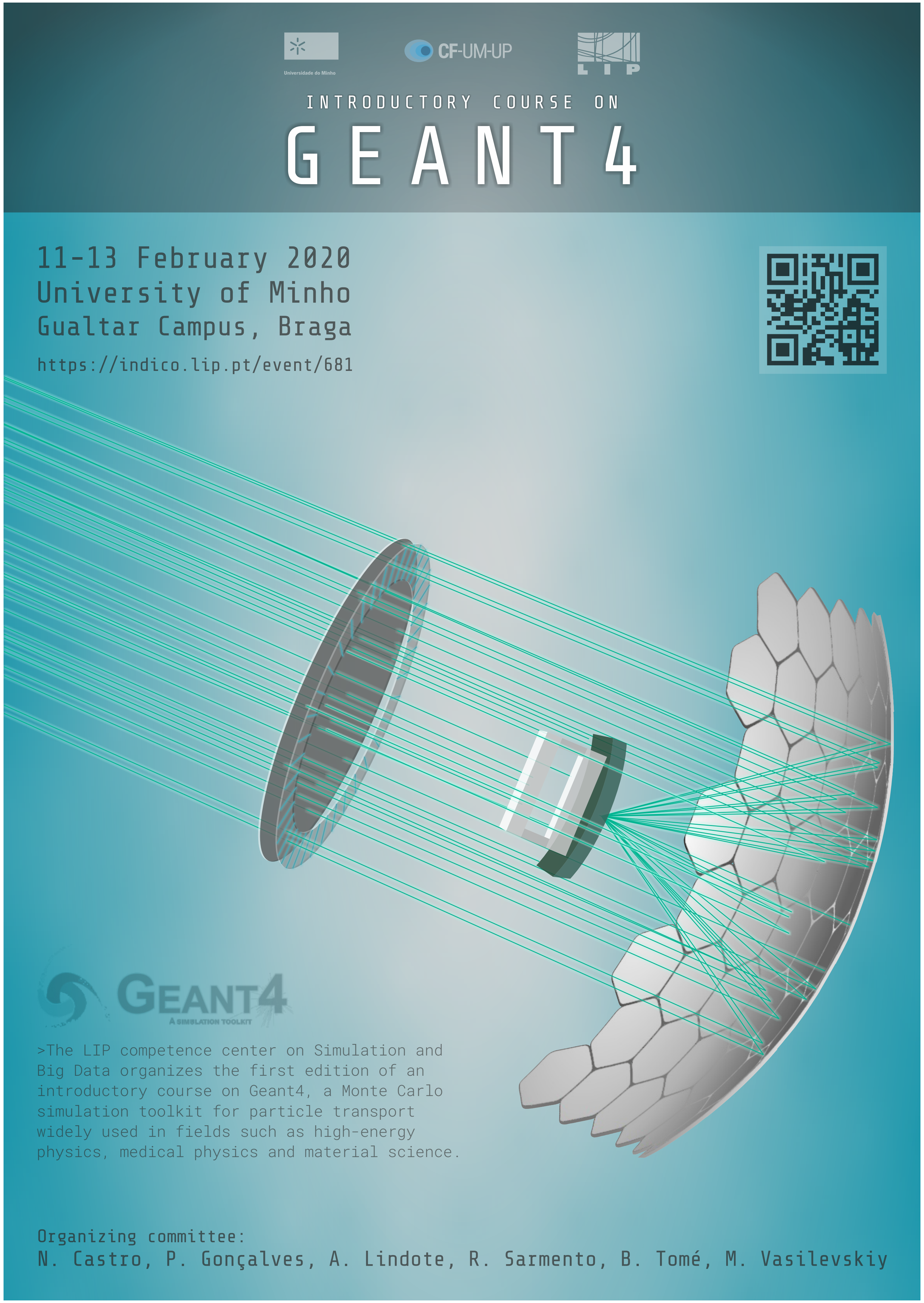
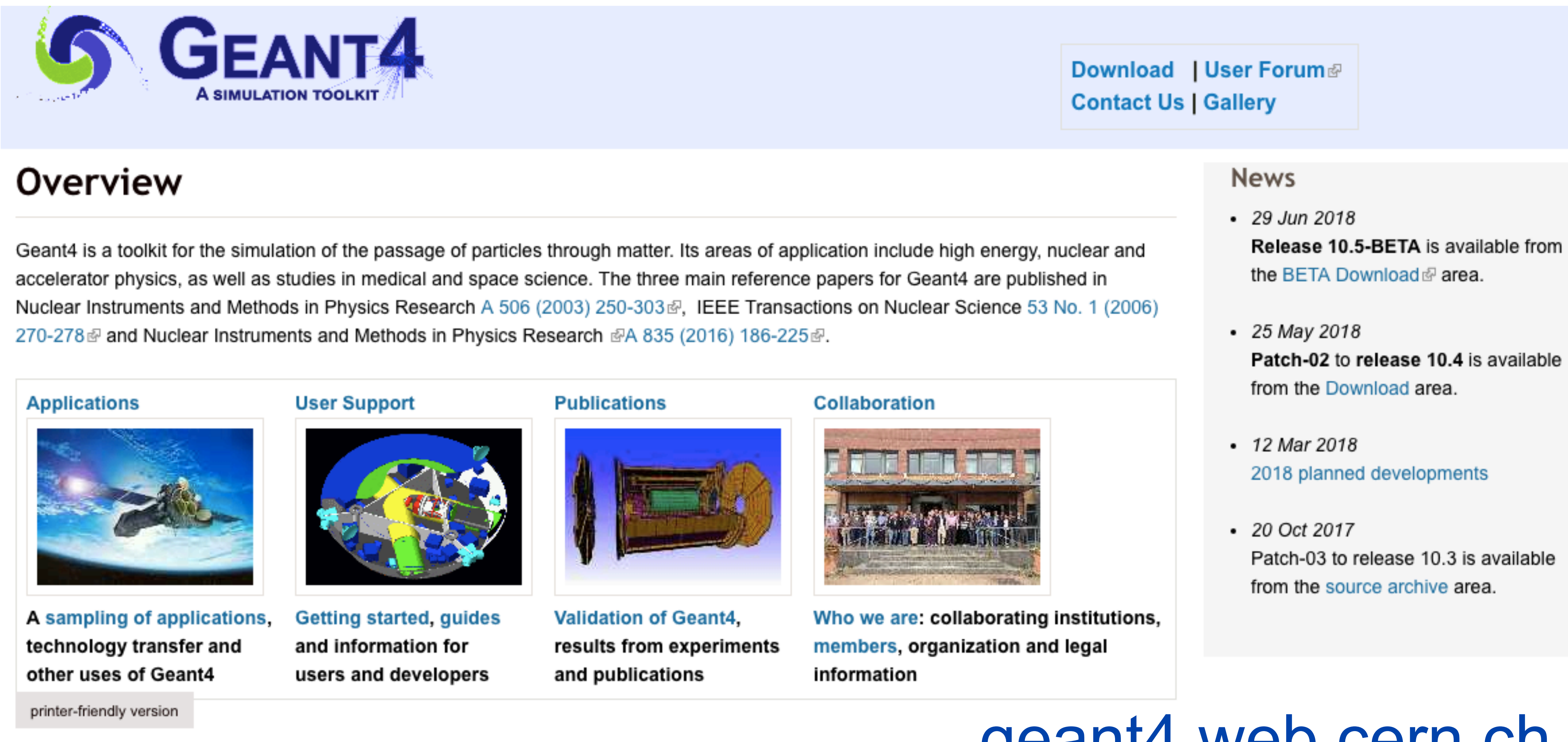
Introduction to Geant4



Geant4

- Particle Transport Monte Carlo simulation toolkit
- Developed at CERN
- Open source C++ framework
- GEometry ANd Tracking, version 4

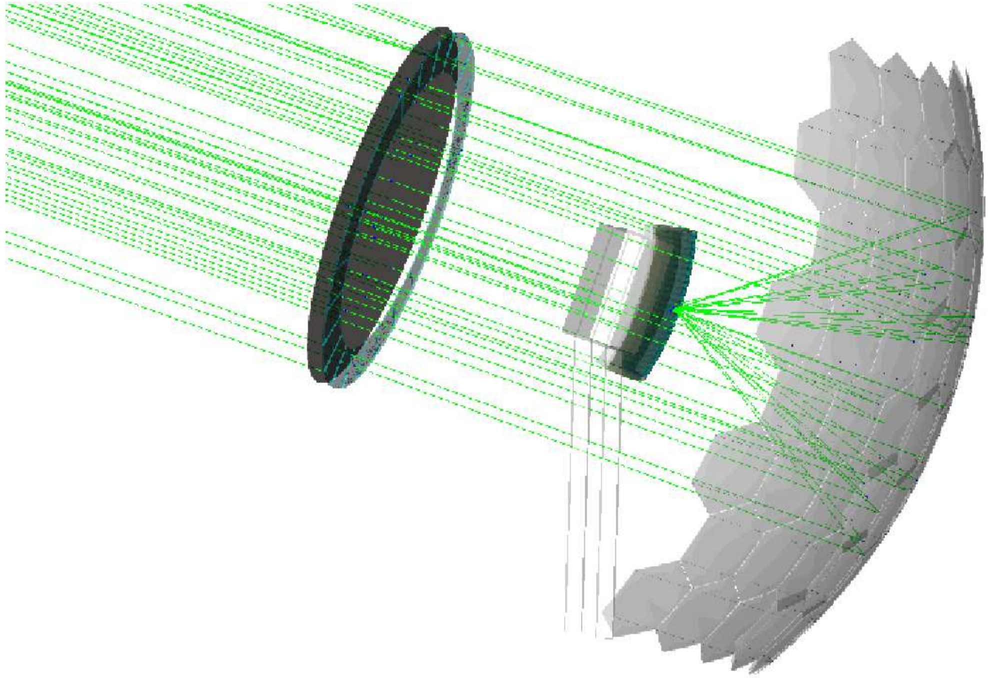


geant4.web.cern.ch

Selected example: particle physics

•A simulation of the fluorescence detectors of the Pierre Auger Observatory using Geant4

Proceedings of the 31st ICRC, Lodz, 2009



•Studies to estimate the light detection efficiency, necessary to evaluate the “source” energy
•Complex geometry (e.g. spot due to the camera, sensitivity to individual PMT position)

Selected example: medical physics

•Geant4 Monte Carlo simulation of absorbed dose and radiolysis yields enhancement from a gold nanoparticle under MeV proton irradiation

NIM B, 373 (2016) 126-139

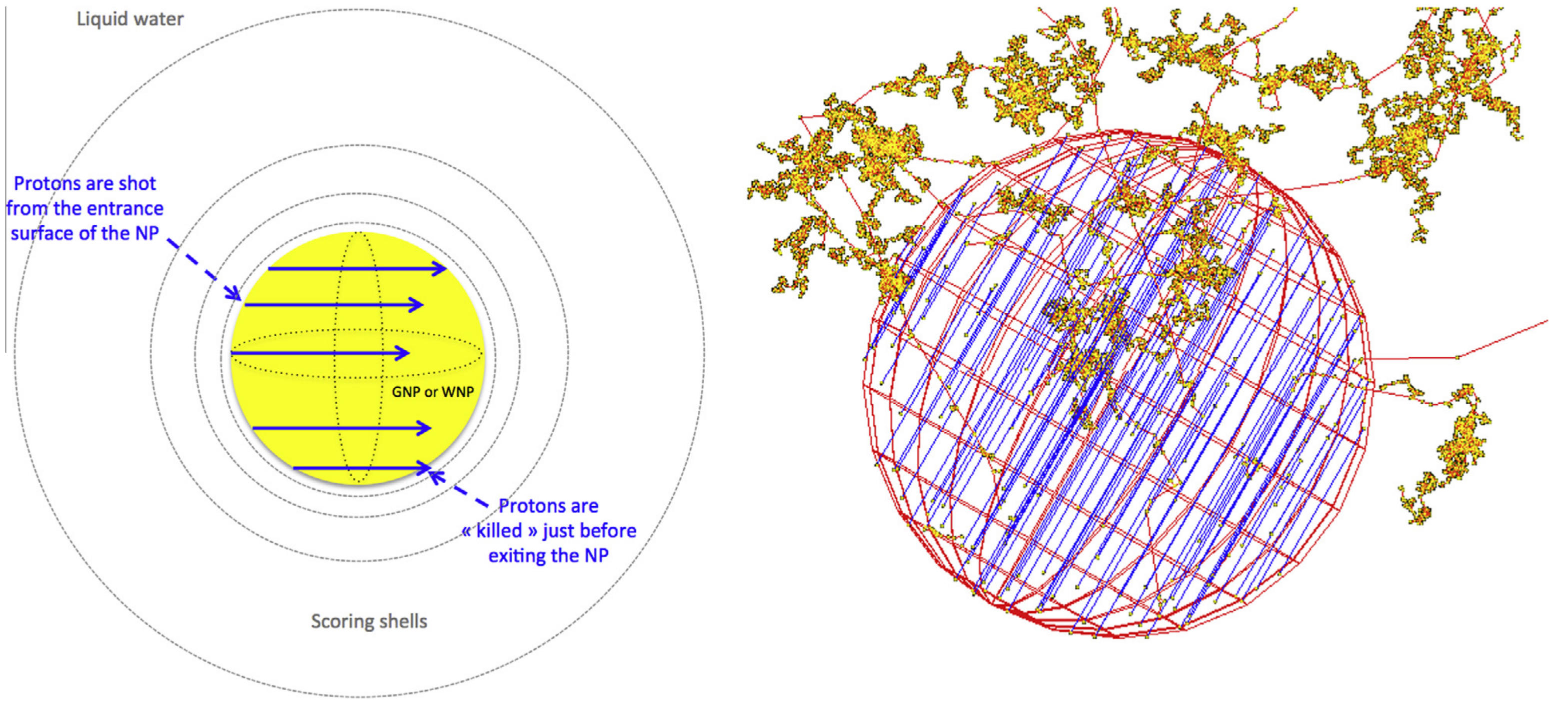


Fig. 1. Left: Schematic diagram of the simulated geometry showing the incident parallel protons (full line) inside the NP (yellow sphere) and the scoring concentric spherical shells. Right: example of visualization obtained with Geant4 when the NP (in red) is irradiated with a parallel proton beam (blue tracks), showing emitted secondary electron interactions (red tracks and yellow vertices). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Monte Carlo

•Simple example: tracking gamma rays in a scintillator detector with a simple shape and determining the deposited energy spectrum

•The physics is known from first principles (photon interactions: photoelectric effect, Compton scattering, optical processes, etc)

•A complete analytical treatment of the problem is nearly impossible

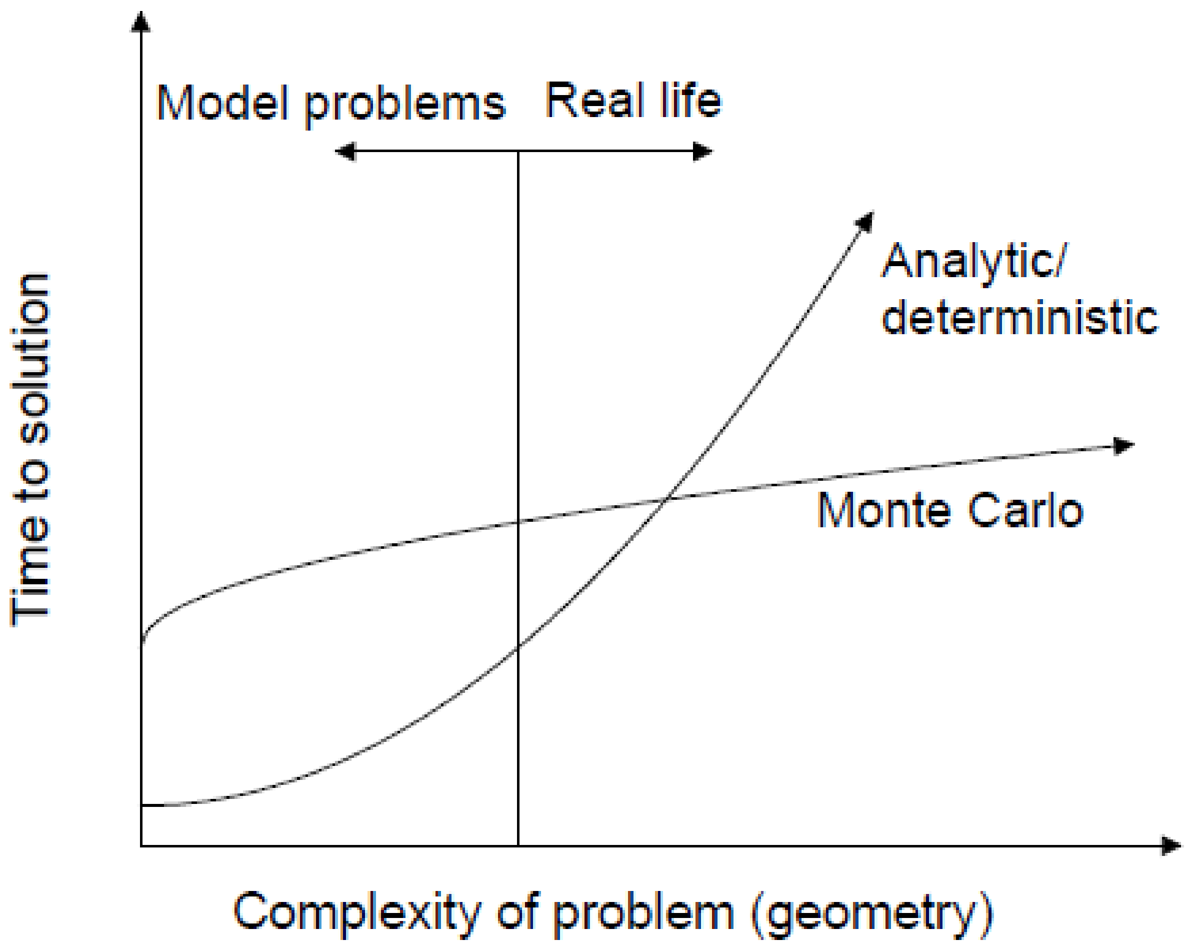
Monte Carlo

•When the analytical calculation from physics law is unpractical, due to:

- -complicated experimental geometries
- -multiple physics processes, variables involved

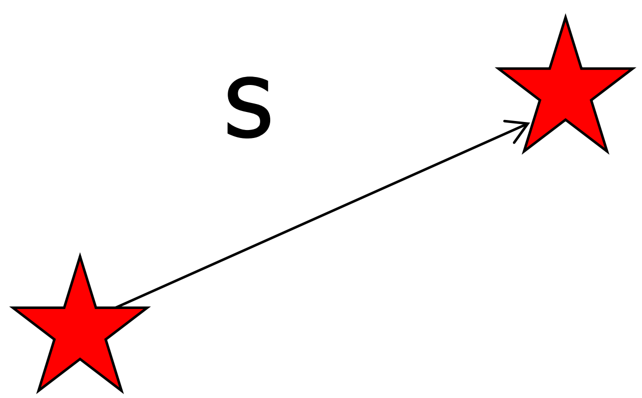
Monte Carlo techniques are used to predict the outcome

•Particle transport MC or tracking in Geant4



Tracking

- Distance s between two subsequent interactions distributed as $\boxed{p(s) = \mu e^{-\mu s}}$



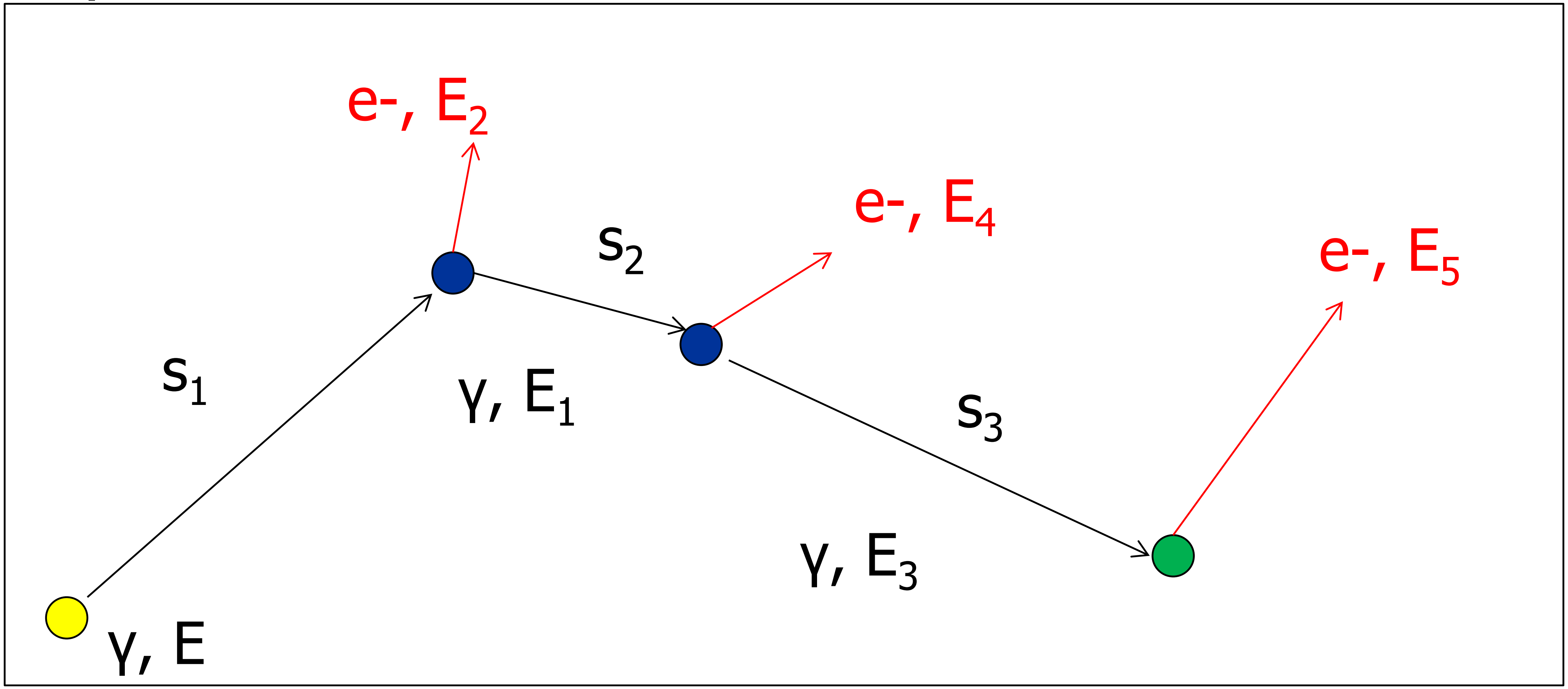
  - μ is a property of the medium (homogeneous) and of the physics
  - μ is proportional to the **total** cross section and depends on the density of the material $\mu = N\sigma = N\sum_i \sigma_i = \sum_i \mu_i$

- All competing processes contribute with their own μ_i
- Each process takes place with probability μ_i/μ → i.e. proportionally to the partial cross sections

Tracking

- Divide the trajectory of the particle in **"steps"**
  - Straight free-flight tracks between consecutive physics interactions
  - Steps can also be limited by geometry boundaries
- Decide the step length s, by sampling according to $p(s)= \mu e^{-\mu s}$, with the proper μ (material+physics)
- Decide which interaction takes place at the end of the step, according to μ_i/μ
- Produce the final state according to the **physics** of the interaction ($d^2\sigma/d\Omega dE$)
  - Update direction of the primary particle
  - Store somewhere the possible secondary particles, to be tracked later on

Tracking



Tracking

- This basic recipe works fine for γ-rays and other neutral particles (e.g. neutrons)
- Not so well for $e^{\pm}$: the cross section (ionization & bremsstrahlung) is very high, so the steps between two consecutive interactions are very small

- Simulate explicitly (i.e. force step) interactions **only if** energy loss (or change of direction) is above threshold W_0
  - Detailed simulation
  - "**hard**" interaction (like γ interactions)
- The effect of all sub-threshold interactions is described **statistically** (= cumulatively)
  - Condensed simulation
  - "**soft**" interactions

Tracking in Geant4

•**Particle** transportation in Geant4, **step** by **step**, takes into account all possible interaction **processes** with **materials** and fields

•Refined algorithm to take into account geometric boundaries, decays, etc

•**Track** ends when particle:
- leaves the simulation world volume
- disappears in an interaction
- kinetic energy goes to zero (and no AtRest process)
- is (artificially) killed

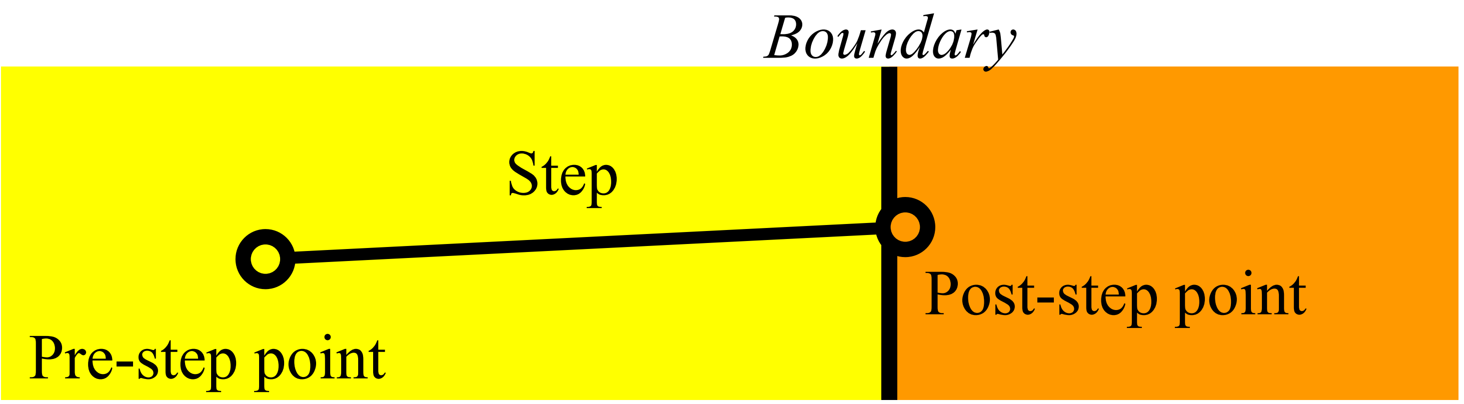
More concepts

- An **event** is the basic unit of simulation
  - begins with generation of a primary, whose track is pushed to the **stack**
  - secondary tracks are also stacked
  - each track from the stack is followed at a time
  - ends when the stack is empty

- A **run** is an event loop
  - geometry and physics are fixed at the onset
  - starts with beamOn

More concepts

Information may be accessed at many levels in the simulation (step, track, etc)



- **Step point**
  - knows the volume where it sits
  - if no interaction, is limited by a volume boundary

- A **process** may occur:
  - at rest (e.g. decay from rest)
  - along step (continuous energy loss)
  - post step (decay on the fly)

Geant4 kernel

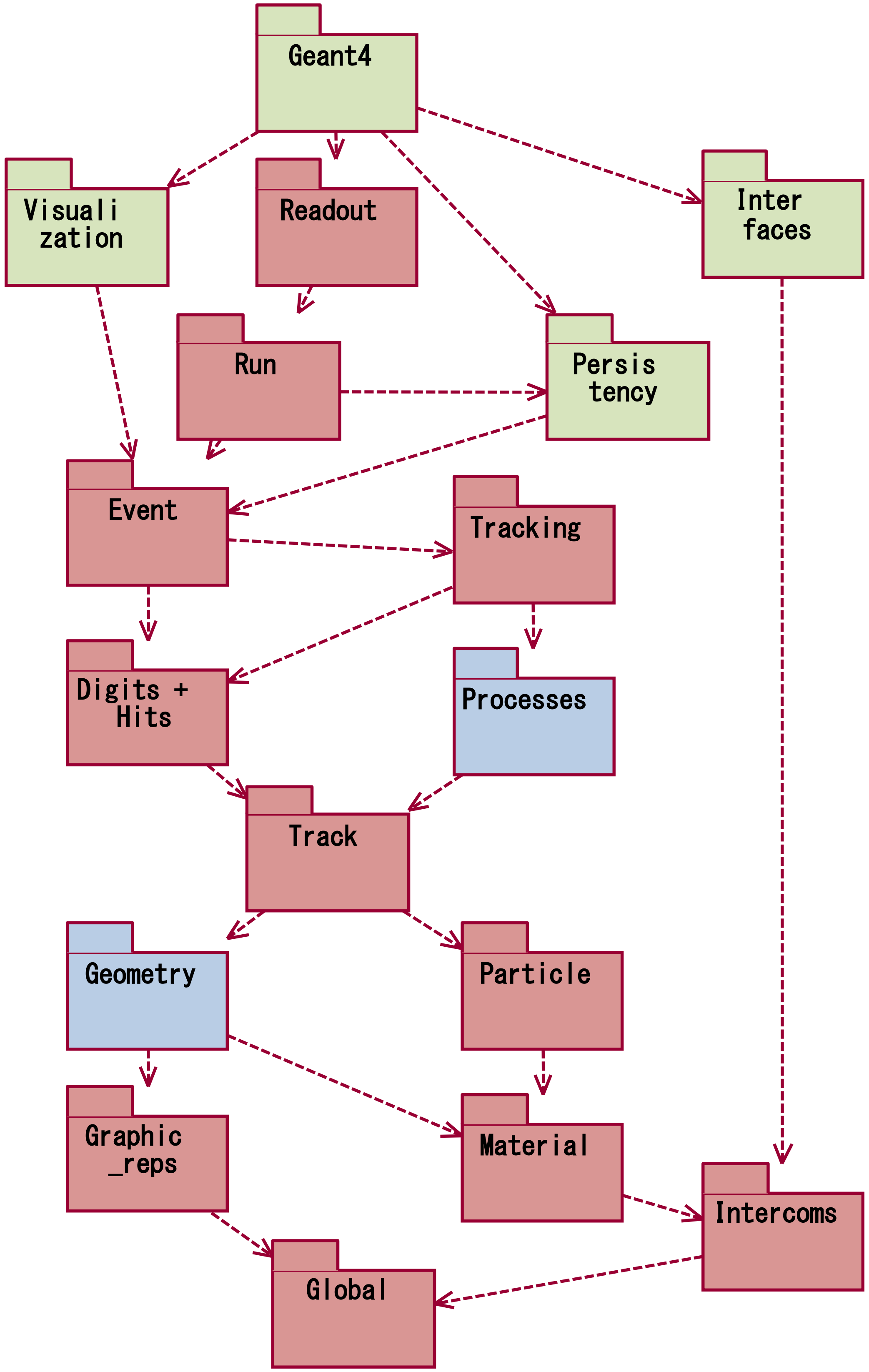
Geant4 is structured in 17 independent categories:

•Geometry and materials

• Track, event, run

•Physics processes (cross sections, final states)

•Auxiliary parts (user interface, visualization)



Build an application

- Geant4 is a toolkit: <u>you must build an application</u>, choosing its tools and following a few rules

- **Must do**:
  - describe the experimental setup materials and geometry
  - select particles and physics models (processes, thresholds) you want to use
  - define the generation of primary particles

- **May also:**
  - control your simulation via interaction with the kernel
  - set visualization
  - extract useful information

Build an application

- **Initialization classes** (invoked at initialization):
  - G4VUserDetectorConstruction
  - G4VUserPhysicsList

- **Action classes** (invoked during an event loop):
  - G4VUserPrimaryGeneratorAction
  - G4UserRunAction
  - G4UserEventAction
  - G4UserStackingAction
  - G4UserTrackingAction
  - G4UserSteppingAction

Mandatory classes

Optional classes

main()

•Geant4 does not provide main(), you have to create it as part of your application

•In main() **you must**:

- Construct G4RunManager

- Set user mandatory classes to RunManager with G4RunManager::SetUserInitialization() and G4RunManager::SetUserAction()

-Optionally, set user action classes and visualization (by default, Geant4 does not take care of retrieving the relevant information)

-Initialize the Geant4 kernel and start a run

-At the end, delete RunManager

main() example

```
//
// --------------------------------------------------------------
//      GEANT 4 simulation of a water tank Cherenkov detector
//
// --------------------------------------------------------------

#include "TankG4SimDetectorConstruction.hh"
#include "TankG4SimPrimaryGeneratorAction.hh"
#include "TankG4SimPhysicsList.hh"

#include "G4RunManager.hh"

int main(int argc,char** argv) {

  // Run manager
  G4RunManager * runManager = new G4RunManager;

  // Mandatory User Initialization classes
  runManager->SetUserInitialization(new TankG4SimDetectorConstruction);
  runManager->SetUserInitialization(new TankG4SimPhysicsList);

  // Mandatory User Action classes
  TankG4SimPrimaryGeneratorAction* PrimGenAct = new TankG4SimPrimaryGeneratorAction();
  runManager->SetUserAction(PrimGenAct);

  //Initialize G4 kernel
  runManager->Initialize();

  // Start a run
  G4int numberOfEvents = 1 ;
  runManager->BeamOn(numberOfEvents);

  delete runManager;

  return EXIT_SUCCESS;
}
```

Useful resources

Geant4 – A Simulation Toolkit

http://www.geant4.org/

S. Agostinelli et al.
Geant4: a simulation toolkit
NIM A, vol. 506, no. 3, pp. 250-303, 2003

J. Allison et al.
Geant4 Developments and Applications
IEEE Trans. Nucl. Sci., vol. 53, no. 1, pp. 270-278, 2006

slide adapted from J. Apostolakis

Useful resources

User Support

Submitted by Anonymous (not verified) on Wed, 06/28/2017 - 11:23

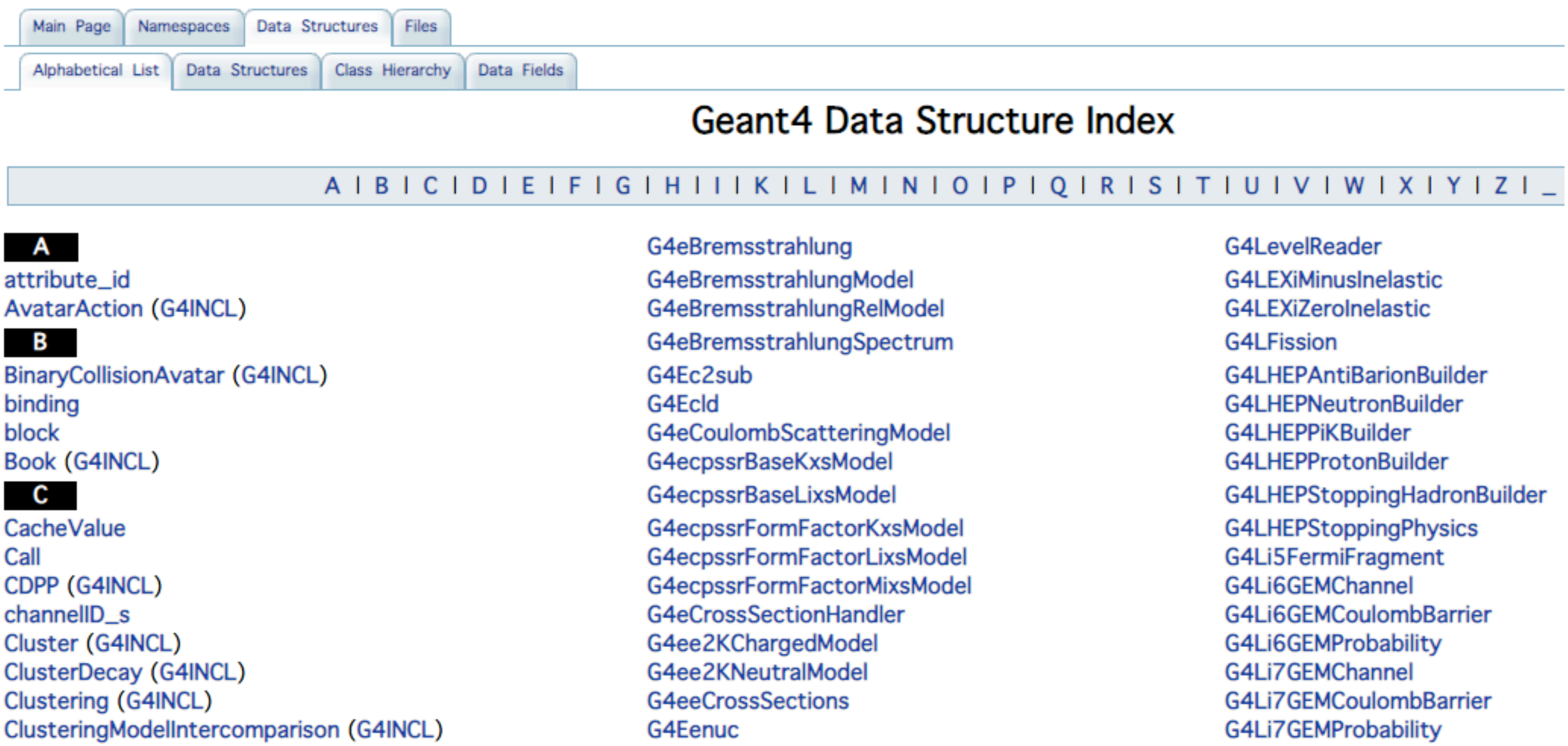
1. Getting started
2. Training courses and materials
3. Source code
   a. Download page
   b. LXR code browser
   c. doxygen documentation
   d. GitHub
   e. GitLab @ CERN
4. Frequently Asked Questions (FAQ)
5. Bug reports and fixes
6. User requirements tracker
7. User Forum
8. Documentation
   a. Introduction to Geant4 [pdf]
   b. Installation Guide: [pdf]
   c. Application Developers [pdf]
   d. Toolkit Developers Guide [pdf]
   e. Physics Reference Manual [pdf]
   f. Physics List Guide [pdf]
9. Examples
10. User Aids
    a. Tips for improving CPU performance
11. Contact Coordinators & Contact Persons

geant4.web.cern.ch/support

Useful resources

- doxygen documentation allows to access the Geant4 class index and browse the source code

http://www.apc.univ-paris7.fr/~franco/g4doxy/html/classes.html


Main Page | Namespaces | Data Structures | Files

Alphabetical List | Data Structures | Class Hierarchy | Data Fields

Geant4 Data Structure Index

A | B | C | D | E | F | G | H | I | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | _

A
attribute_id
AvatarAction (G4INCL)
B
BinaryCollisionAvatar (G4INCL)
binding
block
Book (G4INCL)
C
CacheValue
Call
CDPP (G4INCL)
channelID_s
Cluster (G4INCL)
ClusterDecay (G4INCL)
Clustering (G4INCL)
ClusteringModelIntercomparison (G4INCL)

G4eBremsstrahlung
G4eBremsstrahlungModel
G4eBremsstrahlungRelModel
G4eBremsstrahlungSpectrum
G4Ec2sub
G4EcId
G4eCoulombScatteringModel
G4ecpssrBaseKxsModel
G4ecpssrBaseLixsModel
G4ecpssrFormFactorKxsModel
G4ecpssrFormFactorLixsModel
G4ecpssrFormFactorMixsModel
G4eCrossSectionHandler
G4ee2KChargedModel
G4ee2KNeutralModel
G4eeCrossSections
G4Eenuc

G4LevelReader
G4LEXiMinusInelastic
G4LEXiZeroInelastic
G4LFission
G4LHEPAntiBarionBuilder
G4LHEPNeutronBuilder
G4LHEPPiKBuilder
G4LHEPProtonBuilder
G4LHEPStoppingHadronBuilder
G4LHEPStoppingPhysics
G4Li5FermiFragment
G4Li6GEMChannel
G4Li6GEMCoulombBarrier
G4Li6GEMProbability
G4Li7GEMChannel
G4Li7GEMCoulombBarrier
G4Li7GEMProbability

Questions?

Hands-On Session #1

Fetch and analyze the code
Compile and run
Basic user interface

Fetch the code

- Instructions for using git:

https://git02.ncg.ingrid.pt/raul/LIP_Geant4_Course_Braga_2020/wikis/Using-Git-to-access-the-repository

- Fetch the code from the remote repository:

git clone https://git02.ncg.ingrid.pt/raul/LIP_Geant4_Course_Braga_2020.git

- Alternatively, just download it directly from the webpage

Analyze the code

- Go to HandsOn/Spectroscopy

- Check the code structure for:

  - header files
  - source files
  - application file with main()
  - macros

Compile and run

- Compile with make
  - uses the GNUmakefile
  - run the following commands:

```
source ~/Applications/geant4.10.04.p02-install/share/Geant4-10.4.2/geant4make/geant4make.sh
make
```

- (or) Compile with cmake
  - uses the CMakeLists.txt
  - run the following commands:

your geant4-install path

```
mkdir build
cd build
cmake -DCMAKE_PREFIX_PATH==~/Applications/geant4.10.04.p02-install/ ../.
make
```

- Run the executable

```
      executable_command
(or)  ./name_of_the_executable
```

Basic user interface

1) Hard-coded C++
- everything is specified in the source code
- no user interaction

2) Batch session
- list of commands in macro file

3) Interactive session
- real-time input by the user
- by commands or graphical

Basic user interface

1) Hard-coded C++

- in main()

```
//Initialize G4 kernel
runManager->Initialize();

// Start a run
G4int numberOfEvents = 1 ;
runManager->BeamOn(numberOfEvents);

delete runManager;
```

- Start a run by calling BeamOn and specifying the number of events

Basic user interface

2) Batch session

- in main()

```
//Initialize G4 kernel
runManager->Initialize();

// Get the pointer to the User Interface manager
//
G4UImanager * UImanager = G4UImanager::GetUIpointer();
G4String command = "/control/execute ";
G4String fileName = argv[1];
UImanager->ApplyCommand(command+fileName);

delete runManager;
```

- The macro is passed as command-line argument:

./name_of_the_executable   macro

Basic user interface

3) Interactive session

- in main()

```
//Initialize G4 kernel
runManager->Initialize();

// interactive mode : define UI session
G4UIExecutive * ui = new G4UIExecutive(argc,argv);
ui->SessionStart();
delete ui;

delete runManager;
```

- Opens a graphical user interface