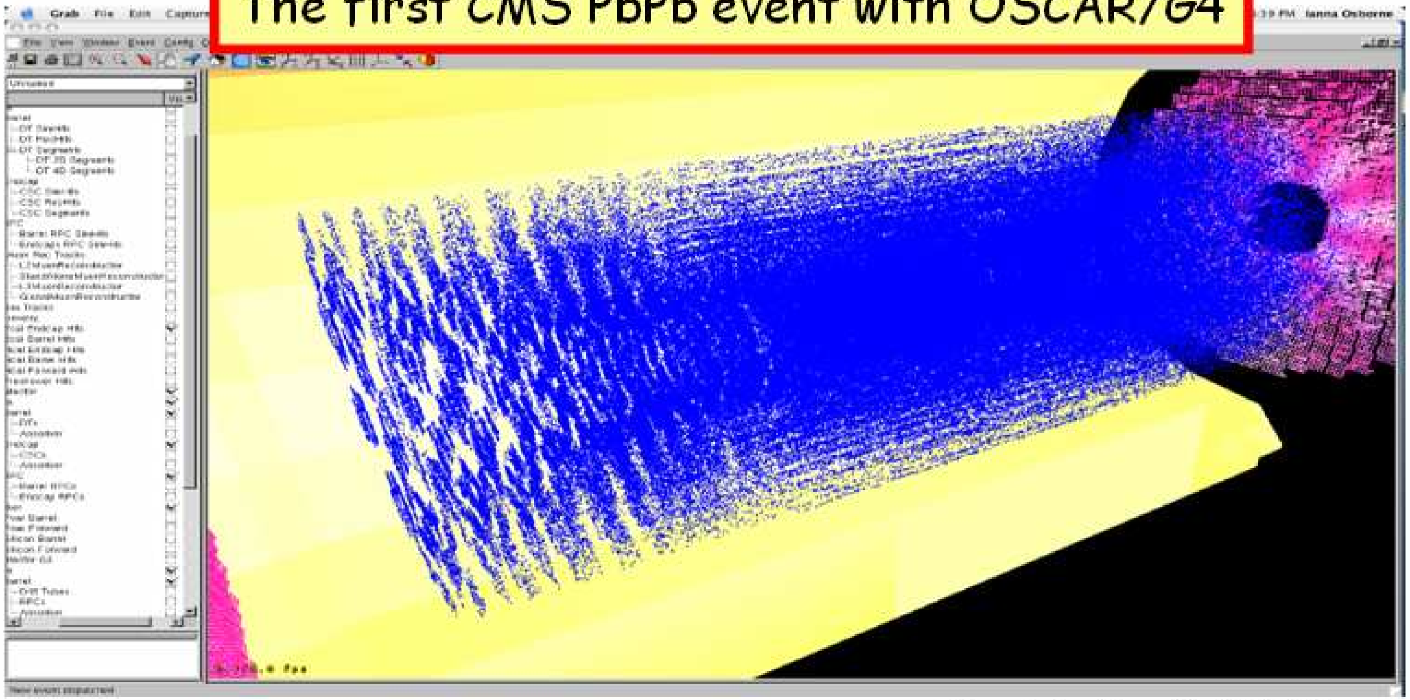
OSCAR/Geant4 can run full heavy ion events.

- Timing is good/Memory > 500 Mbyte (2GB memory machines used)
- Have now run > 100 events without problems

~ Timing for the first event with 55K generator tracks

Program	CPU (2.8GHz) (min)
CMSIM	230
OSCAR 2_4_5	320
OSCAR 3_4_0	180

The first CMS PbPb event with OSCAR/G4



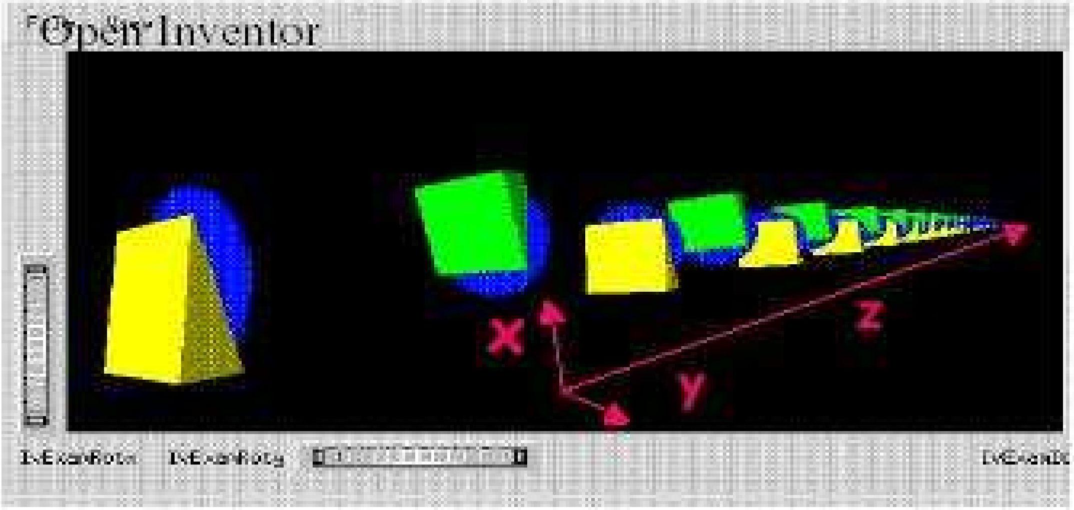
Geant4 for beam transportation

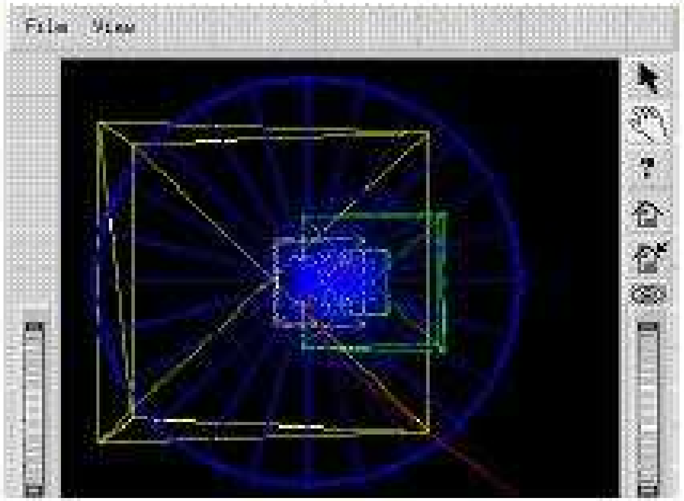
Example: Helical Channel

Published in proc. of PAC 2001 (Fermilab-Conf-01-182-T)

72 m long solenoidal + dipole field with wedge absorbers and thin cavities

$$B_{xy} = B_T \cos,\sin\left(\frac{2p}{L} z\right) \qquad B_z = B_0$$



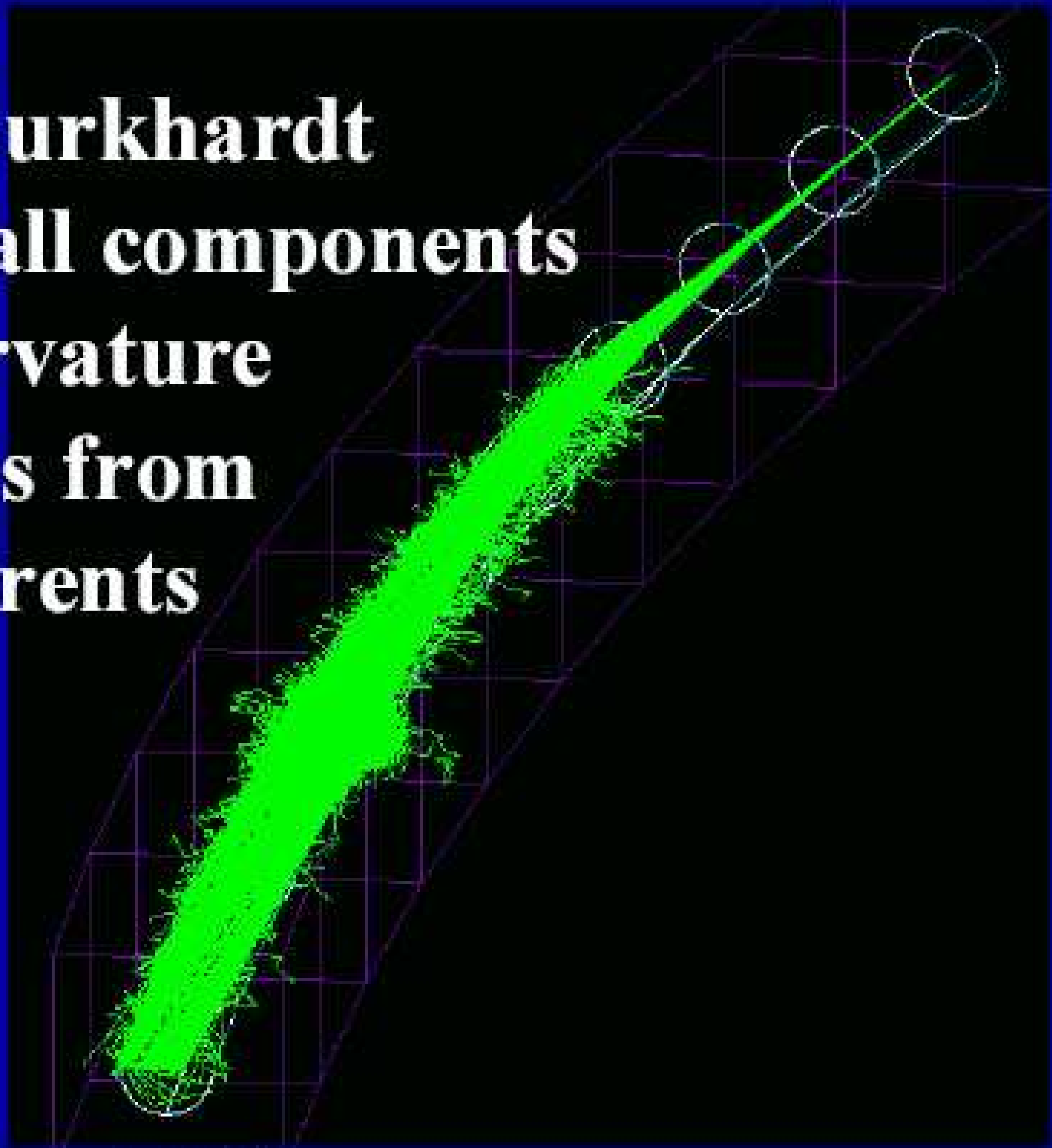


Other simulations:

- Alternate Solenoid Channel (sFoFo), published in proceedings of PAC2001 and Feasibility Study II for a Neutrino Factory at BNL (2001)
- Bent Solenoid Channel, presented at Emittance Exchange Workshop, BNL 2000
- Low Frequency r.f. Cooling Channel, presented at International Cooling Experiment Workship, CERN 2001
- Cooling Experiment (MICE) Simulation (in progress)

Courtesy of V.Daniel Elvira (FNAL, 2002)

Synchrotron Radiation



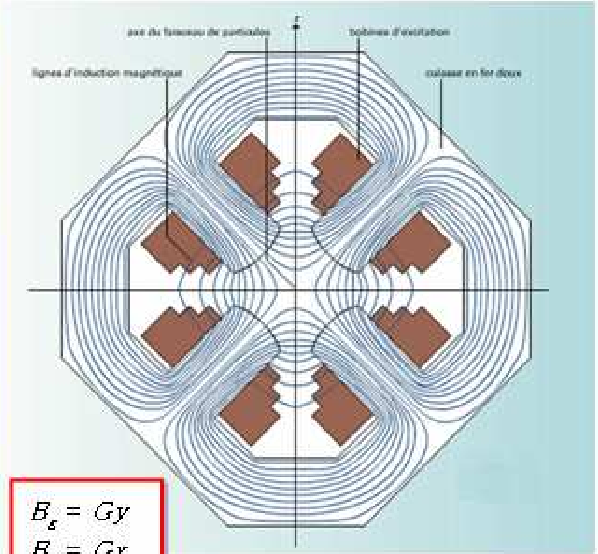
Generator of H. Burkhardt
Implemented for all components
Based on local curvature
Individual photons from individual parents

Courtesy of G.Blair (CERN)

Ray tracing in perfect quadrupoles

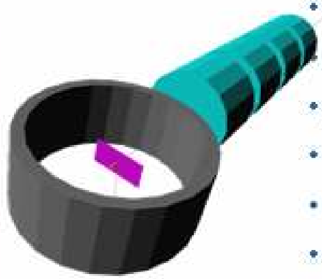
In our microbeam line, four quadrupoles to focus the beam

A pure quadrupole field



$B_x = Gy$
$B_y = Gx$
$B_z = 0$

G is the field gradient



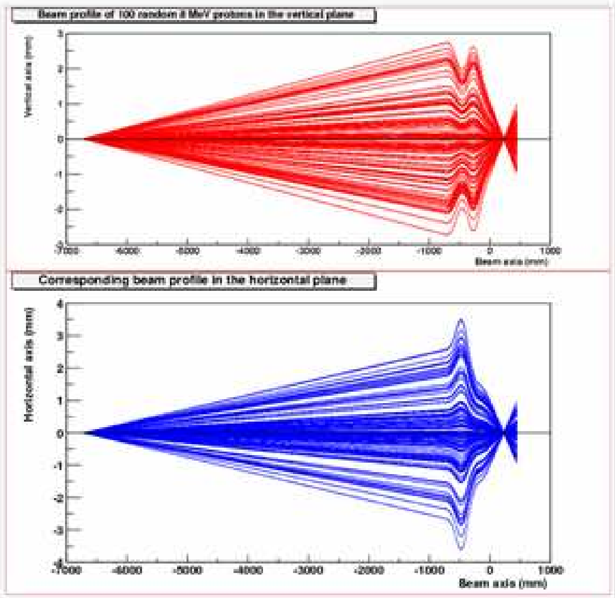
- Focus$_1$ Defocus$_2$ F$_3$ D$_4$ "Russian" configuration
- Quad length = 15 cm, gap radius = 1 cm, distance between quads = 4 cm
- $G_1 = -G_4 = 5.8928$ T/m and $G_2 = -G_3 = -14.6466$ T/m
- proton or alpha beam
- gaussian T = 8 MeV or 2 MeV (standard deviation is 4 keV)
- angular divergence : 0.5 m rad
- gaussian position distribution of **10 µm FWMH**

GEANT4.1+P01 predicts :

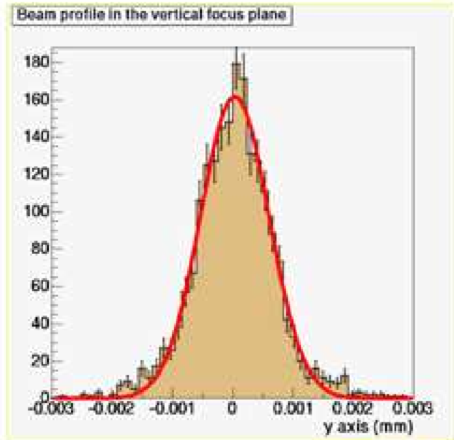
- Focus plane position : **230.15 ± 0.05 mm**
- FWMH of beam in image plane : **1.3 µm**
- **same prediction as the OXRAY code :**

  focus plane position : 230.1 ± 0.1 mm

  FWMH = 1.1 µm



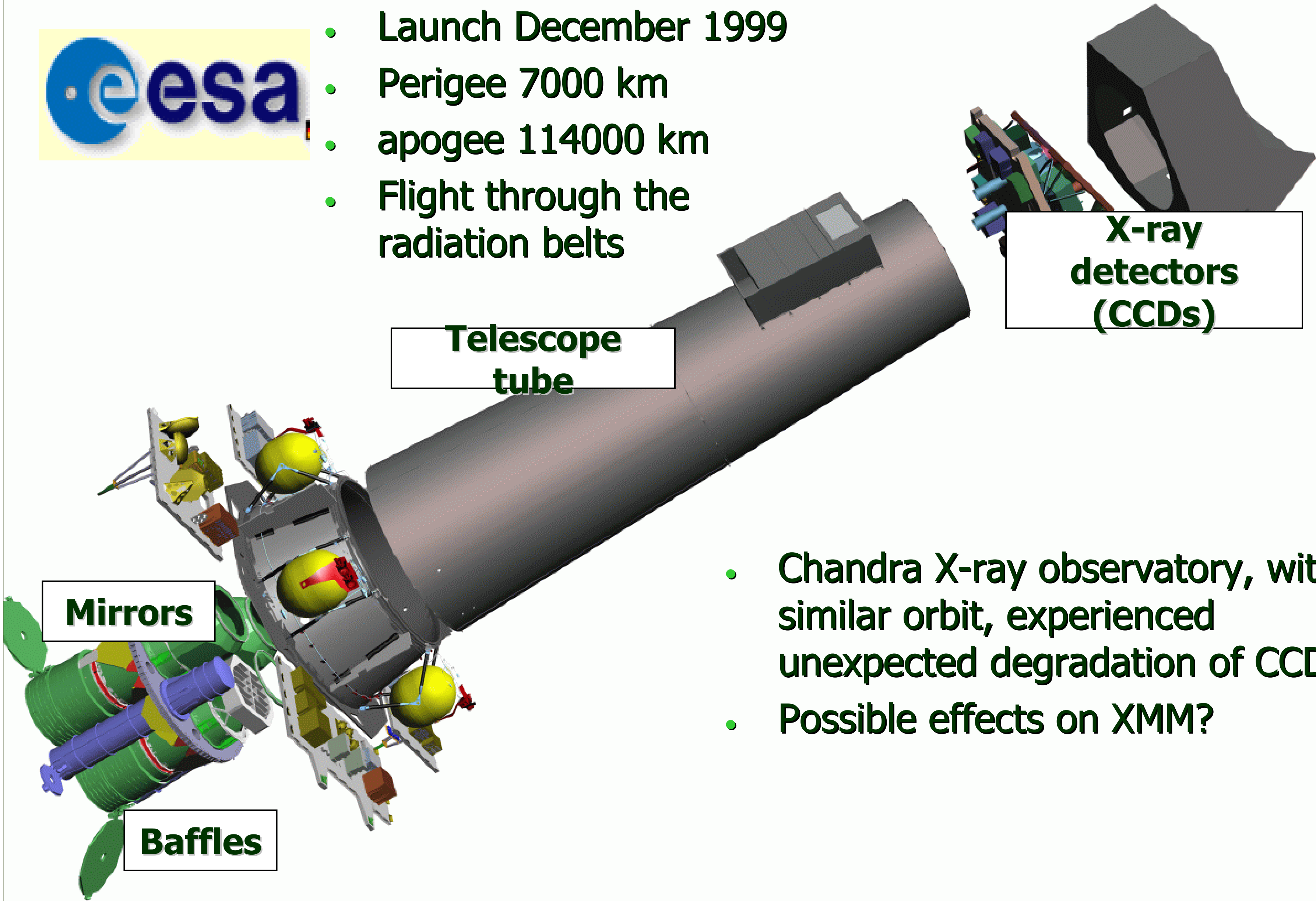
GEANT4 4.1+P01



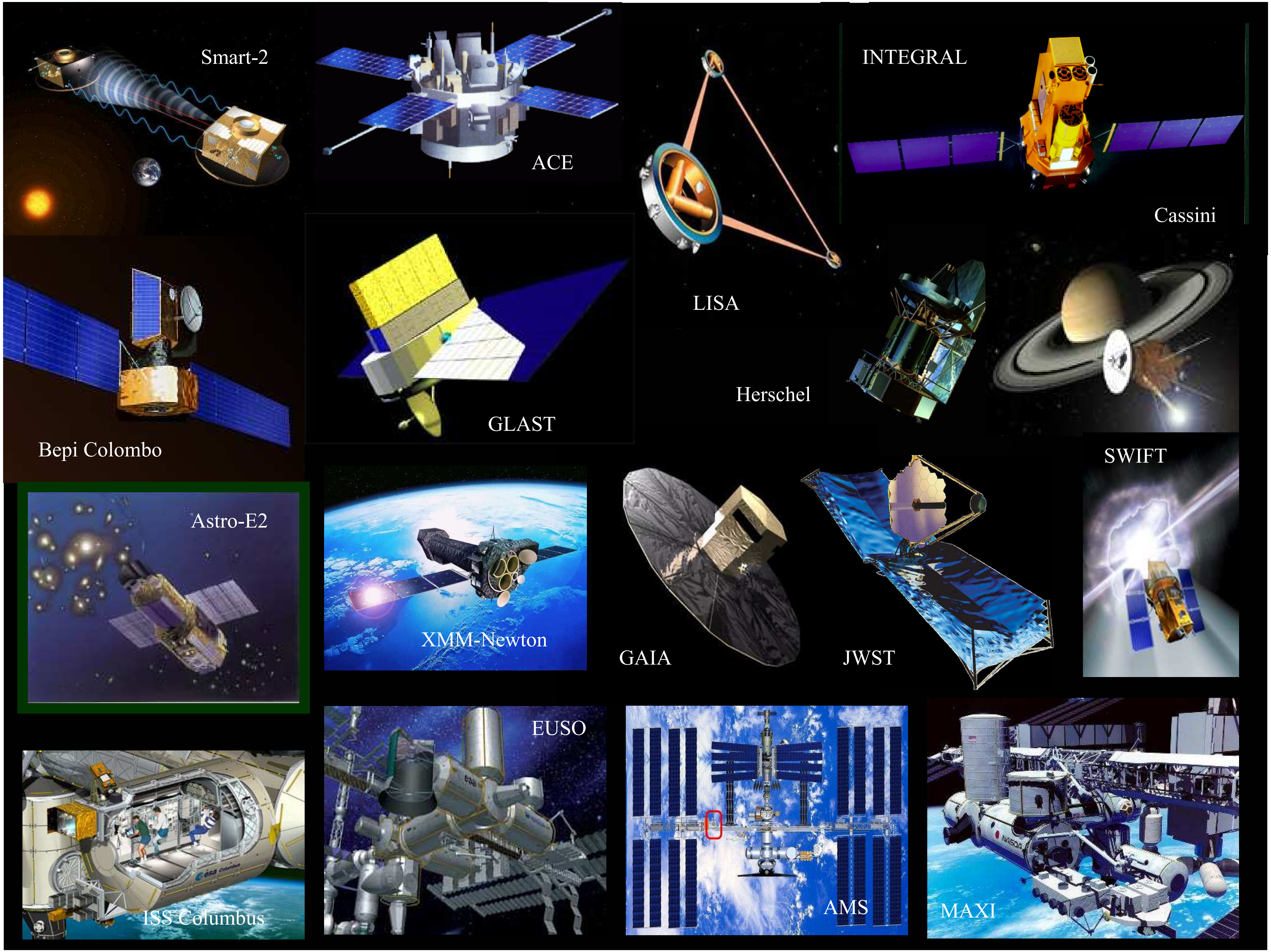
GEANT4 4.1+P01

X-ray Multi-Mirror mission (XMM)

- Launch December 1999
- Perigee 7000 km
- apogee 114000 km
- Flight through the radiation belts



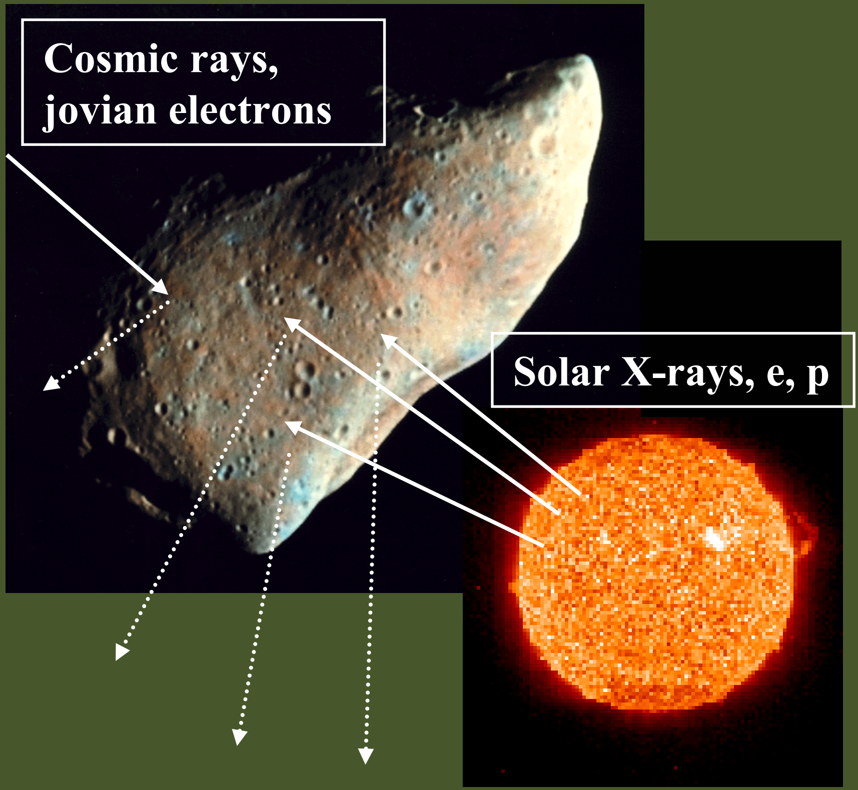
- Chandra X-ray observatory, with similar orbit, experienced unexpected degradation of CCDs
- Possible effects on XMM?


Smart-2
ACE
INTEGRAL
Cassini
LISA
Bepi Colombo
GLAST
Herschel
SWIFT
Astro-E2
XMM-Newton
GAIA
JWST
EUSO
ISS Columbus
AMS
MAXI

Geant4 in space science

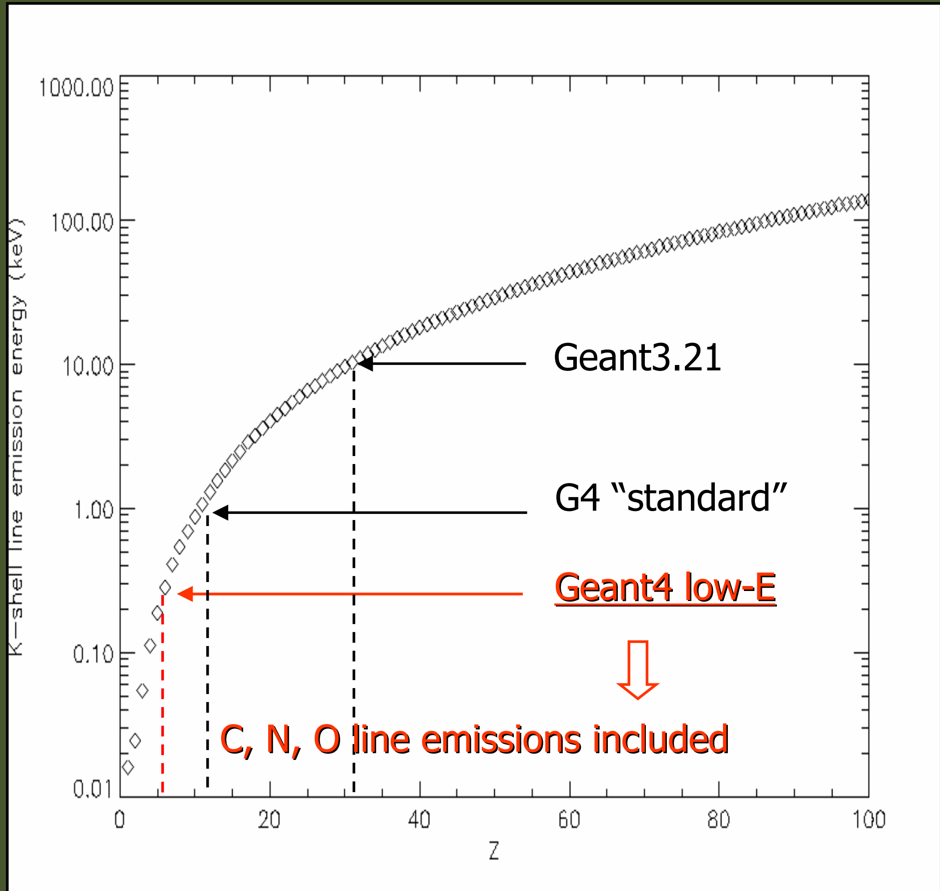
ESA Space Environment & Effects Analysis Section

X-Ray Surveys of Asteroids and Moons

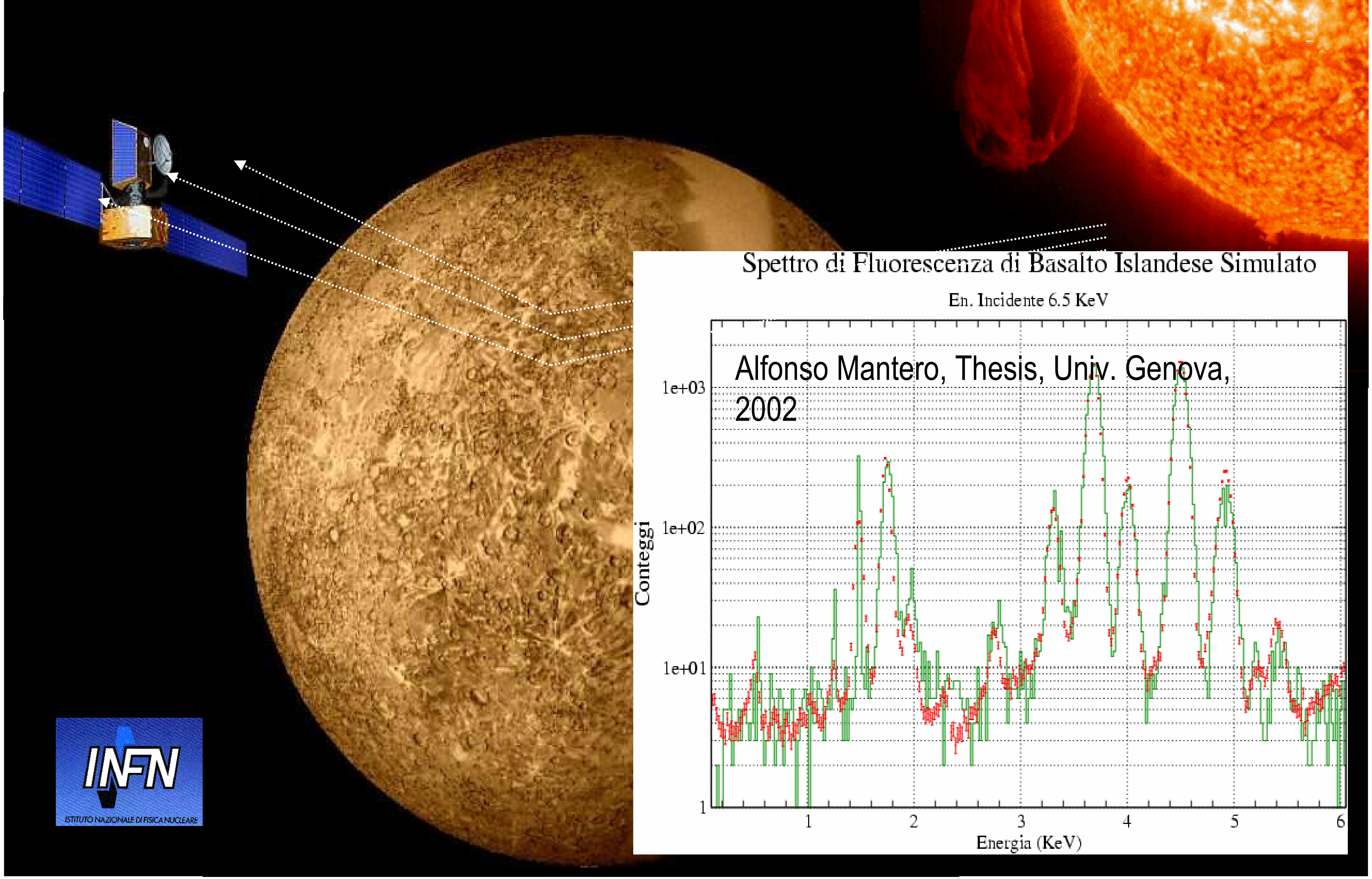


Courtesy SOHO EIT

Induced X-ray line emission: indicator of target composition
(~100 μm surface layer)

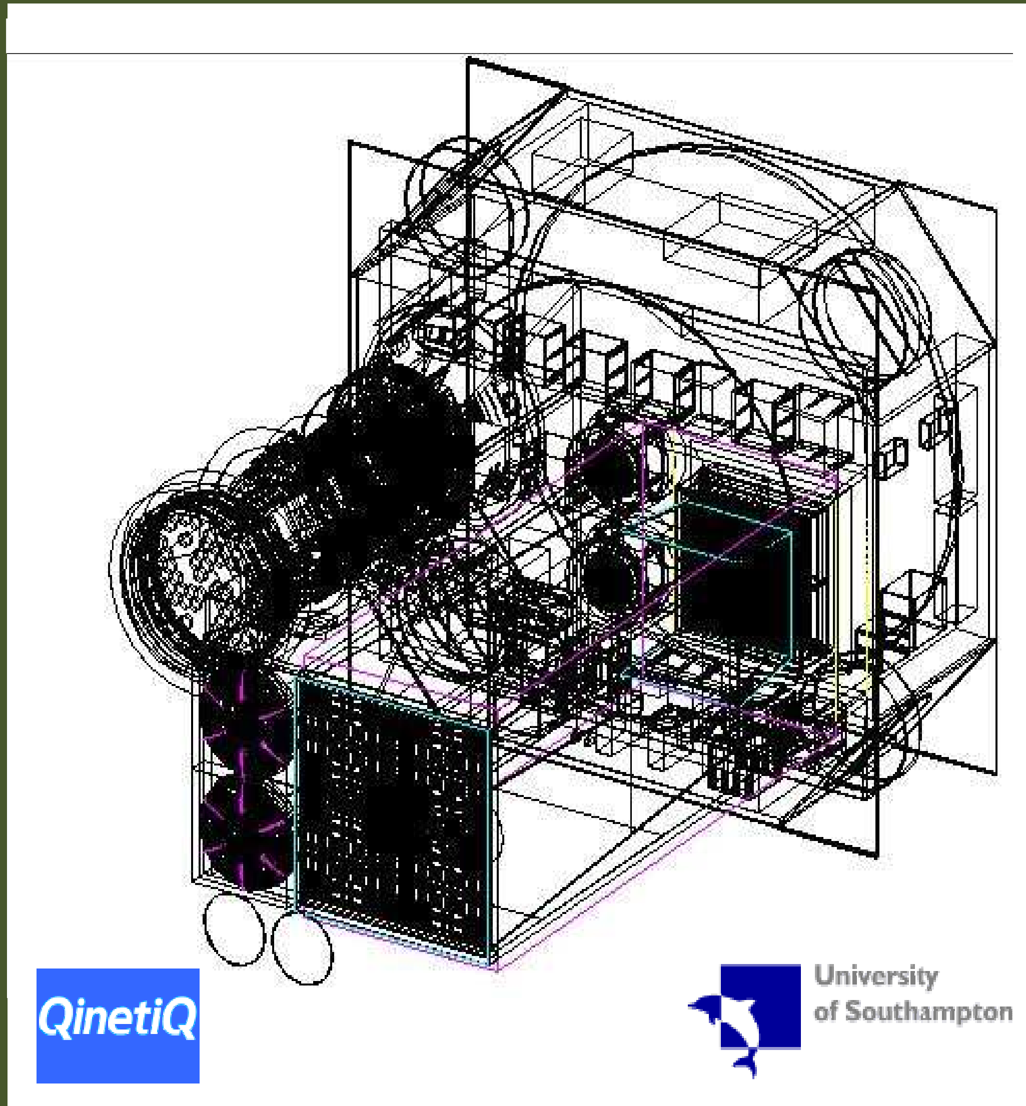


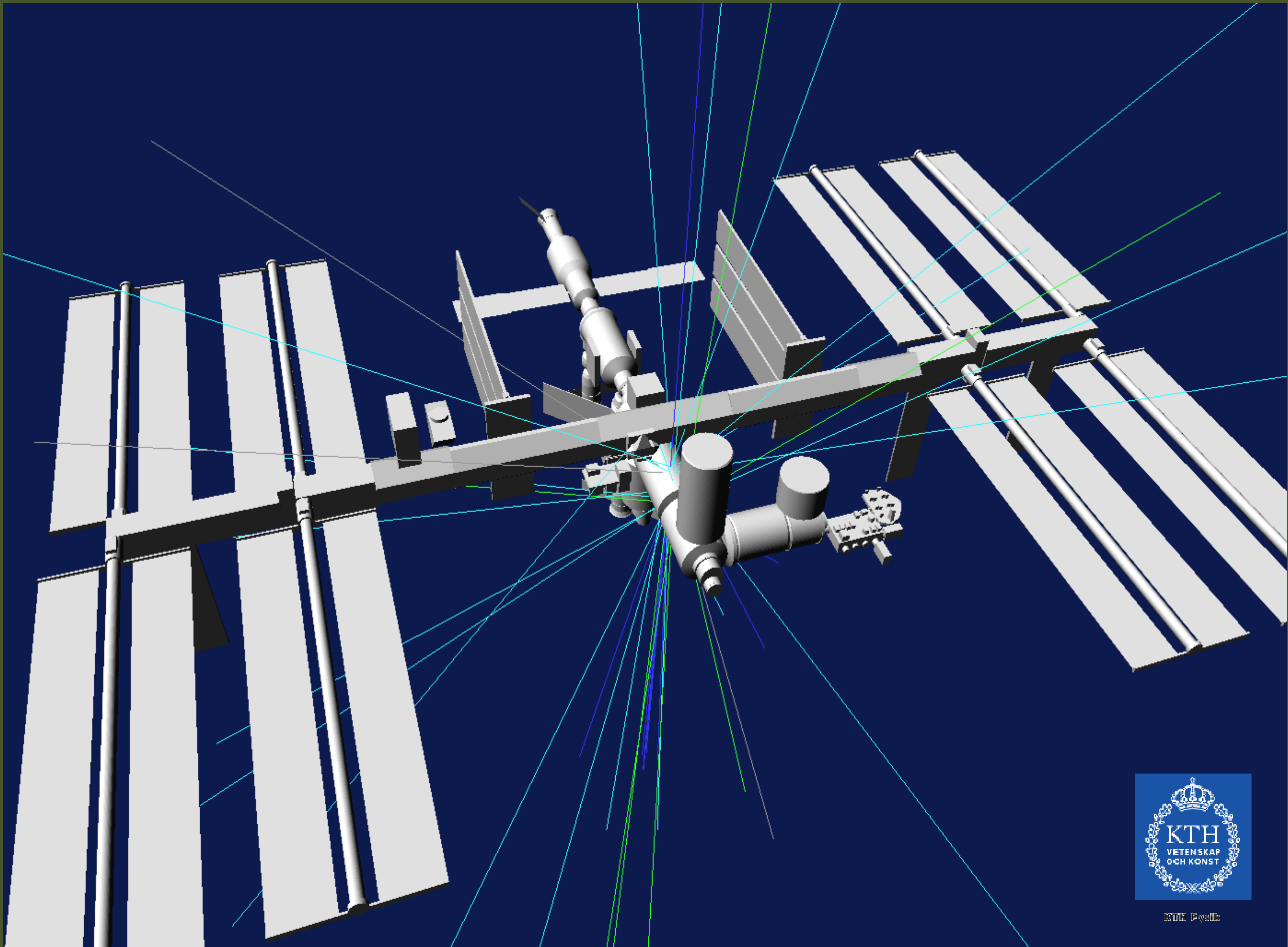
Bepi Colombo: X-Ray Mineralogical Survey of Mercury



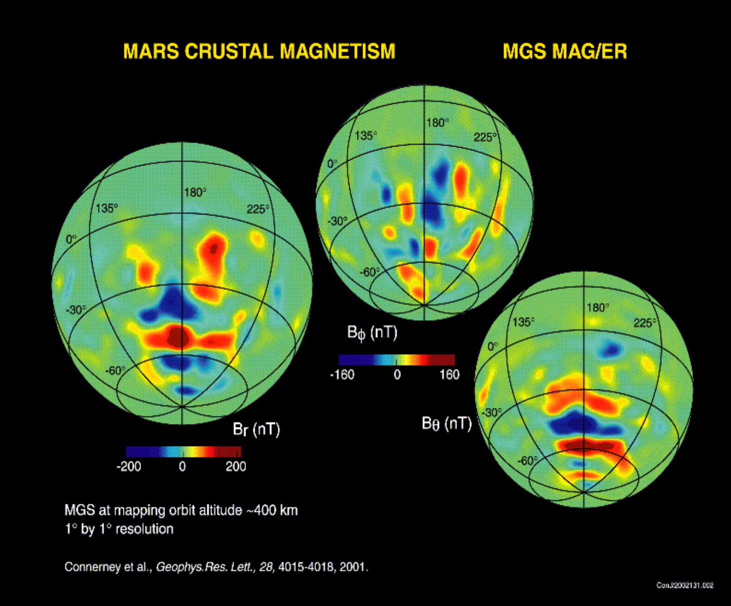
INTEGRAL Geant4 model by University of Southampton

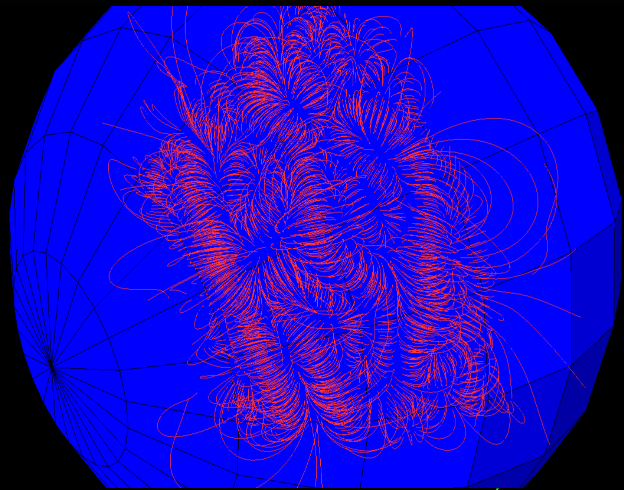
INTEGRAL in the ESA/ESTEC test center

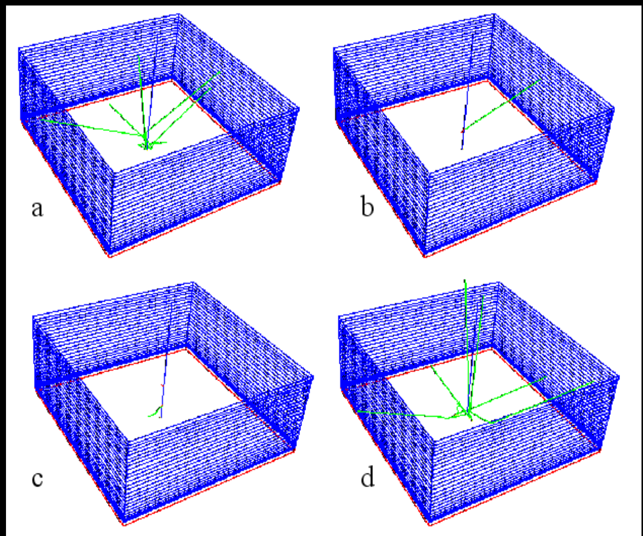


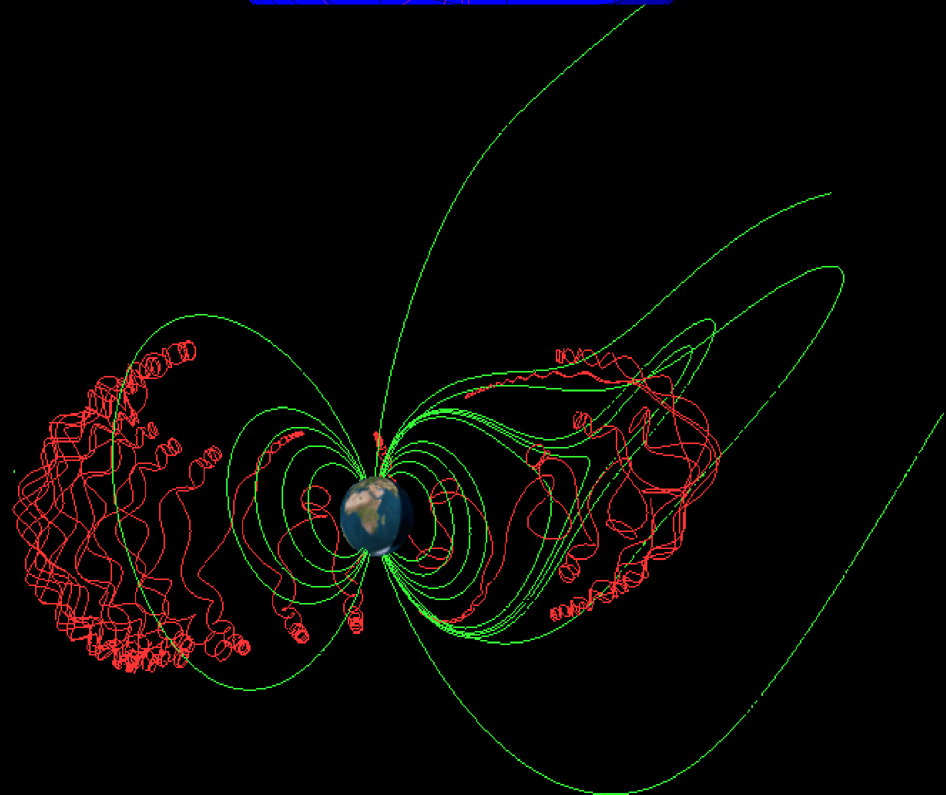


Courtesy T. Ersmark, KTH Stockholm

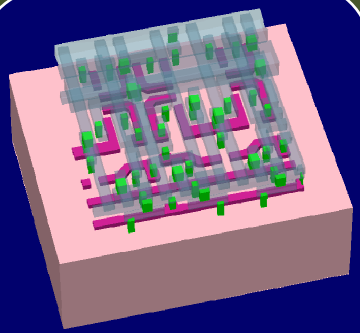

MARS CRUSTAL MAGNETISM
MGS MAG/ER
Br (nT)
-200
0
200
Bϕ (nT)
-160
0
160
Bθ (nT)
135°
180°
225°
0°
-30°
-60°
MGS at mapping orbit altitude ~400 km
1° by 1° resolution
Connerney et al., Geophys.Res. Lett., 28, 4015-4018, 2001.



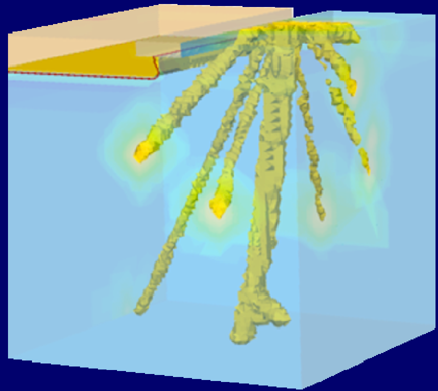

a
b
c
d



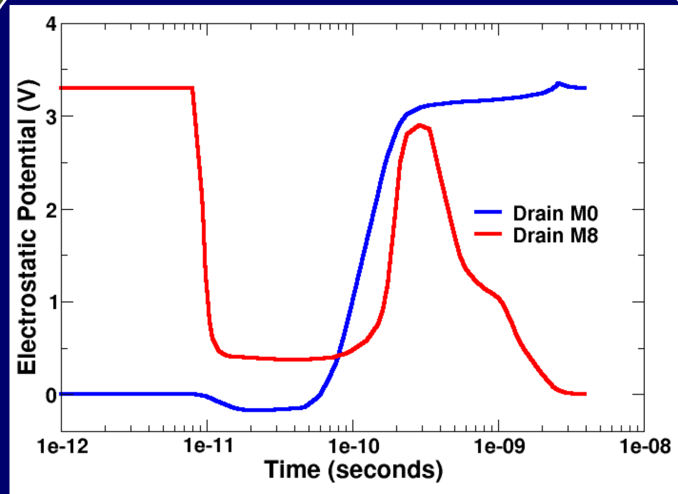
RADSAFE on SEE in SRAMs



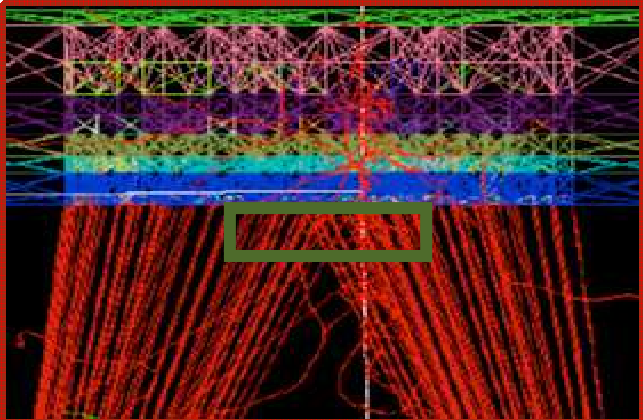
TCAD Cell Structure: SRAM Cell



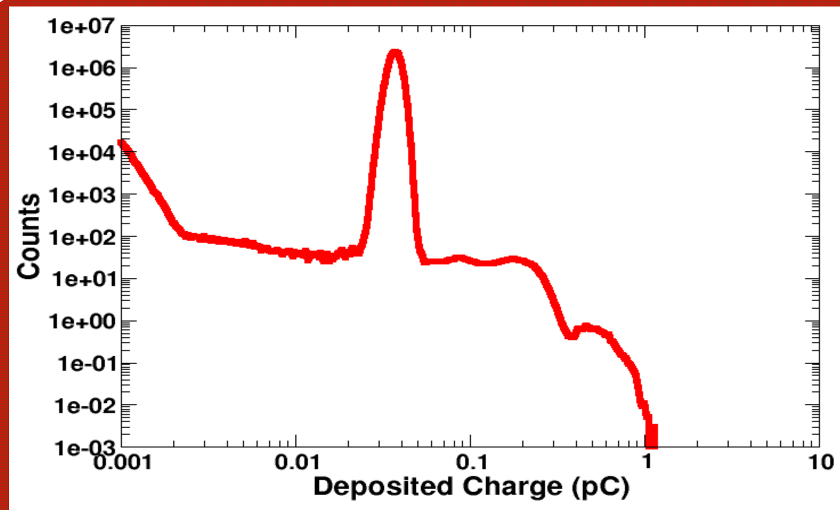
Single Charge Deposition in TCAD: Ne+W Event



SRAM Cell Upset



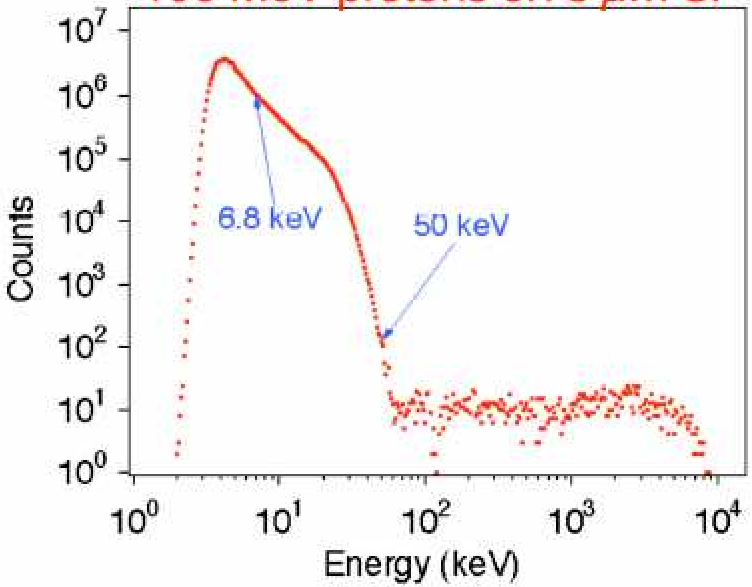
Geant4 Geometry and 523 MeV Neon Event

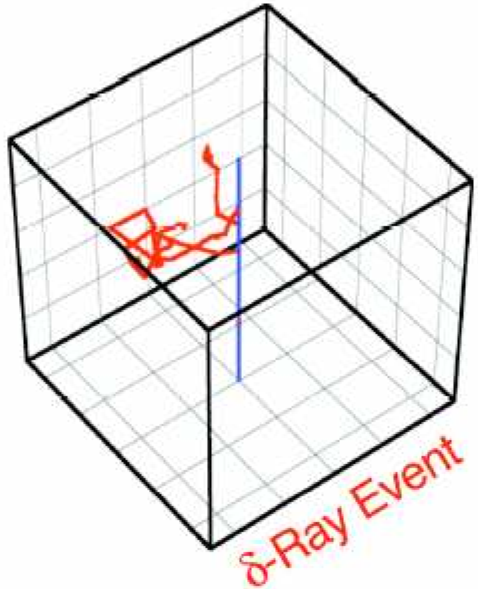


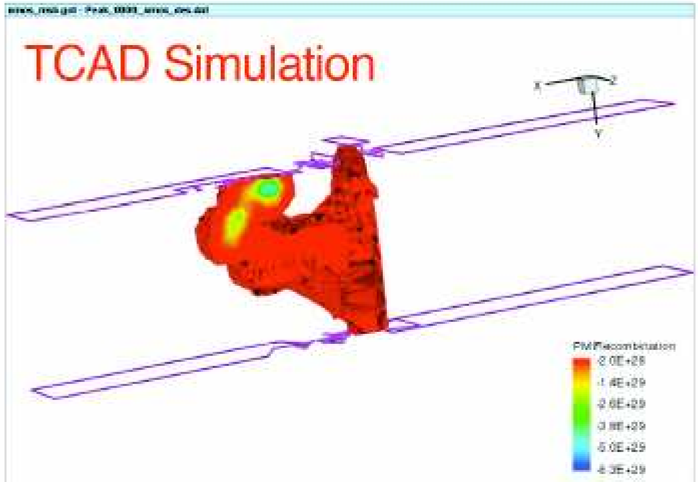
MRED Energy Deposition for 10^8 Events

Some preliminary pictures

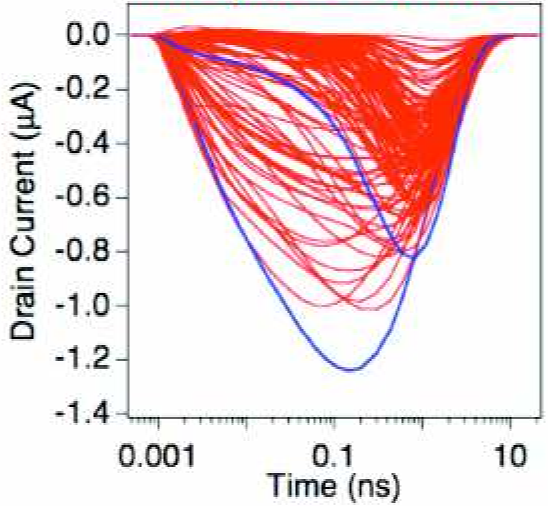
100 MeV protons on 5 μm Si



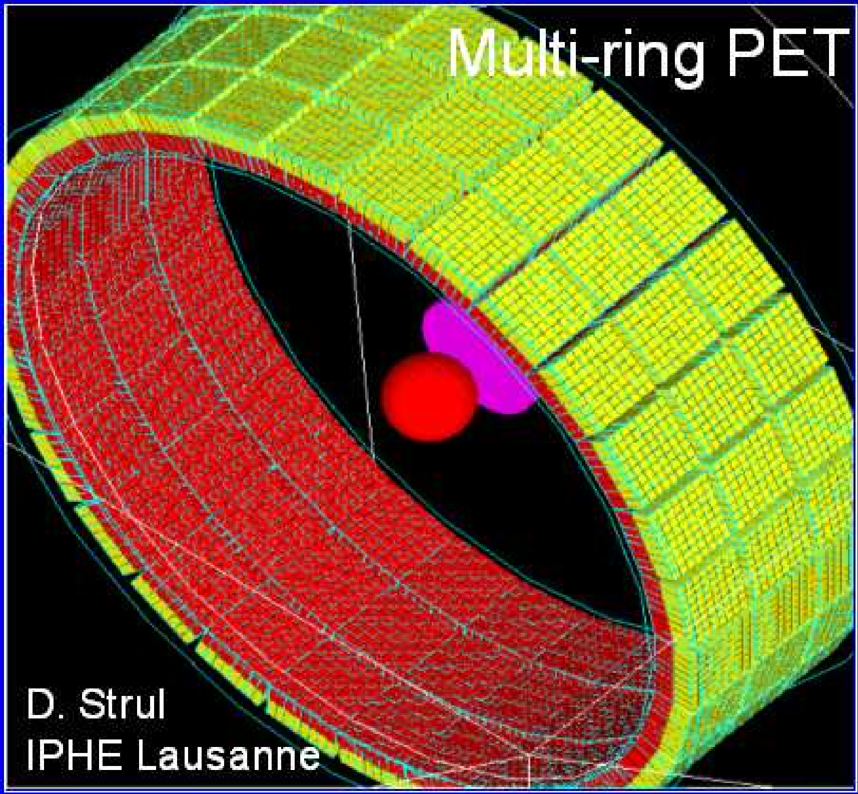


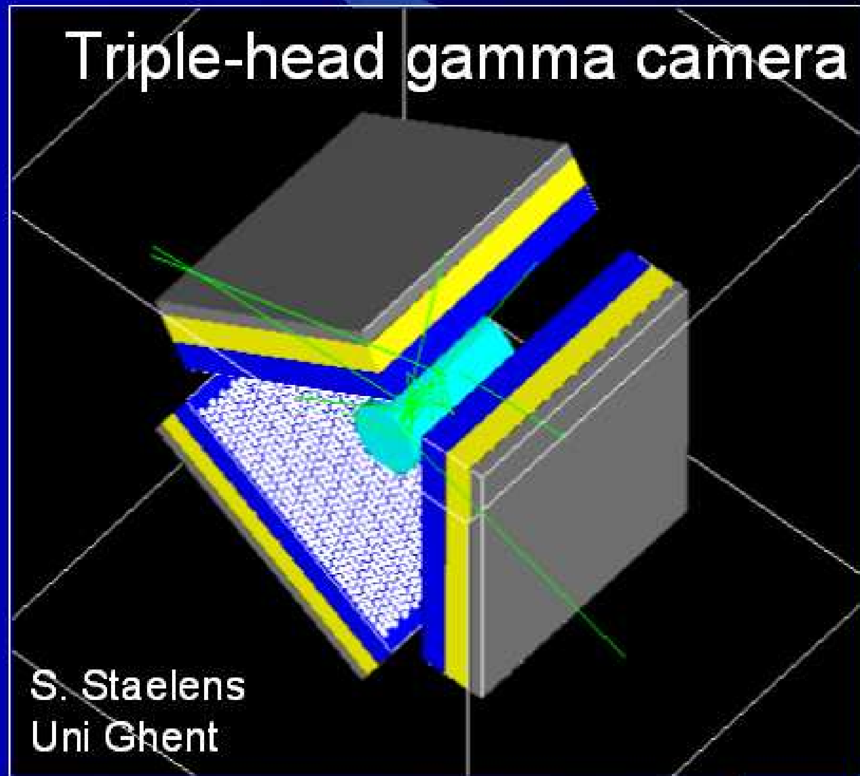


Response Curves



Geometry examples of GATE applications

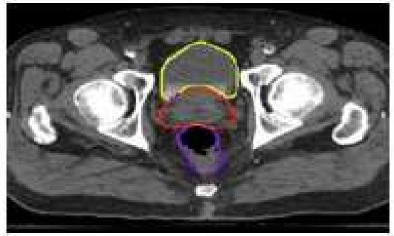




Geant4 DICOM Interface

Pavillon L'Hôtel-Dieu
Centre hospitalier universitaire de Québec

Modeling complex structures



file

......................DICM....UL......
....OB.............UI..1.2.840.10008.
5.1.4.1.1.2.....UI..1.3.12.2.1107.5.
1.4.40238.5.0.14178572411846....UI..
1.2.840.10008.1.2.1.....UI..1.3.12.2
.1107.5.1.4.....SH..SIEMENS_S5VA30A
....CS..ISO_IR 100....CS".ORIGINAL\P
RIMARY\AXIAL\CT_SOM5 SPI....UI..1.2.
840.10008.5.1.4.1.1.2.....UI..1.3.12
.5.0.141785724118
!.DA..20020109..
..20020109..0.TM
TM..094310.18600
987 ..3.TM..0943
.CS..CT..p.LO..
L-DIEU DE QUEBEC
S.QUEBEC.QUEBEC
SH..HNCT40238 ..

Reproduce patient's anatomy in a Geant4 application

UNIVERSITÉ LAVAL

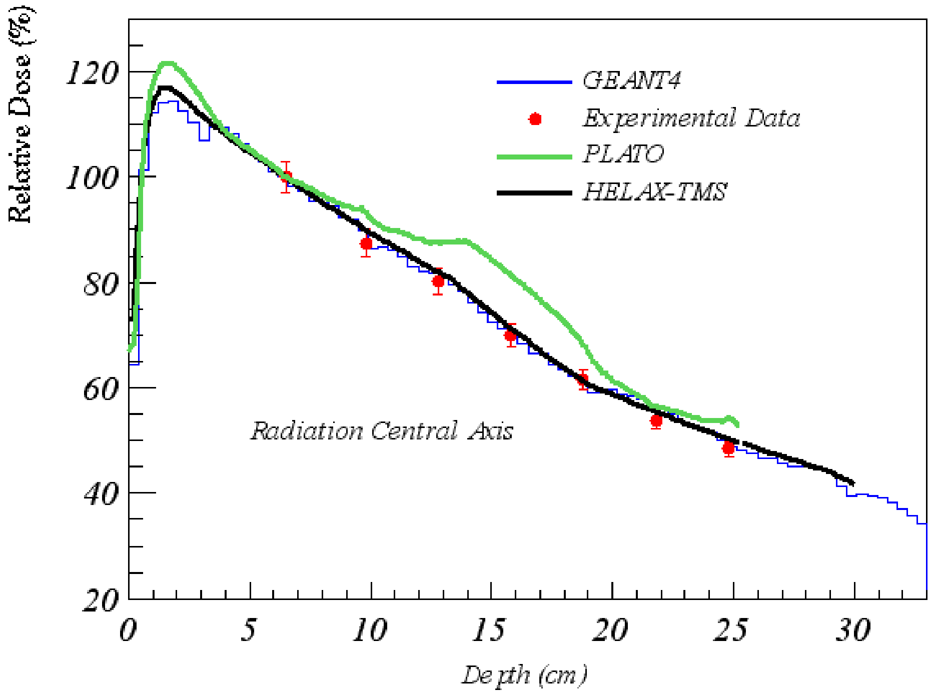
Authors: L. Archambault, L. Beaulieu, V.-H. Tremblay

(Univ. Laval and l'Hôtel-Dieu, Québec)

Comparison with commercial treatment planning systems

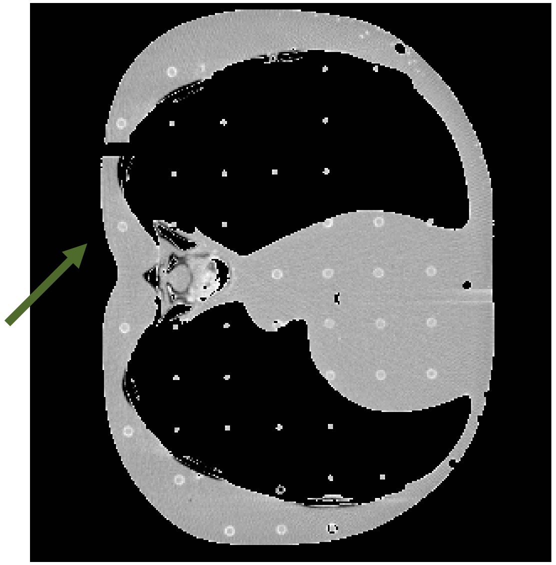
M. C. Lopes [1], L. Peralta [2], P. Rodrigues [2], A. Trindade [2]
[1] IPOFG-CROC Coimbra Oncological Regional Center - [2] LIP - Lisbon

CT-simulation with a Rando phantom
Experimental data obtained with TLD LiF dosimeter

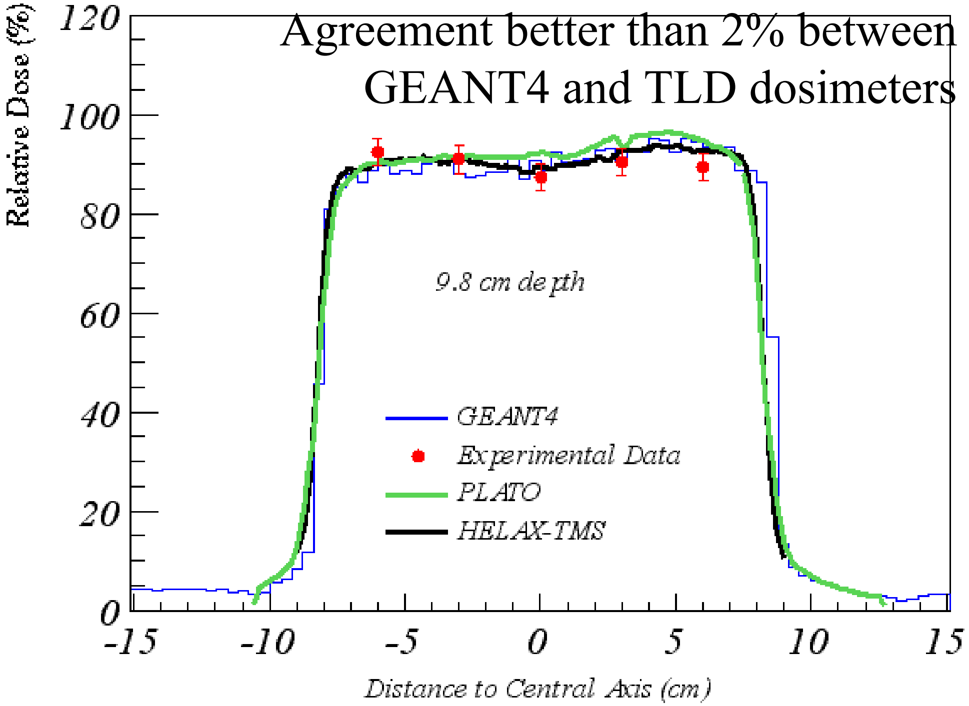


CT images used to define the geometry:

a thorax slice from a Rando anthropomorphic phantom

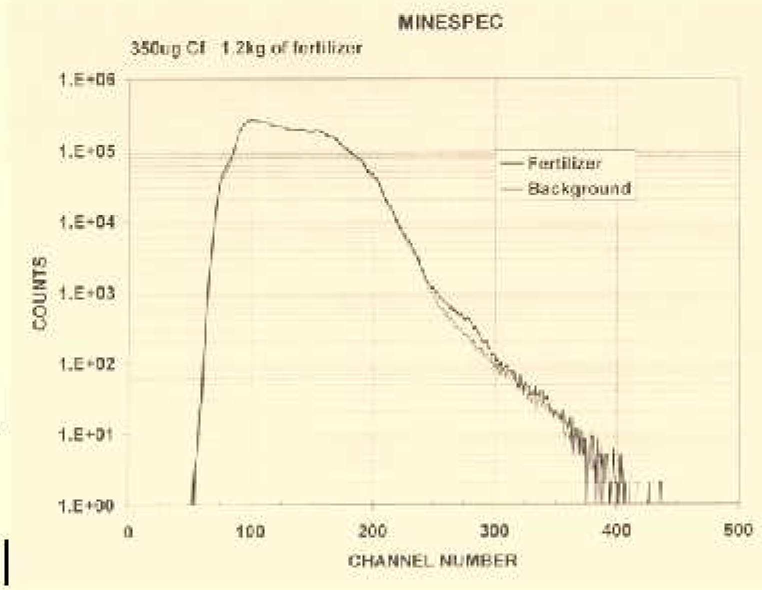


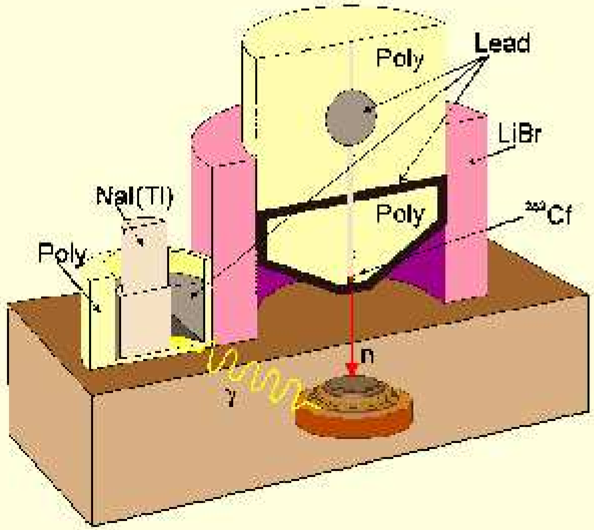
Agreement better than 2% between GEANT4 and TLD dosimeters



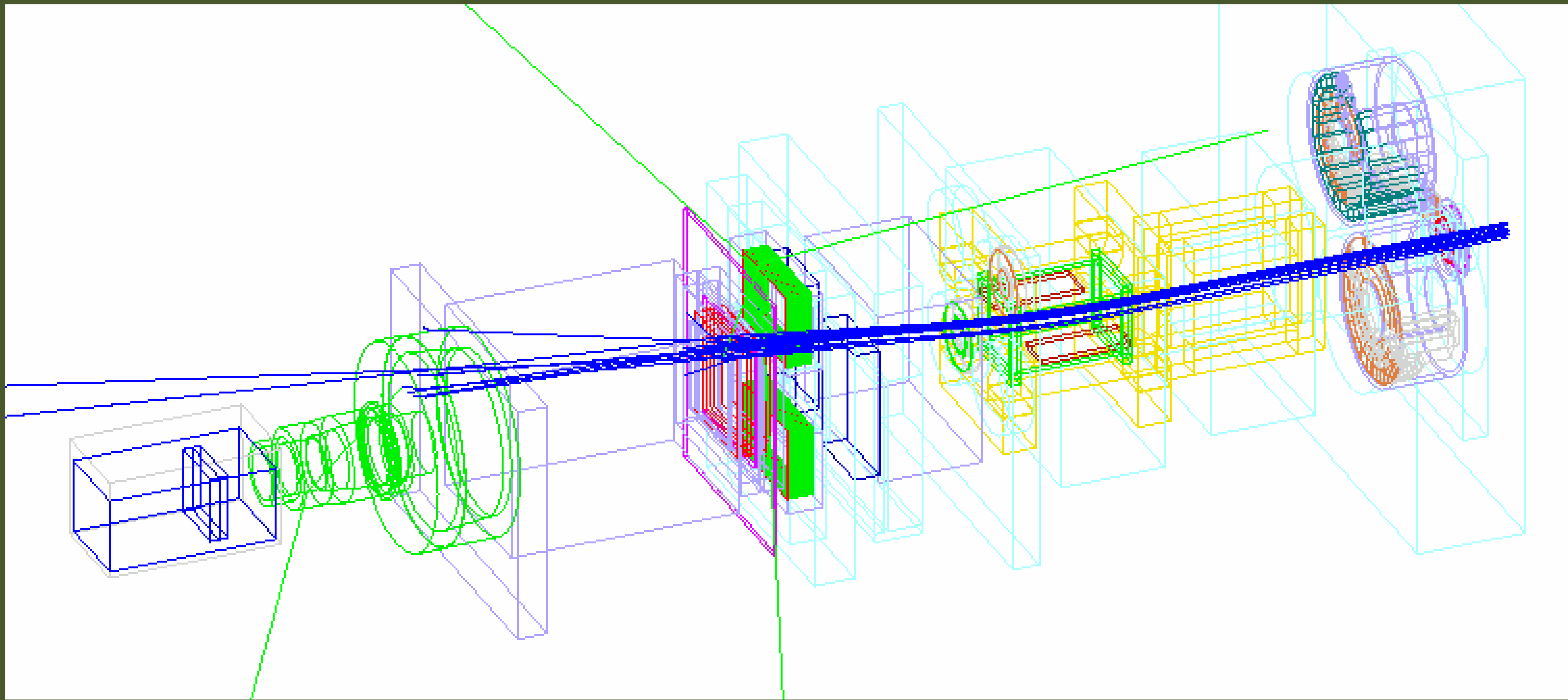
Thermal Neutron Activation

- TNA detects explosive by properties of constituents
    - High concentration of N
    - Does not ID explosive
- Can confirm presence of all surface laid or shallow AT mines in few seconds to 1 minute
- AT up to 20 cm deep and large AP mines in < 5 minutes





Magnetic fields

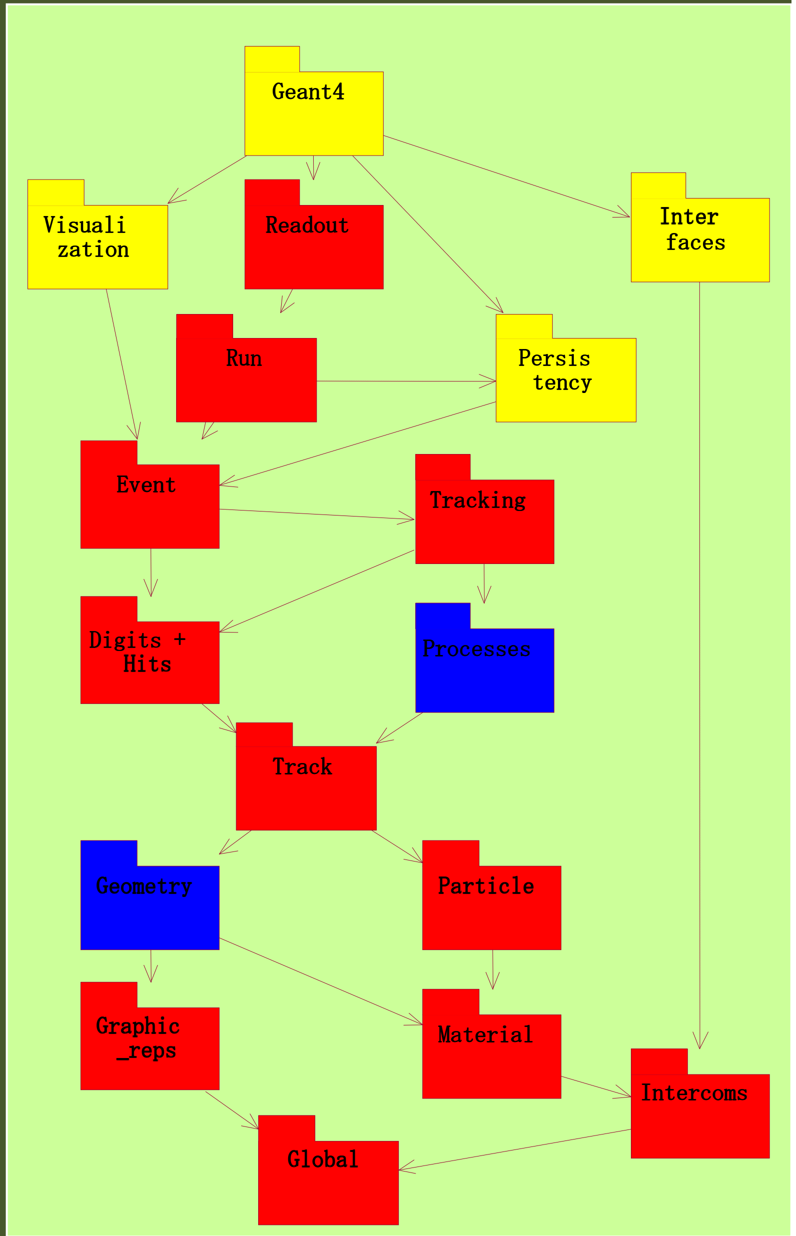


Courtesy of H. Paganetti

Basic concepts and kernel structure

Geant4 kernel

- Geant4 consists of 17 categories.
  - Independently developed and maintained by WG(s) responsible to each category.
  - Interfaces between categories (e.g. top level design) are maintained by the global architecture WG.
- Geant4 Kernel
  - Handles run, event, track, step, hit, trajectory.
  - Provides frameworks of geometrical representation and physics processes.



Run in Geant4

- As an analogy of the real experiment, a run of Geant4 starts with "Beam On".
- Within a run, the user cannot change
  - detector geometry
  - settings of physics processes

    ---> detector is inaccessible during a run
- Conceptually, a run is a collection of events which share the same detector conditions.
- At the beginning of a run, geometry is optimized for navigation and cross-section tables are calculated according to materials appear in the geometry and the cut-off values defined.
- G4RunManager class manages processing a run, a run is represented by G4Run class or a user-defined class derived from G4Run.
- G4UserRunAction is the optional user hook.

Event in Geant4

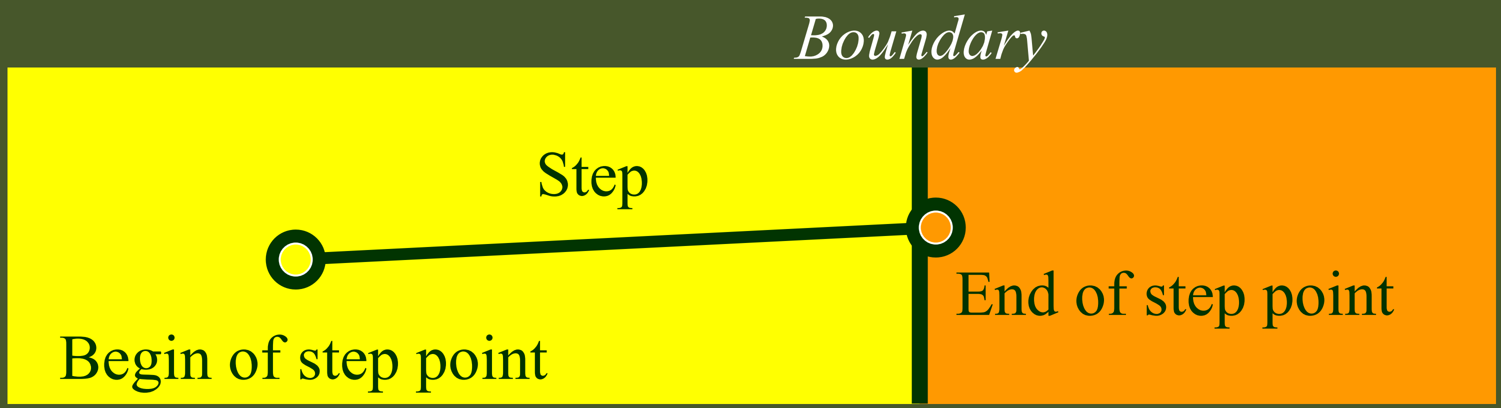
- At beginning of processing, an event contains primary particles. These primaries are pushed into a stack.
  - Physics processes create secondary particles
  - When the stack becomes empty, processing of an event is over.
- G4EventManager class manages processing an event.
- G4Event class represents an event. It has following objects at the end of its processing.
  - List of primary vertexes and particles (as input)
  - Hits collections
  - Trajectory collection (optional)
  - Digits collections (optional)
- G4UserEventAction is the optional user hook.

Track in Geant4

- Track is a snapshot of a particle.
  - It has only position and physical quantities of current instance.
- Step is a “delta” information to a track.
  - Track is not a collection of steps.
- Track is deleted when
  - it goes out of the world volume
  - it disappears (e.g. decay)
  - it goes down to zero kinetic energy and no “AtRest” additional process is required
  - the user decides to kill it
- No track object persists at the end of event.
  - For the record of track, use trajectory class objects.
- G4TrackingManager manages processing a track, a track is represented by G4Track class.
- G4UserTrackingAction is the optional user hook.

Step in Geant4

- Step has two points and also "delta" information of a particle (energy loss on the step, time-of-flight spent by the step, etc.).
- Each point knows the volume (and material). In case a step is limited by a volume boundary, the end point physically stands on the boundary, and it logically belongs to the next volume.
  - Because one step knows materials of two volumes, boundary processes such as transition radiation or refraction could be simulated.
- G4SteppingManager class manages processing a step, a step is represented by G4Step class.
- G4UserSteppingAction is the optional user hook.



Particle in Geant4

- A particle in Geant4 is represented in three layers of classes.
- G4Track
  - Position, geometrical information, etc.
  - This is a class representing a particle to be tracked.
- G4DynamicParticle
  - "Dynamic" physical properties of a particle, such as momentum, energy, spin, etc.
  - Each G4Track object has its own and unique G4DynamicParticle object.
  - This is a class representing an individual particle (which is not necessarily to be tracked).
- G4ParticleDefinition
  - "Static" properties of a particle, such as charge, mass, life time, decay channels, etc.
  - G4ProcessManager which describes processes involving to the particle
  - All G4DynamicParticle objects of same kind of particle share the same G4ParticleDefinition.

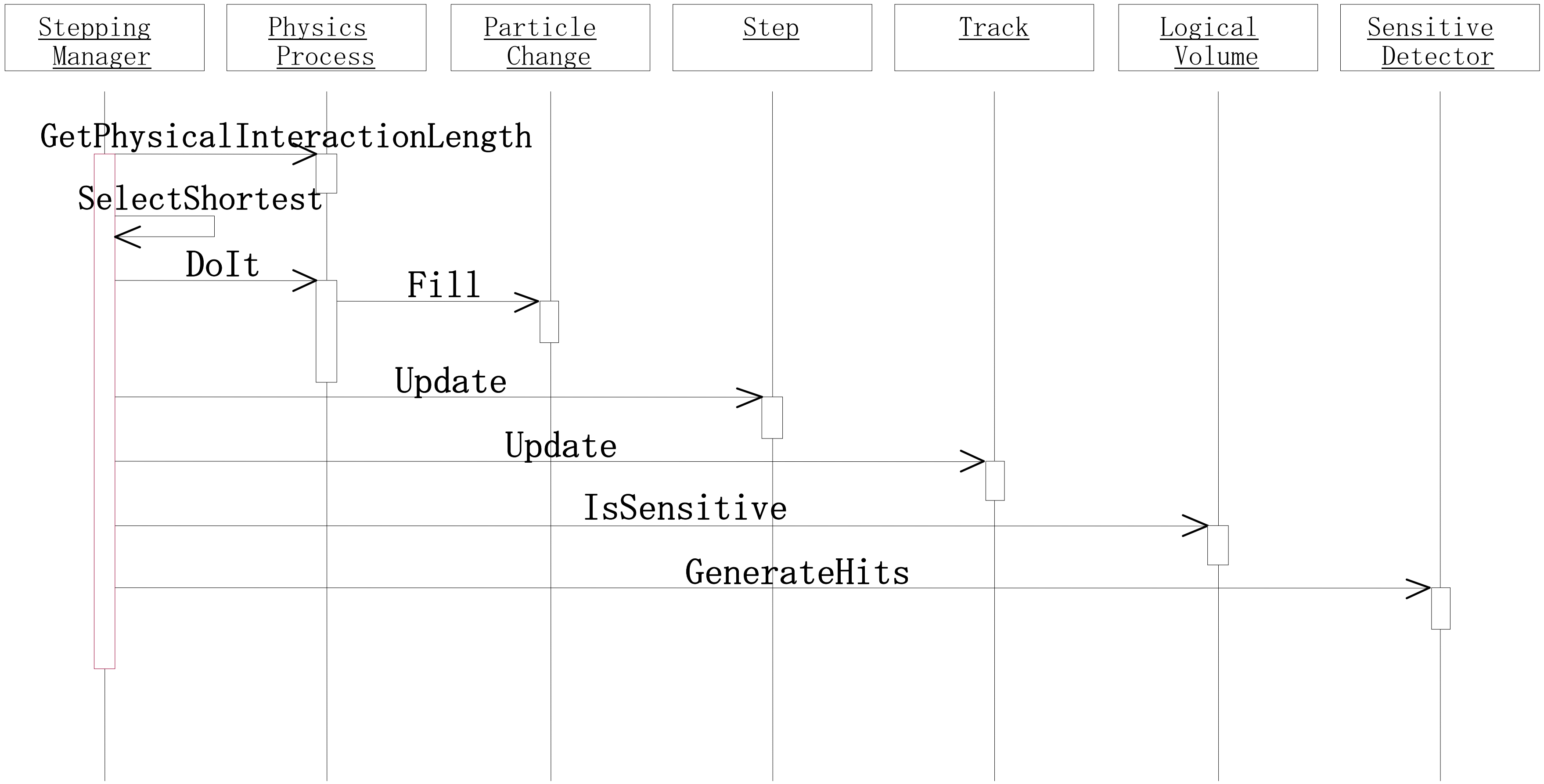
Tracking and processes

- Geant4 tracking is general.
  - It is independent to
    - the particle type
    - the physics processes involving to a particle
  - It gives the chance to all processes
    - To contribute to determining the step length
    - To contribute any possible changes in physical quantities of the track
    - To generate secondary particles
    - To suggest changes in the state of the track
      - e.g. to suspend, postpone or kill it.

Processes in Geant4

- In Geant4, particle transportation is a process as well, by which a particle interacts with geometrical volume boundaries and field of any kind.
  - Because of this, shower parameterization process can take over from the ordinary transportation without modifying the transportation process.
- Each particle has its own list of applicable processes. At each step, all processes listed are invoked to get proposed physical interaction lengths.
- The process which requires the shortest interaction length (in space-time) limits the step.
- Each process has one or combination of the following natures.
  - AtRest
    - e.g. muon decay at rest
  - AlongStep (a.k.a. continuous process)
    - e.g. Celenkov process
  - PostStep (a.k.a. discrete process)
    - e.g. decay on the fly

How Geant4 runs (one step)



Cuts in Geant4

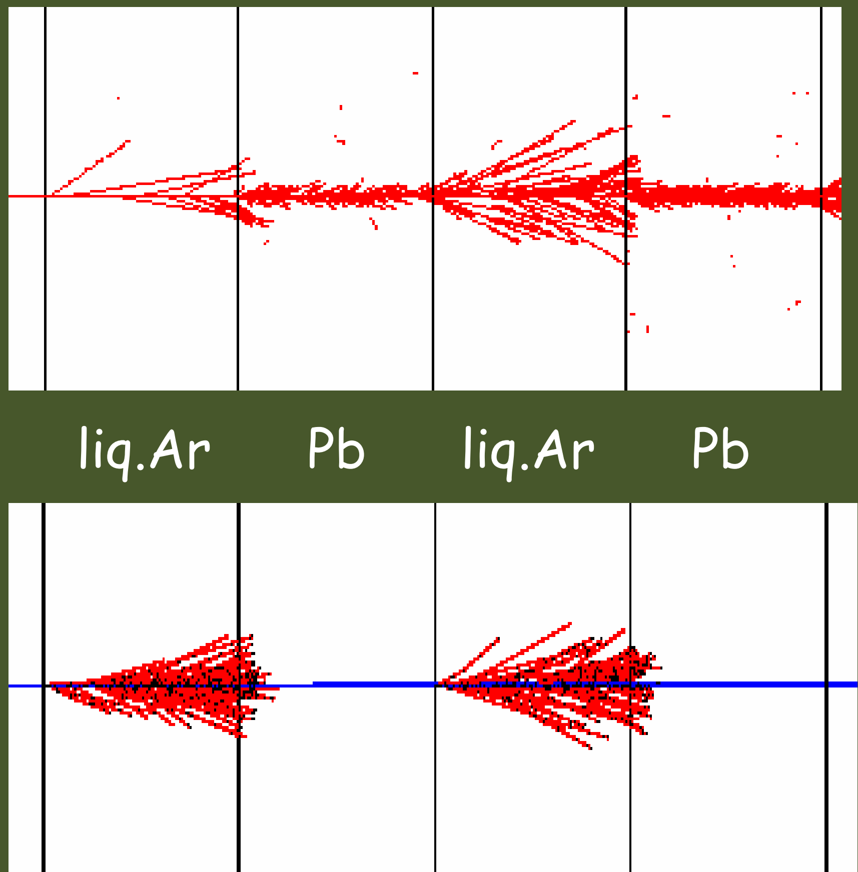
- A Cut in Geant4 is a production threshold.
  - Only for physics processes that have infrared divergence
  - Not tracking cut, which does not exist in Geant4 as default
- Energy threshold must be determined at which discrete energy loss is replaced by continuous loss
  - Old way:
    - Create secondaries only above cut-off energy, or add to continuous loss of primary for less energetic secondaries
    - Track primary particle until cut-off energy is reached, calculate continuous loss and dump it at that point, stop tracking primary
  - Geant4 way:
    - Create secondaries only above specified range, or add to continuous loss of primary for secondaries of less energetic than travelling the required range in the current material
    - Track primary down to zero range

Energy cut vs. range cut

- 500 MeV/c proton in liq.Ar (4mm) / Pb (4mm) sampling calorimeter

- Geant3 (energy cut)
  - Ecut = 450 keV

- Geant4 (range cut)
  - Rcut = 1.5 mm
  - Corresponds to
    Ecut in liq.Ar = 450 keV,
    Ecut in Pb = 2 MeV



Unit system

- Internal unit system used in Geant4 is completely hidden not only from user's code but also from Geant4 source code implementation.
- Each hard-coded number must be multiplied by its proper unit.

```
radius = 10.0 * cm;
kineticE = 1.0 * GeV;
```

- To get a number, it must be divided by a proper unit.

```
G4cout << eDep / MeV << " [MeV]" << G4endl;
```

- Most of commonly used units are provided and user can add his/her own units.
- By this unit system, source code becomes more readable and importing / exporting physical quantities becomes straightforward.
  - For particular application, user can change the internal unit to suitable alternative unit without affecting to the result.

G4cout, G4cerr

- G4cout and G4cerr are *ostream* objects defined by Geant4.
  - G4endl is also provided.

```
G4cout << "Hello Geant4!" << G4endl;
```

- Some GUIs are buffering output streams so that they display print-outs on another window or provide storing / editing functionality.
  - The user should not use std::cout, etc.
- The user should not use std::cin for input. Use user-defined commands provided by intercoms category in Geant4.
  - Ordinary file I/O is OK.

User classes

The user has to…

- Define material and geometry
- Select appropriate particles and processes
- Define production threshold(s)
- Define the way of primary particle generation
- Define the way to extract useful information from Geant4

- Optionally,
  - Define the way of visualization and interactivity
  - Provide the way of I/O
  - Select or provide some artificial mechanism for effective simulation
  - etc.

User classes

- Initialization classes
  - Use G4RunManager::SetUserInitialization() to define.
  - Invoked at the initialization
    - G4VUserDetectorConstruction
    - G4VUserPhysicsList
- Action classes
  - Use G4RunManager::SetUserAction() to define.
  - Invoked during an event loop
    - G4VUserPrimaryGeneratorAction
    - G4UserRunAction
    - G4UserEventAction
    - G4UserStackingAction
    - G4UserTrackingAction
    - G4UserSteppingAction
- main()
  - Geant4 does not provide *main().*

Note : classes written in yellow are mandatory.

The main program

- Geant4 does not provide the *main*().
- In your *main(),* you have to
  - Construct G4RunManager (or your derived class)
  - Set user mandatory classes to RunManager
    - G4VUserDetectorConstruction
    - G4VUserPhysicsList
    - G4VUserPrimaryGeneratorAction
- You can define VisManager, (G)UI session, optional user action classes, and/or your persistency manager in your *main()*.

Describe your detector

- Derive your own concrete class from G4VUserDetectorConstruction abstract base class.
- In the virtual method *Construct()*,
  - Instantiate all necessary materials
  - Instantiate volumes of your detector geometry
  - Instantiate your sensitive detector classes and set them to the corresponding logical volumes
- Optionally you can define
  - Regions for any part of your detector
  - Visualization attributes (color, visibility, etc.) of your detector elements

Select physics processes

- Geant4 does not have any default particles or processes.
  - Even for the particle transportation, you have to define it explicitly.
- Derive your own concrete class from G4VUserPhysicsList abstract base class.
  - Define all necessary particles
  - Define all necessary processes and assign them to proper particles
  - Define cut-off ranges applied to the world (and each region)
- Geant4 provides lots of utility classes/methods and examples.
  - "Educated guess" physics lists for defining hadronic processes for various use-cases.

Generate primary event

- Derive your concrete class from G4VUserPrimaryGeneratorAction abstract base class.
- Pass a G4Event object to one or more primary generator concrete class objects which generate primary vertices and primary particles.
- Geant4 provides several generators in addition to the G4VPrimaryParticlegenerator base class.
  - G4ParticleGun
  - G4HEPEvtInterface, G4HepMCInterface
    - Interface to /hepevt/ common block or HepMC class
  - G4GeneralParticleSource
    - Define radioactivity

Optional user action classes

- All user action classes, methods of which are invoked during "Beam On", must be constructed in the user's *main*() and must be set to the RunManager.

- G4UserRunAction
  - G4Run* GenerateRun()
    - Instantiate user-customized run object
  - void BeginOfRunAction(const G4Run*)
    - Define histograms
  - void EndOfRunAction(const G4Run*)
    - Store histograms
- G4UserEventAction
  - void BeginOfEventAction(const G4Event*)
    - Event selection
    - Define histograms
  - void EndOfEventAction(const G4Event*)
    - Analyze the event

Optional user action classes

- G4UserStackingAction
  - void PrepareNewEvent()
    - Reset priority control
  - G4ClassificationOfNewTrack ClassifyNewTrack(const G4Track*)
    - Invoked every time a new track is pushed
    - Classify a new track -- priority control
      - Urgent, Waiting, PostponeToNextEvent, Kill
  - void NewStage()
    - Invoked when the Urgent stack becomes empty
    - Change the classification criteria
    - Event filtering (Event abortion)

Optional user action classes

- G4UserTrackingAction
  - void PreUserTrackingAction(const G4Track*)
    - Decide trajectory should be stored or not
    - Create user-defined trajectory
  - void PostUserTrackingAction(const G4Track*)
- G4UserSteppingAction
  - void UserSteppingAction(const G4Step*)
    - Kill / suspend / postpone the track
    - Draw the step (for a track not to be stored as a trajectory)

Geant4 as a state machine

- Geant4 has six application states.
  - G4State_PreInit
    - Material, Geometry, Particle and/or Physics Process need to be initialized/defined
  - G4State_Idle
    - Ready to start a run
  - G4State_GeomClosed
    - Geometry is optimized and ready to process an event
  - G4State_EventProc
    - An event is processing
  - G4State_Quit
    - (Normal) termination
  - G4State_Abort
    - A fatal exception occurred and program is aborting