A Geant 4 introduction

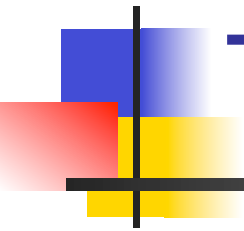
Pablo Cirrone

INFN-LNS

XI Seminar on software for Nuclear, subnuclear and Applied Physics
Porto Conte, May 2014

Where you can find this material?

- Pablo Cirrone and Francesco Romano
  - Researchers at the Italian National Institute for Nuclear Physics (INFN)
  - pablo.cirrone@lns.infn.it francesco.romano@lns.infn.it giuliana.milluzzo@lns.infn.it
- We (as Geant4 Collaboration and as other Geant4 Members) regularly offer tutorials and schools - see Official Geant4 pages
- The official Geant4 web pages www.cern.ch/geant4
- The Italian Geant4 group: http://geant4.lngs.infn.it/



The Monte Carlo method

The Monte Carlo (MC) method: brief history

- Comte du Buffon (1777): needle tossing experiment to calculate π
- Laplace (1886): random points in a rectangle to calculate π
- Fermi (1930): random method to calculate the properties of the newly discovered neutron
- Manhattan project (40's): simulations during the initial development of thermonuclear weapons. Von Neumann and Ulam coined the term "Monte Carlo"
- Exponential growth in the availability of digital computers in 40's-60's
- Berger (1963): first complete coupled electron-photon transport code - ETRAN
- Exponential growth in Medical Physics since the 80's

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION

Number 247 SEPTEMBER 1949 *Volume 44*

THE MONTE CARLO METHOD

NICHOLAS METROPOLIS AND S. ULAM
Los Alamos Laboratory

THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

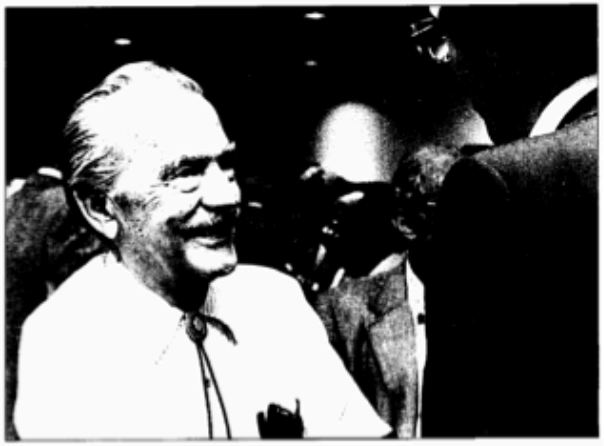
Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)



Nick Metropolis enjoying a break in the quantum Monte Carlo conference, Septem-ber 1985.

With MANIAC: the first electronic digital computer

The Monte Carlo method: a definition

It is a **mathematical approach** using a sequence of random numbers **to solve a problem**

"If we are interested in a parameter of, i.e., an equation:

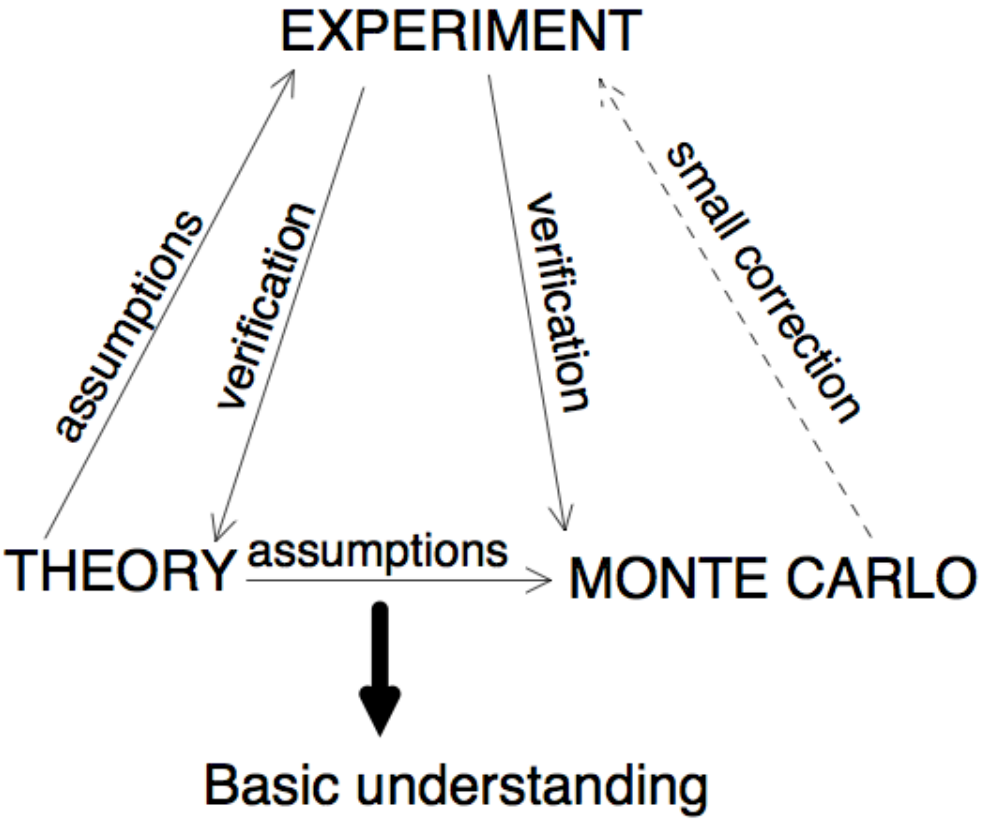
we must construct a big number of this equations, using

different random numbers, and

estimate the parameter and its variance"

A. F. Bielajew, 2001

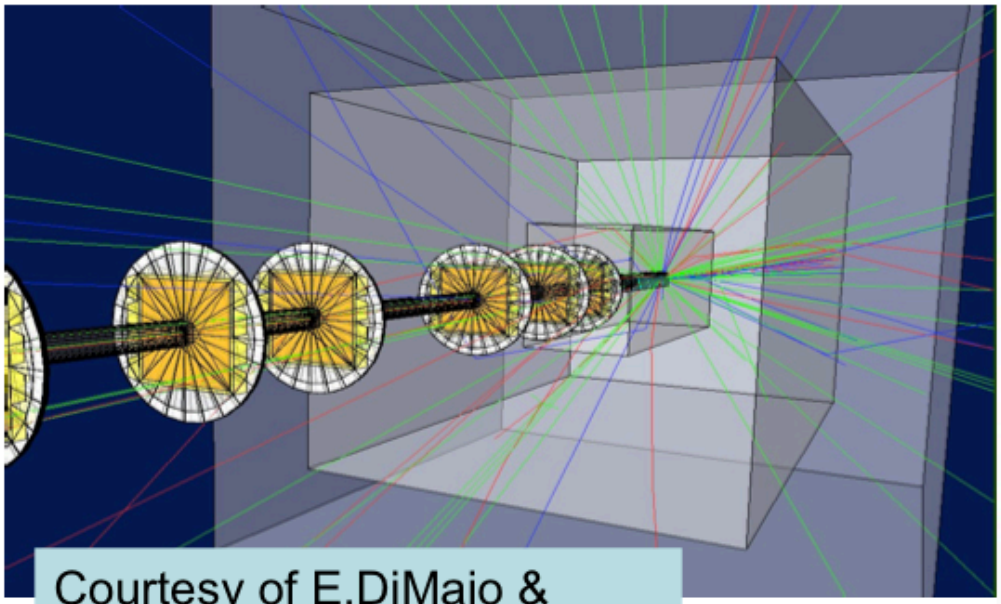
The Monte Carlo method



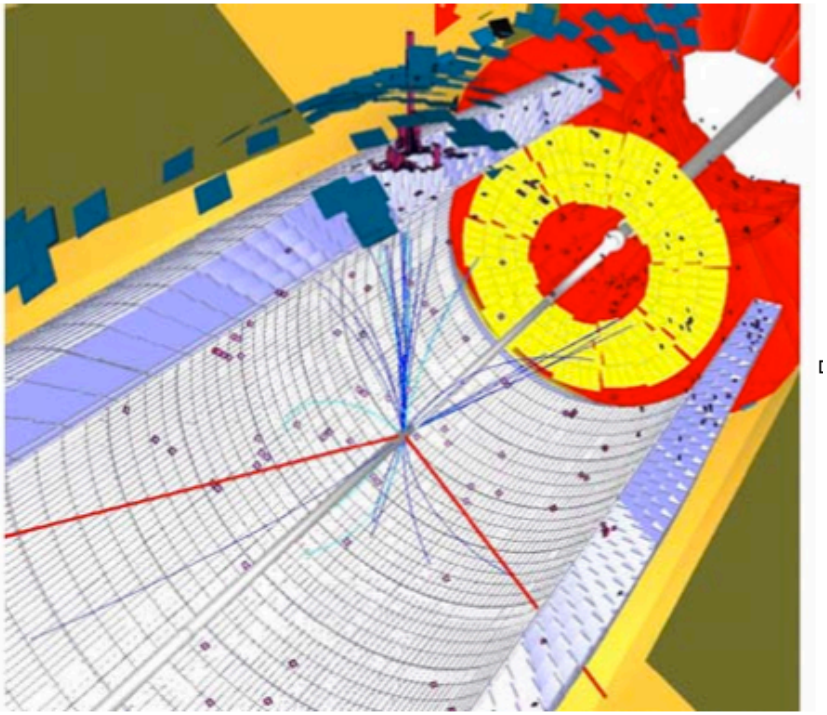
The MC method in Physics

- In some cases the microscopic interactions are well known (electromagnetic interactions, etc.)

  ✓ In these cases MC is usefull to predict the trajectories of high energetic particles through detectors and other complex assemblies of materials
  D.W.O. Rogers, 1996



Courtesy of E.DiMaio & S.Magni (INFN/Milano)

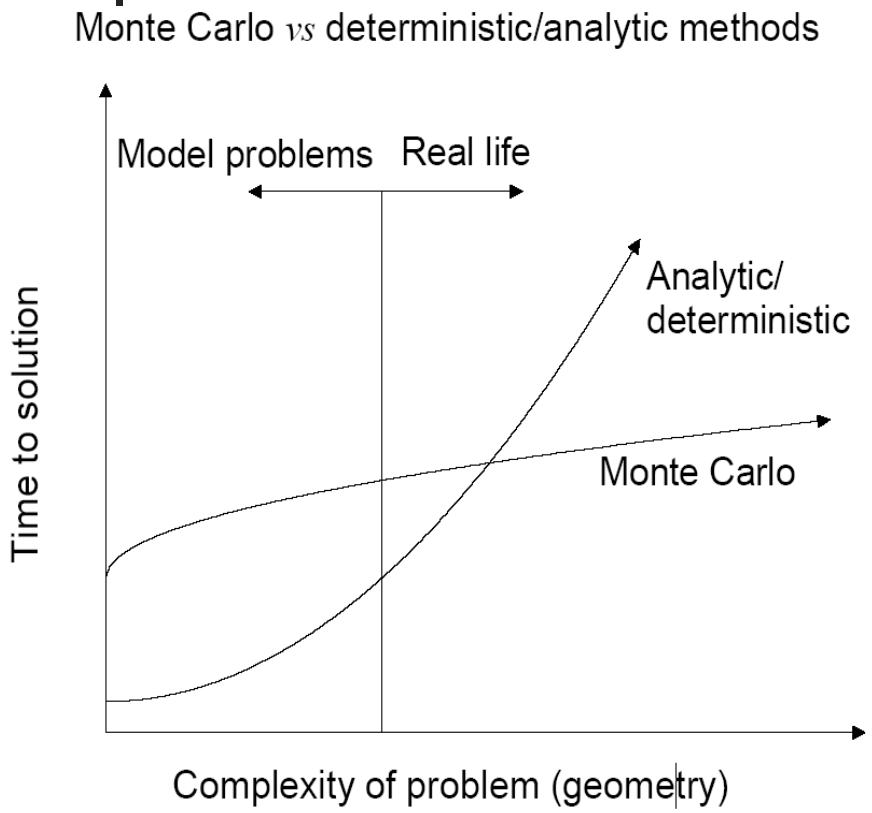


The Monte Carlo method

- Particles are tracked one-by-one, step-by-step and, after a reasonable number, the correct information can be extracted

- MC is very time consuming but
  - ..... sometime necessary and
  - .....with many advantages

MC or analytic approach?



Plot from Alex F. Bielajew, 2001

Mathematical proofs exist demonstrating that

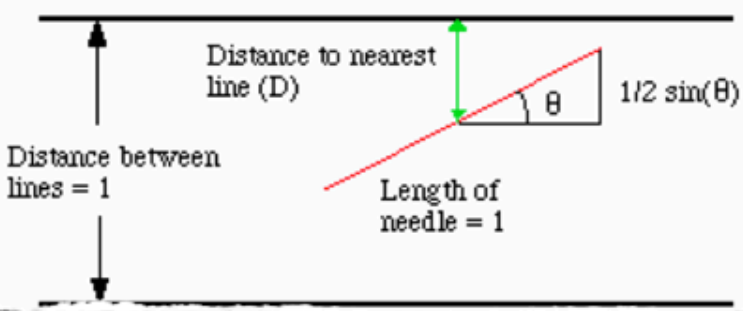
MC is the most efficient way of estimate quantity in 3D when compared to first-order deterministic method

The Buffon experiment: The MC approach for the π estimation

Two variables: θ and D

$$0 \le \vartheta \le \pi$$

$$0 \le D \le \frac{1}{2}$$



Georges Louis Leclerc
Comte de Buffon
(07.09.1707.-16.04.1788.)

The needle will hit the line if the closest distance to a line D is

$$D \le \frac{1}{2}\sin(\vartheta)$$

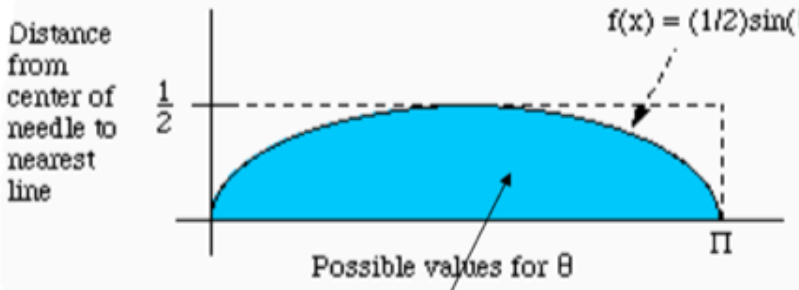
The Buffon experiment: The MC approach for the π estimation

The probability of an hit is the ratio of the blue area (S_{blue}) to the entire rectangle R

$$S_{blue} = \int_0^{\pi} \frac{1}{2}\sin(\vartheta) = 1$$

$$R = \frac{1}{2} \cdot \pi$$

$$\frac{S_{blue}}{R} = \frac{2}{\pi}$$



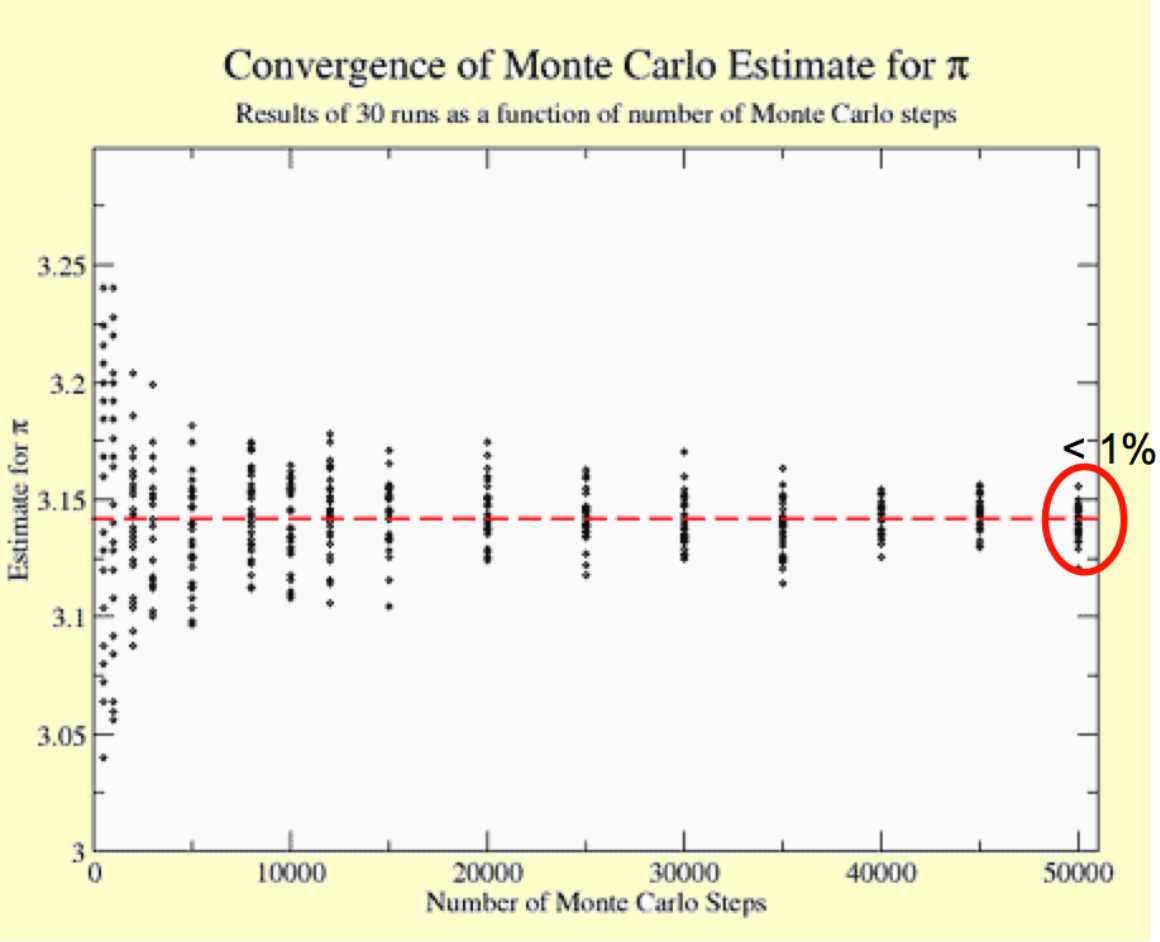
$$D \le \frac{1}{2}\sin(\vartheta)$$

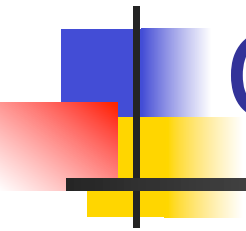
N_0 times the needle was shot

N times the needle hit the line

$$\frac{N}{N_0} = \frac{2}{\pi}; \rightarrow \pi = 2 \cdot \frac{N_0}{N}$$

The Buffon experiment: The Monte Carlo approach for the π estimation





Geant4 and the Geant4 Collaboration

Monte Carlo codes on the market

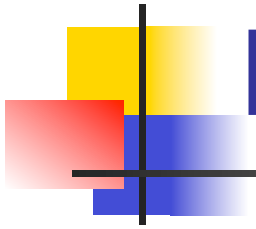
- MCNP (neutrons mainly)
- Penelope (e- and gamma)
- PETRA (protons)
- EGSnrc (e- and gammas)
- PHIT (protons/ions)
- FLUKA (any particle)

Geant4

-GEometry ANd Traking

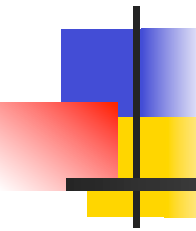
-Geant4 - a simulation toolkit Nucl. Inst. and Methods Phys. Res. A, **506** 250-303

-Geant4 developments and applications
Transaction on Nuclear Science **53**, 270-278

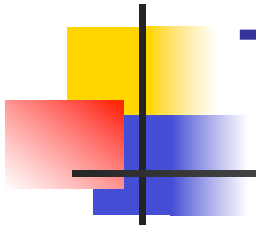


Facts about Geant 4

- Developed by an International Collaboration
  - Established in 1998
  - Approximately 100 members, from Europe, US and Japan
  - **http://geant4.cern.ch**
- Written in C++ language
  - Takes advantage from the Object Oriented software technology
- Open source
- Typically two releases per year
  - Major release, minor release, beta release

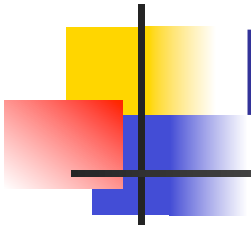


Basic concept of Geant4



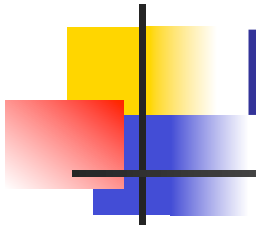
Toolkit and User Application

- Geant4 is a **toolkit** (= a collection of tools)
  - i.e. you cannot "run" it out of the box
  - You must **write an application**, which uses Geant4 tools
- Consequences:
  - There are no such concepts as "Geant4 defaults"
  - You must provide the necessary information to configure your simulation
  - You must deliberately choose which Geant4 tools to use
- Guidance: many **examples** are provided
  - **Basic/Novice Examples**: overview of Geant4 tools
  - **Advanced Examples**: Geant4 tools in real-life applications



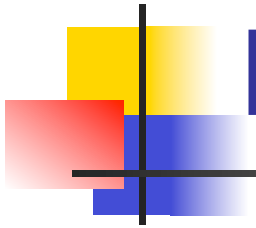
Basic concepts

- What you **MUST** do:
  - Describe your **experimental set-up**
  - Provide the **primary particles** input to your simulation
  - Decide which **particles** and **physics models** you want to use out of those available in Geant4 and the precision of your simulation (cuts to produce and track secondary particles)
- You **may also want**
  - To interact with Geant4 kernel to **control** your simulation
  - To **visualise** your simulation configuration or results
  - To produce **histograms, tuples** etc. to be further analysed



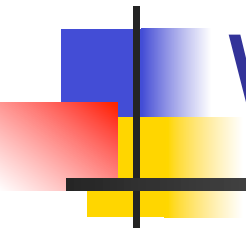
Main Geant4 capabilities

- **Transportation of a particle 'step-by-step'** taking into account all the possible interactions with materials and fields
- **The transport ends** if the particle
  - is slowed down to zero kinetic energy (and it doesn't have any interaction at rest)
  - disappears in some interaction
  - reaches the end of the simulation volume
- Geant4 allows the User to access the transportation process and retrieve the results (USER ACTIONS)
  - at the beginning and end of the transport
  - at the end of each step in transportation
  - if a particle reaches a sensitive detector
  - Others…



News from version 10.0

- Released on December 6th, 2013
- Supports multi-thread approach for multi-core machines
  - Simulation is automatically split on an event-by-event basis
    - different events are processed by different cores
  - **Unique** copy (master) of geometry and physics
    - All cores have them as read-only (saves memory)
- Backwards compatible with the sequential mode
  - The MT programming requires some care
  - Need to avoid conflicts between threads
  - Merge information at the end coming from the cores

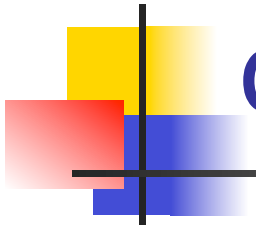


Who/why is using Geant4?

Experiments and MC

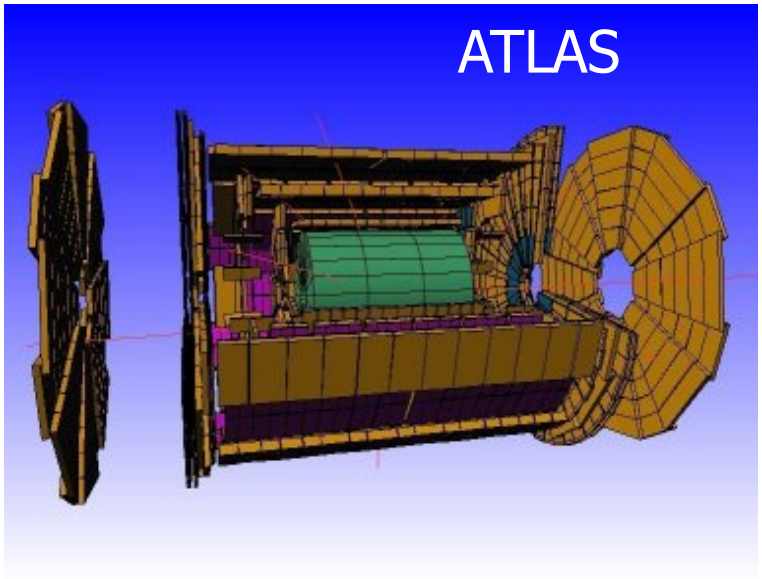
- In my knowledge, **all experiments** have a (more or less detailed) full-scale Monte Carlo simulation
- Design phase
  - Evaluation of background
  - Optimization of setup to maximize scientific yield
    - Minimize background, maximize signal efficiency
- Running/analysis phase
  - Support of data analysis (e.g. provide efficiency for signal, background, coincidences, tagging, …).
    - often, Monte Carlo is the only way to convert *relative rates* (events/day) in *absolute yields*

Why Geant4 is a common choice in the market



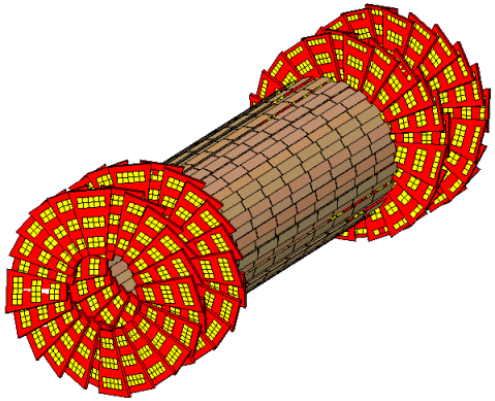
- Open source and object oriented/C++
  - No black box
  - Freely available on all platforms
  - Can be easily extended and customized by using the existing interfaces
    - New processes, new primary generators, interface to ROOT analysis, …
- Can handle complex geometries
- Regular development, updates, bug fixes and validation
- Good physics, customizable per use-cases
- End-to-end simulation (all particles, including optical photons)

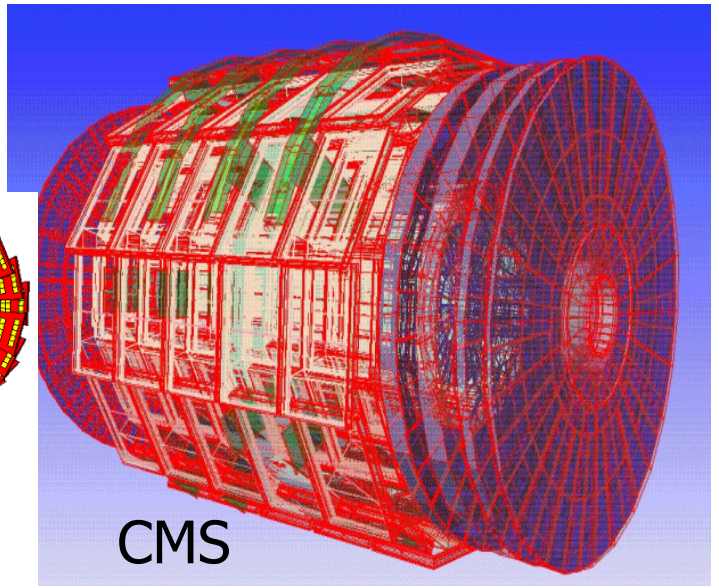
LHC @ CERN



- All four big **LHC experiments** have a Geant4 simulation
  - M of volumes
  - Physics at the TeV scale

- Benchmark with test-beam data
- Key role for the Higgs searches

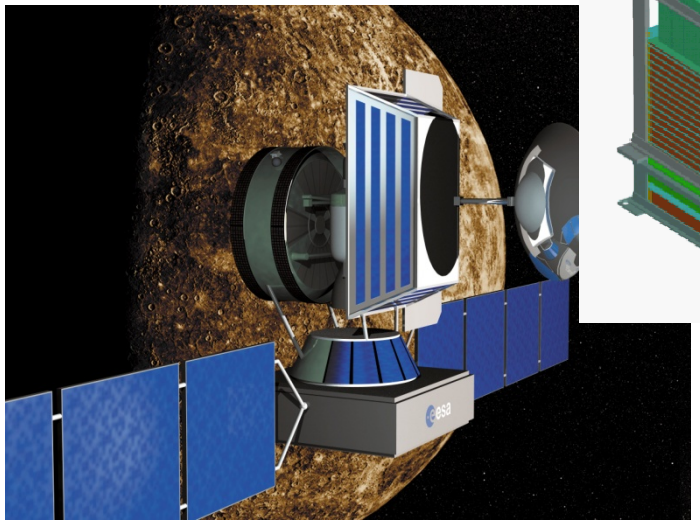


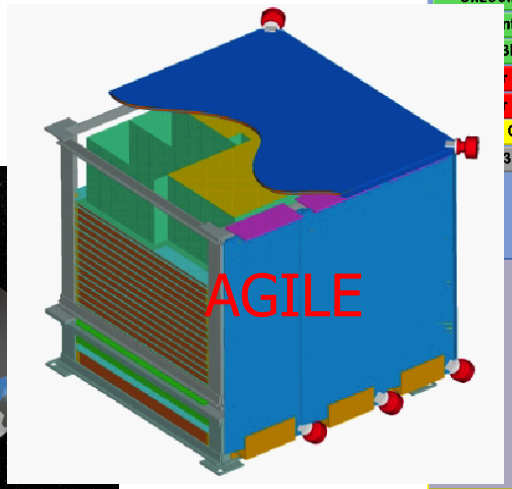


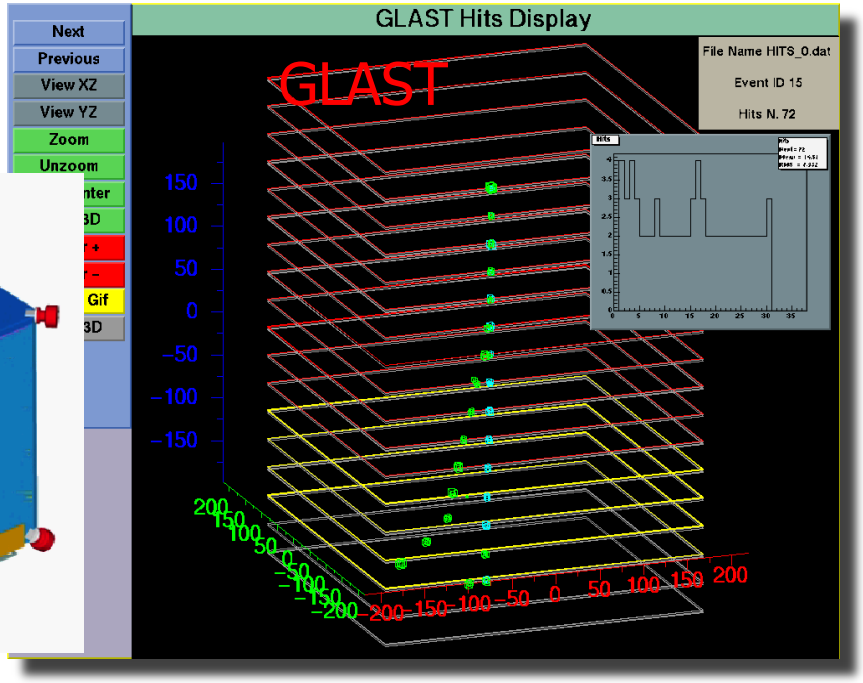
Space applications

- Satellites (γ astrophysics, planetary sciences)
- Funding from ESA

Typical telescope:
Tracker
Calorimeter
Anticoincidence

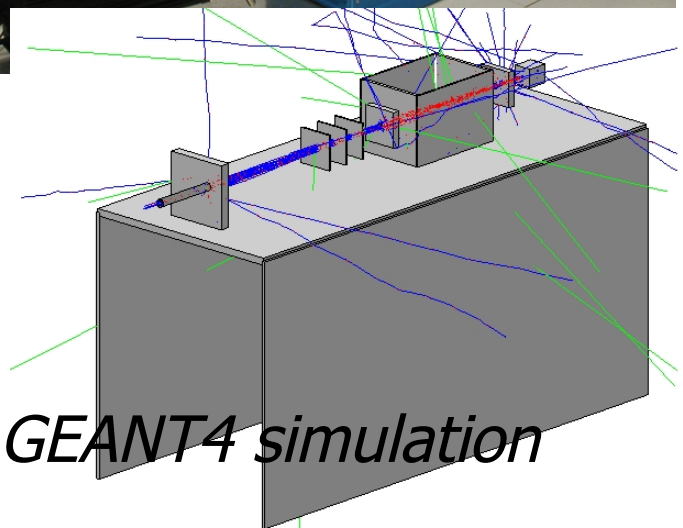






Medical applications

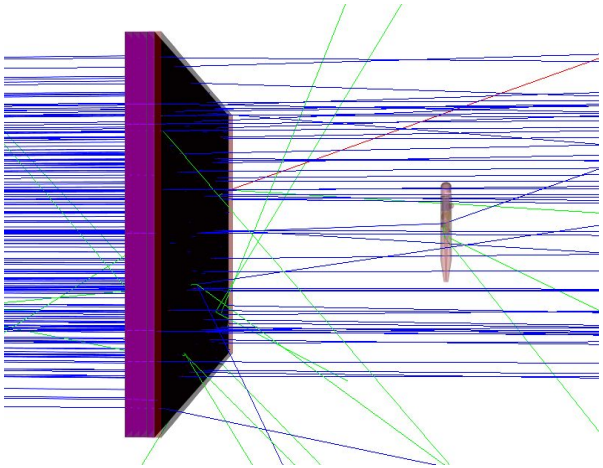
Proton-therapy beam line

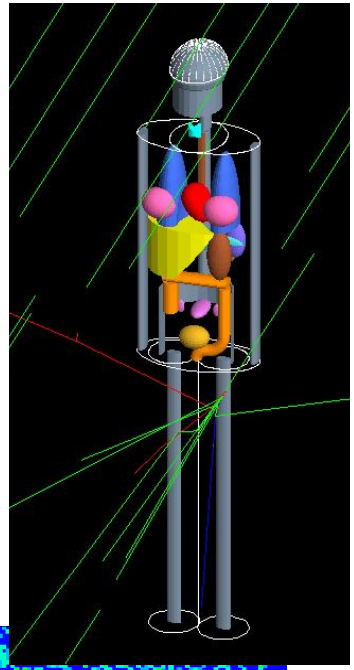

GEANT4 simulation

- **Treatment planning** for hadrontherapy and proton-therapy systems
  - Goal: deliver dose to the tumor while sparing the healthy tissues
  - Alternative to less-precise (and commercial) TP software
- Medical imaging
- Radiation fields from medical accelerators and devices
  - medical_linac
  - gamma-knife
  - brachytherapy

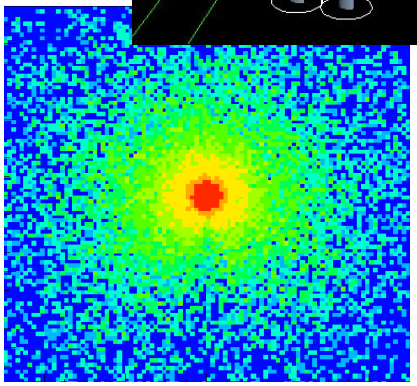
Dosimetry with Geant4

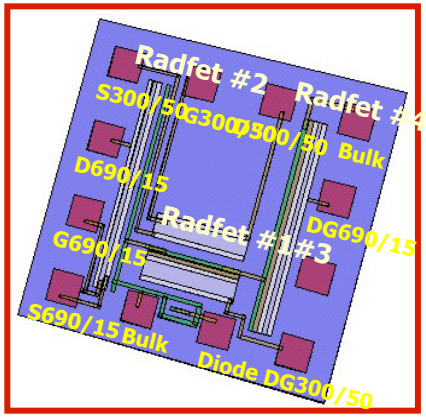
Space science



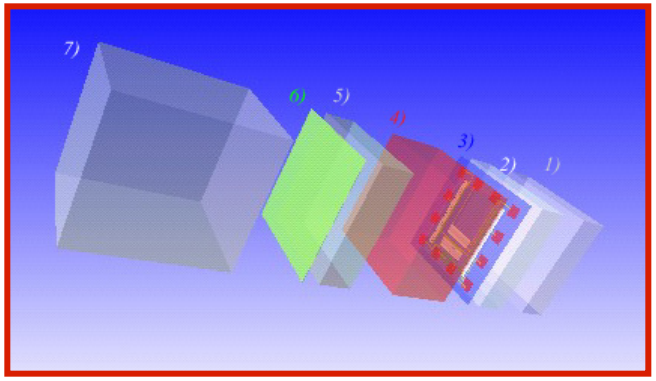


Radiotherapy





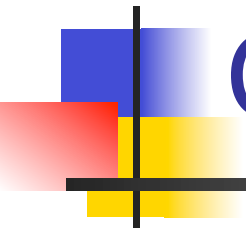
Effects on electronics components



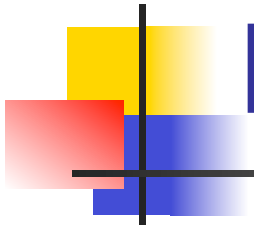
Minimum software requirements

- **C++**
  - A basic knowledge is required being Geant4 a collection of C++ libraries
  - It is complex but also no C++ experts can use Geant4
- **Object oriented technology (OO)**
  - Very basic knowledge
  - Expertise needed for the development of complex applications
- **Unix/Linux**
  - These are the standard OSs for Geant4 and a basic knowledge is required
  - Principal shell commands
  - How to compile a program, How to install from source code

The (conceptual) recipe for a Geant4-based application



Interaction with the Geant4 kernel



- Geant4 design provides **tools** for a user application
  - To tell the kernel about your simulation configuration
  - To interact with Geant4 kernel itself
- Geant4 tools for user interaction are **base classes**
  - You create **your own concrete class** derived from the base classes → interface to the Geant4 kernel
  - Geant4 kernel handles your own derived classes transparently through their base class interface (polymorphism)

User Classes (<= 9.6)

Initialisation classes

Invoked at the initialization

- G4VUserDetectorConstruction
- G4VUserPhysicsList

Classes having name starting with G4V are abstract classes (containing purely virtual methods)

Action classes

Invoked during the execution loop

- G4VUserPrimaryGeneratorAction
- G4UserRunAction
- G4UserEventAction
- G4UserTrackingAction
- G4UserStackingAction
- G4UserSteppingAction

User Classes (from 10.0)

Initialisation classes

Invoked at the initialization

- G4VUserDetectorConstruction
- G4VUserPhysicsList

Global: only one instance of them exists in memory, shared by all threads (**readonly**). Managed only by the master thread.

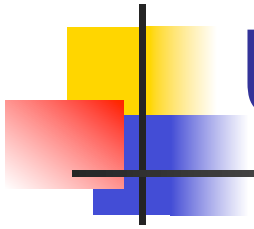
Action classes

Invoked during the execution loop

- G4VUserActionInitialization
  - G4VUserPrimaryGeneratorAction
  - G4UserRunAction (*)
  - G4UserEventAction
  - G4UserTrackingAction
  - G4UserStackingAction
  - G4UserSteppingAction

Local: an instance of each action class exists **for each thread**.

(*) Two RunAction's allowed: one for master and one for threads

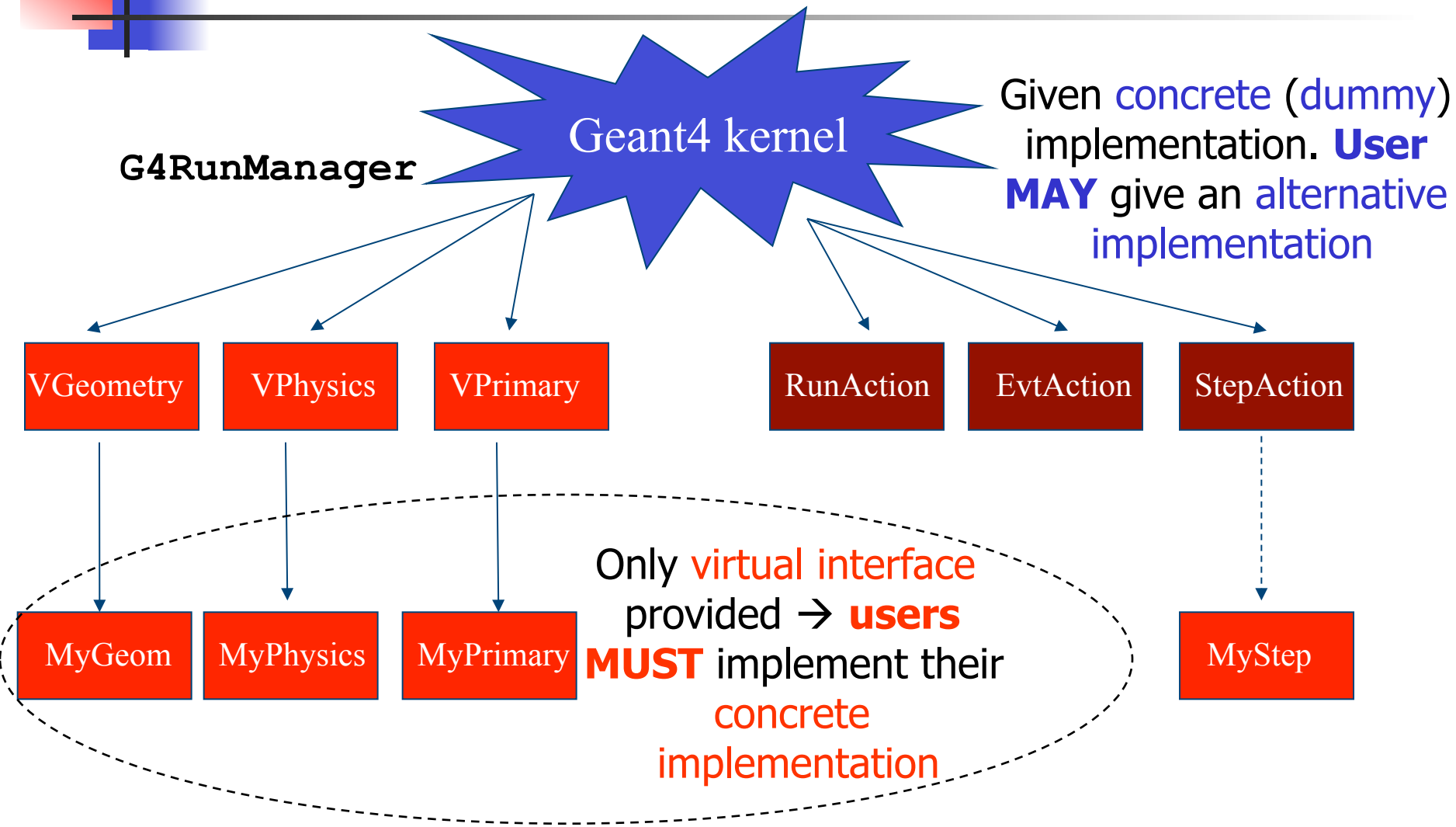


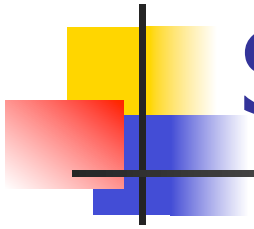
User Classes - 2

Mandatory classes in ANY Geant4 User Application

- G4VUserDetectorConstruction
  describe the experimental set-up
- G4VUserPhysicsList
  select the physics you want to activate
- G4VUserPrimaryGeneratorAction
  generate primary events

Geant4 concept

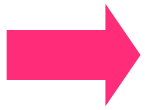




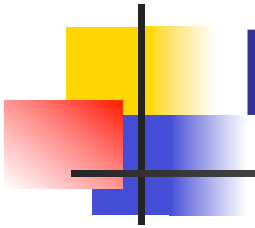
Select physics processes

- Geant4 doesn't have any default particles or processes
- Derive your own concrete class from the **`G4VUserPhysicsList`** abstract base class
  - define all necessary particles
  - define all necessary processes and assign them to proper particles
  - define γ/δ production thresholds (in terms of range)
- Pure virtual methods of **`G4VUserPhysicsList`**

ConstructParticles()
ConstructProcesses()
SetCuts()



must be implemented by the user in his/her concrete derived class



Physics Lists

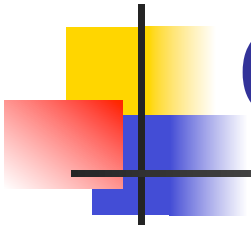
- Geant4 doesn't have any default particles or processes
- *Partially true*: there is no default, but there are a set of **"ready-for-use" physics lists** released with Geant4, tailored to different use cases. Mix and match:
  - Different sets of hadronic models (depending on the energy scale and modeling of the interactions)
  - Different options for neutron tracking
    - Do we need (CPU-intensive) description of thermal neutrons, neutron capture, etc?
  - Different options for EM physics
    - Do you need (CPU-intensive) precise description at the low-energy scale (< 1 MeV)? E.g. fluorescence, Doppler effects in the Compton scattering, Auger emission, Rayleigh diffusion
    - Only a waste of CPU time for LHC, critical for many low-background experiments

Optional user classes - 1

- Five concrete base classes whose virtual member functions the user may override to gain control of the simulation at various stages
  - G4User**Run**Action
  - G4User**Event**Action
  - G4User**Tracking**Action
  - G4User**Stacking**Action
  - G4User**Stepping**Action

e.g. actions to be done at the beginning and end of each event

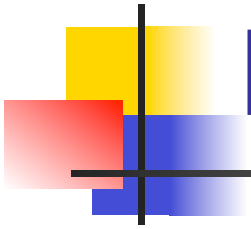
- Each member function of the base classes has a dummy implementation (**not** purely virtual)
  - Empty implementation: does nothing



Optional user classes - 2

- The user may implement the member functions he desires in his/her derived classes
  - E.g. one may want to perform some action at each tracking step
- Objects of user action classes must be **registered** to **`G4RunManager`**

```
runManager->
   SetUserAction(new MyEventActionClass);
```



Methods of user classes - 1

G4UserRunAction

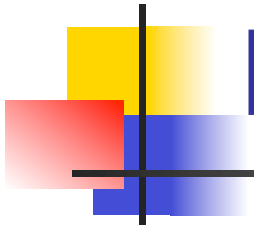
- `BeginOfRunAction(const G4Run*)` // book histos
- `EndOfRunAction(const G4Run*)` //store histos

G4UserEventAction

- `BeginOfEventAction(const G4Event*)` //initialize event
- `EndOfEventAction (const G4Event*)` // analyze event

G4UserTrackingAction

- `PreUserTrackingAction(const G4Track*)`
  //decide to store/not store a given track
- `PostUserTrackingAction(const G4Track*)`



Methods of user classes - 2

G4UserSteppingAction

- `UserSteppingAction(const G4Step*)`

  //kill, suspend, pospone the track, draw the step, …

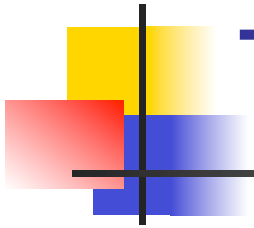
G4UserStackingAction

- `PrepareNewEvent()` //reset priority control
- `ClassifyNewTrack(const G4Track*)`

  // Invoked when a new track is registered (e.g. kill, pospone)

- `NewStage()`

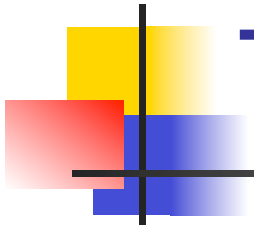
  // Invoked when the Urgent stack becomes empty (re-classify, abort event)



The main() program - 1

- Geant4 does **not** provide the **main()**
  - Geant4 is a toolkit!
  - The main() is part of the user application

- In his/her main(), the user **must**
  - construct **`G4RunManager`** (or his/her own derived class)
  - notify the **`G4RunManager`** mandatory user classes derived from
    - **`G4VUserDetectorConstruction`**
    - **`G4VUserPhysicsList`**
    - **`G4VUserPrimaryGeneratorAction`**
  - In MT mode, use **`G4MTRunManager`**
    - **`G4VUserDetectorConstruction`**
    - **`G4VUserPhysicsList`**
    - **`G4VUserActionInitialization`** (takes care of Primary)



The main() program - 2

- The user may define in his/her main()
  - optional user action classes
  - VisManager, (G)UI session
- The user also has to take care of retrieving and saving the relevant information from the simulation (Geant4 will not do that by default)
- Don’t forget to delete the G4RunManager at the end

An example of (sequential) main()

```
{
  …
  // Construct the default run manager
  G4RunManager* runManager = new G4RunManager;
  // Set mandatory user initialization classes
  MyDetectorConstruction* detector = new MyDetectorConstruction;
  runManager->SetUserInitialization(detector);
  MyPhysicsList* physicsList = new MyPhysicsList;
  runManager->SetUserInitialization(myPhysicsList);
  // Set mandatory user action classes
  runManager->SetUserAction(new MyPrimaryGeneratorAction);
  // Set optional user action classes
  MyEventAction* eventAction = new MyEventAction();
   runManager->SetUserAction(eventAction);
  MyRunAction* runAction = new MyRunAction();
   runManager->SetUserAction(runAction);
  …
}
```

An example of (MT) main()

```
{
  …
  // Construct the default run manager
  G4MTRunManager* runManager = new G4MTRunManager;
 // Set mandatory user initialization classes
  MyDetectorConstruction* detector = new MyDetectorConstruction;
  runManager->SetUserInitialization(detector);
  MyPhysicsList* physicsList = new MyPhysicsList;
  runManager->SetUserInitialization(myPhysicsList);

  // Set mandatory user action classes
  runManager->SetUserAction(new MyActionInitialization);

  …
}
```

MyActionInitialization (MT mode)

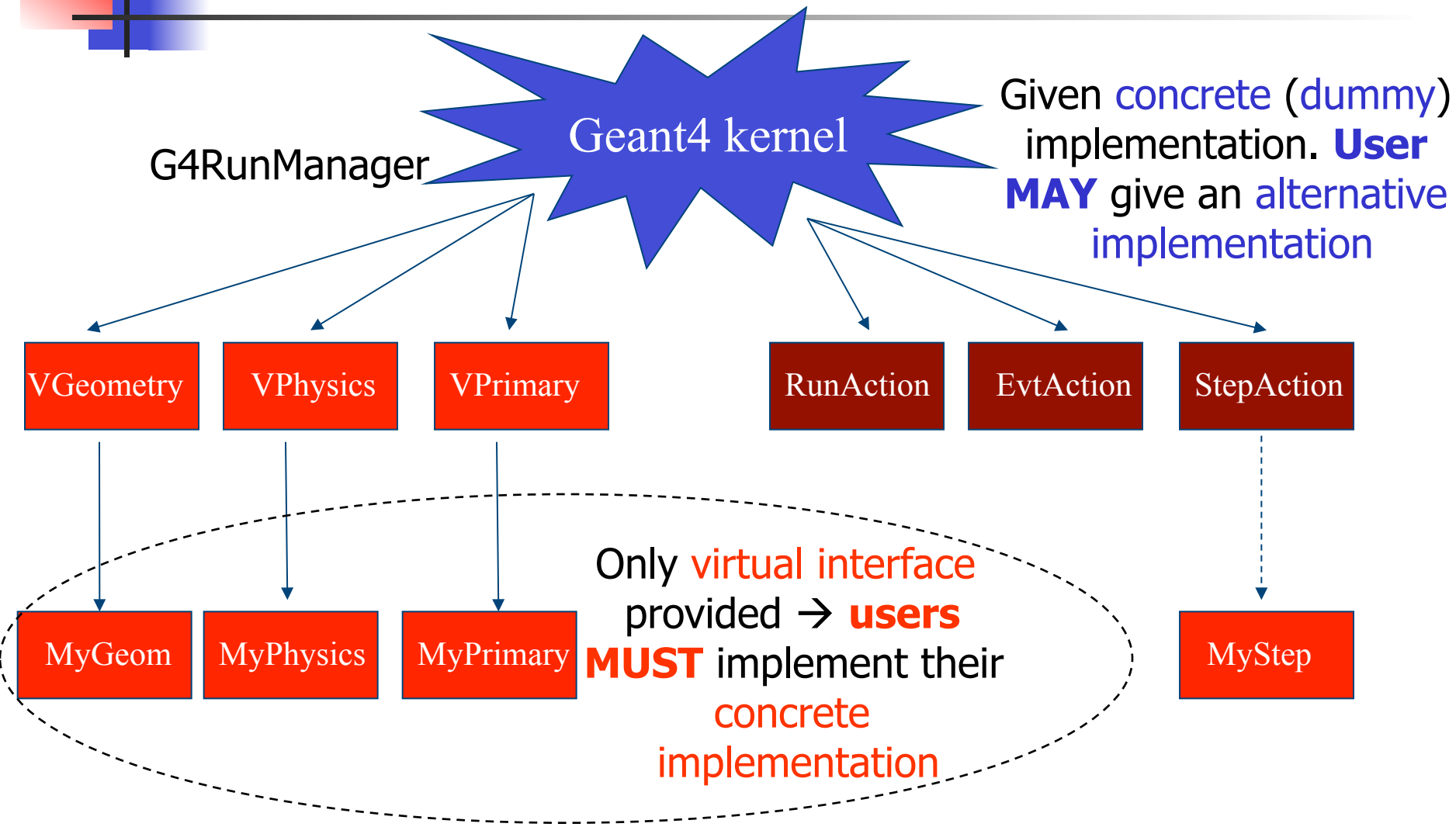
- Register thread-local user actions

```
void MyActionInitialization::Build() const
{
  //Set mandatory classes
  SetUserAction(new MyPrimaryGeneratorAction());
  // Set optional user action classes
  SetUserAction(new MyEventAction());
  SetUserAction(newMyRunAction());
}
```

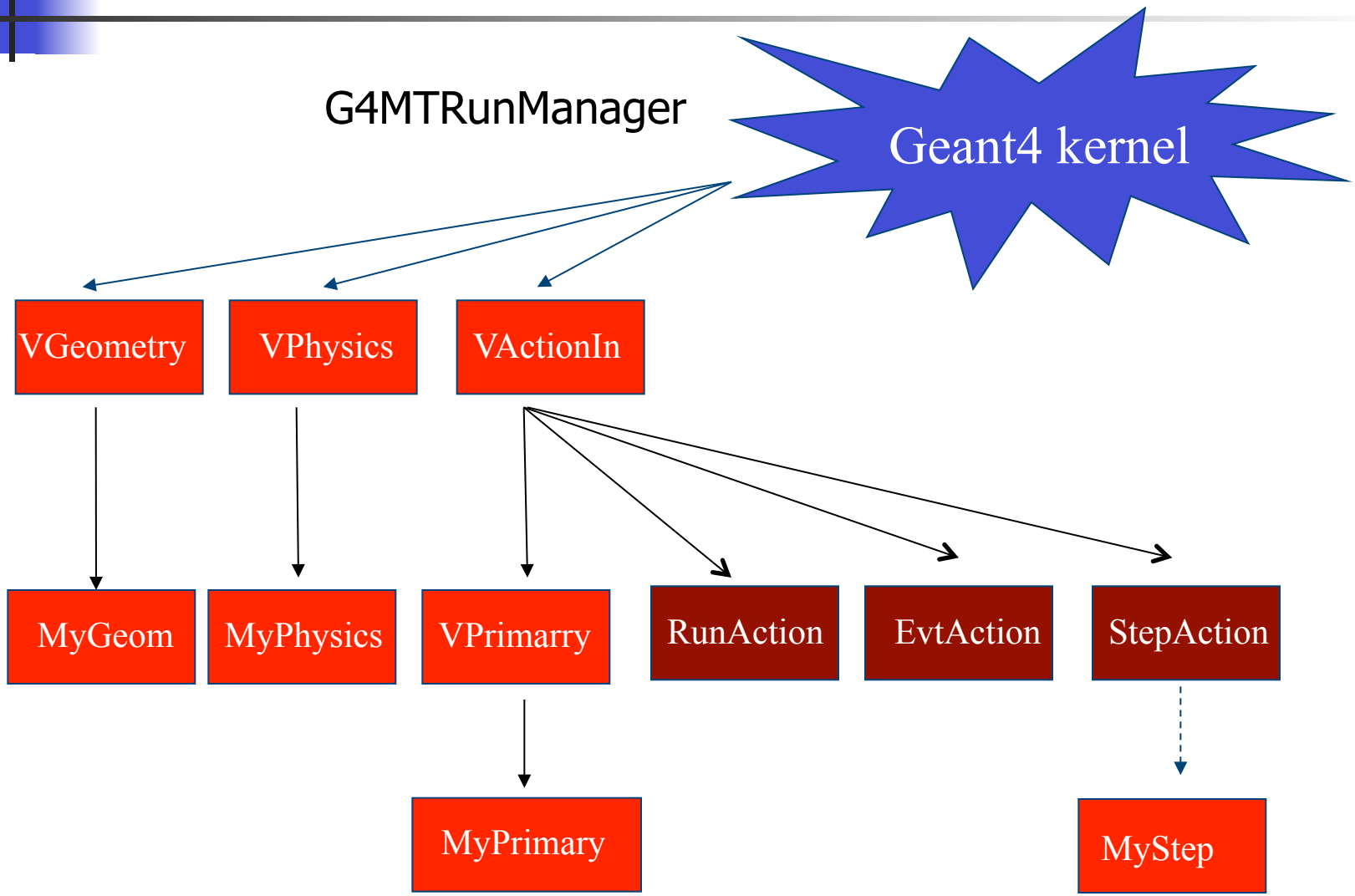
- Register RunAction for the master

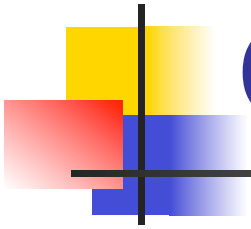
```
void MyActionInitialization::BuildForMaster() const
{
  // Set optional user action classes
SetUserAction(newMyMasterRunAction());
}
```

Geant4 concept



Geant4 concept (MT)



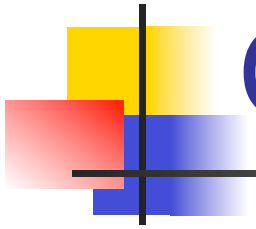


Optional: select (G)UI

- In your **main()**, taking into account your computer environment, instantiate a **G4UIsession** concrete/ derived class provided by Geant4 and invoke its **SessionStart()** method

```
mysession->SessionStart();
```

- Geant4 provides:
  - G4UIterminal
  - csh or tcsh like character terminal
  - G4GAG
  - Opacs
  - G4UIBatch
  - batch job with macro file
  - …



Optional: select visualization

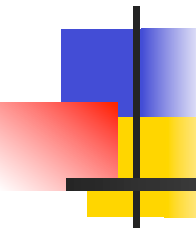
- In your **main()**, taking into account your computer environment, instantiate a **G4VisExecutive** and invoke its **Initialize()** method

- Geant4 provides interfaces to various graphics drivers:
  - DAWN *(Fukui renderer)*
  - WIRED
  - RayTracer *(ray tracing by Geant4 tracking)*
  - OpenGL
  - OpenInventor
  - VRML
  - X11-compliant

General recipe for novice users

Experienced users may do much more, but the conceptual process is still the same…

- Design your application… requires some preliminar **thinking** (what is it supposed to do?)
- Create your derived **mandatory** user classes
  - My**DetectorConstruction**
  - My**PhysicsList**
  - My**PrimaryGeneratorAction**
- Create optionally your derived user action classes
  - My**UserRunAction**, My**UserEventAction**, …
- Create your **main()**
  - Instantiate `G4RunManager` or your own derived MyRunManager
  - Notify the RunManager of your mandatory and optional user classes
  - Optionally initialize your favourite User Interface and Visualization
- That's all!



Installation tips

Geant4 Software Download

Geant4 9.5

released 2 December 2011

The Geant4 source code is freely available. See the licence conditions.

Please read the **Release Notes** before downloading or using this release.

Source files

Please choose the archive best suited to your system and archiving tool:

Download GNU or Linux tar format, compressed using gzip (27Mbytes, 28458437 bytes).
After downloading, gunzip, then unpack using GNU tar.

Download ZIP format (39Mbytes, 40826089 bytes).
After downloading, unpack using e.g. WinZip.

Data files (*)

For specific, optional physics processes some of the following files are required. The file format is compatible with Unix, GNU, and Windows utilities.

Download Neutron data files with thermal cross sections - version 4.0 (381Mbytes, 400001140 bytes) NEW

Download Neutron data files without thermal cross sections - version 0.2 (12Mbytes, 12465281 bytes)

Download Data files for low energy electromagnetic processes - version 6.23 (15Mbytes, 15960390 bytes) NEW

Download Data files for photon evaporation - version 2.2 (7.3Mbytes, 7704178 bytes) NEW

Download Data files for radioactive decay hadronic processes - version 3.4 (716Kbytes, 732861 bytes) NEW

Download Data files for nuclear shell effects in INCL/ABLA hadronic model - version 3.0 (54Kbytes, 54909 bytes)

Download Data files for evaluated neutron cross sections on natural composition of elements - version 1.1 (1.2Mbytes, 1247160 bytes) NEW

Download Data files for shell ionisation cross sections - version 1.3 (4.1Mbytes, 4293607 bytes) NEW

Download Data files for measured optical surface reflectance - version 1.0 (1.2Mbytes, 1257863 bytes)

Pre-compiled Libraries

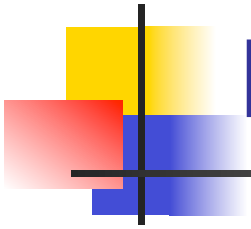
These are compiled with Geant4 default settings and optimization turned on. Please choose according to your system/compiler:

Download compiled using gcc 4.1.2 on Scientific Linux CERN 5 (SLC5, based on Redhat Linux Enterprise 5), 64 bits - (32Mbytes, 33212295 bytes)

Download compiled using gcc 4.2.1 on Mac (MacOSX 10.7), 64 bits - (31Mbytes, 32039379 bytes)

Related Links

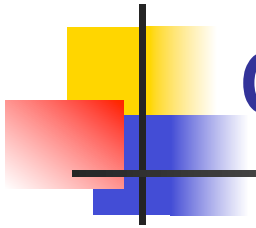
- Geant4-MT **prototype**.
- **Previous Releases** of Geant4 (since release 8.3).
- LXR source code browser.
- Installation Guide tutorials for Linux, Mac and Windows.
- **Windows CygWin installation note**.



Download and installation tips

- You can download the compiled libraries of Geant4 but **the compilation in your computer is strongly suggested**
- **Download the source file** from the Geant4 web site
- Two ways to proceed:
  - Using cmake via terminal
  - Using the GUI version of cmake

cmake Geant4 installation

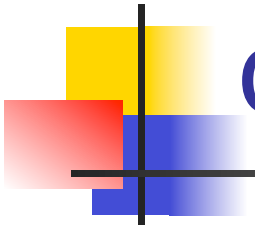


- **cmake** version greater than 2.8.3
- Locate the **source folder**
  Ex: **`/home/Username/geant4-09-05`**
- Create the **build folder**
  Ex: **`/home/Username/geant4-09-05-build`**
- Create the **install folder**
  Ex: **`/home/Username/geant4-09-05-install`**
- **`cmake -DCMAKE_INSTALL_PREFIX=/home/Username/geant4-09-05-install/`**
- Define and/or activate the **additional features/package you require** using the same cmake interface
- make -jN
- make install

cmake Geant4 installation

```
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4Physics2DVector.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4Physics2DVectorCache.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4Pow.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4SliceTimer.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4StateManager.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4Timer.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4UnitsTable.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4VExceptionHandler.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4VNotifier.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4VStateDependent.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4coutDestination.cc.o
[  4%] Building CXX object source/global/CMakeFiles/G4global.dir/management/src/G4ios.cc.o
Linking CXX shared library ../../outputs/library/Darwin-g++/libG4global.dylib
[  4%] Built target G4global
Scanning dependencies of target G4analysis
Scanning dependencies of target G4intercoms
[  4%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIaliasList.cc.o
[  4%] Building CXX object source/analysis/CMakeFiles/G4analysis.dir/src/G4AnalysisVerbose.cc.o
[  4%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIbatch.cc.o
[  4%] Building CXX object source/analysis/CMakeFiles/G4analysis.dir/src/G4CsvAnalysisManager.cc.o
[  4%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWith3Vector.cc.o
[  4%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWith3VectorAndUnit.cc.o
[  4%] Building CXX object source/analysis/CMakeFiles/G4analysis.dir/src/G4VAnalysisManager.cc.o
[  5%] Building CXX object source/analysis/CMakeFiles/G4analysis.dir/src/G4RootAnalysisManager.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWithABool.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWithADouble.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWithADoubleAndUnit.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWithAString.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWithAnInteger.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcmdWithoutParameter.cc.o
[  5%] Building CXX object source/intercoms/CMakeFiles/G4intercoms.dir/src/G4UIcommand.cc.o
```

cmake Geant4 installation options

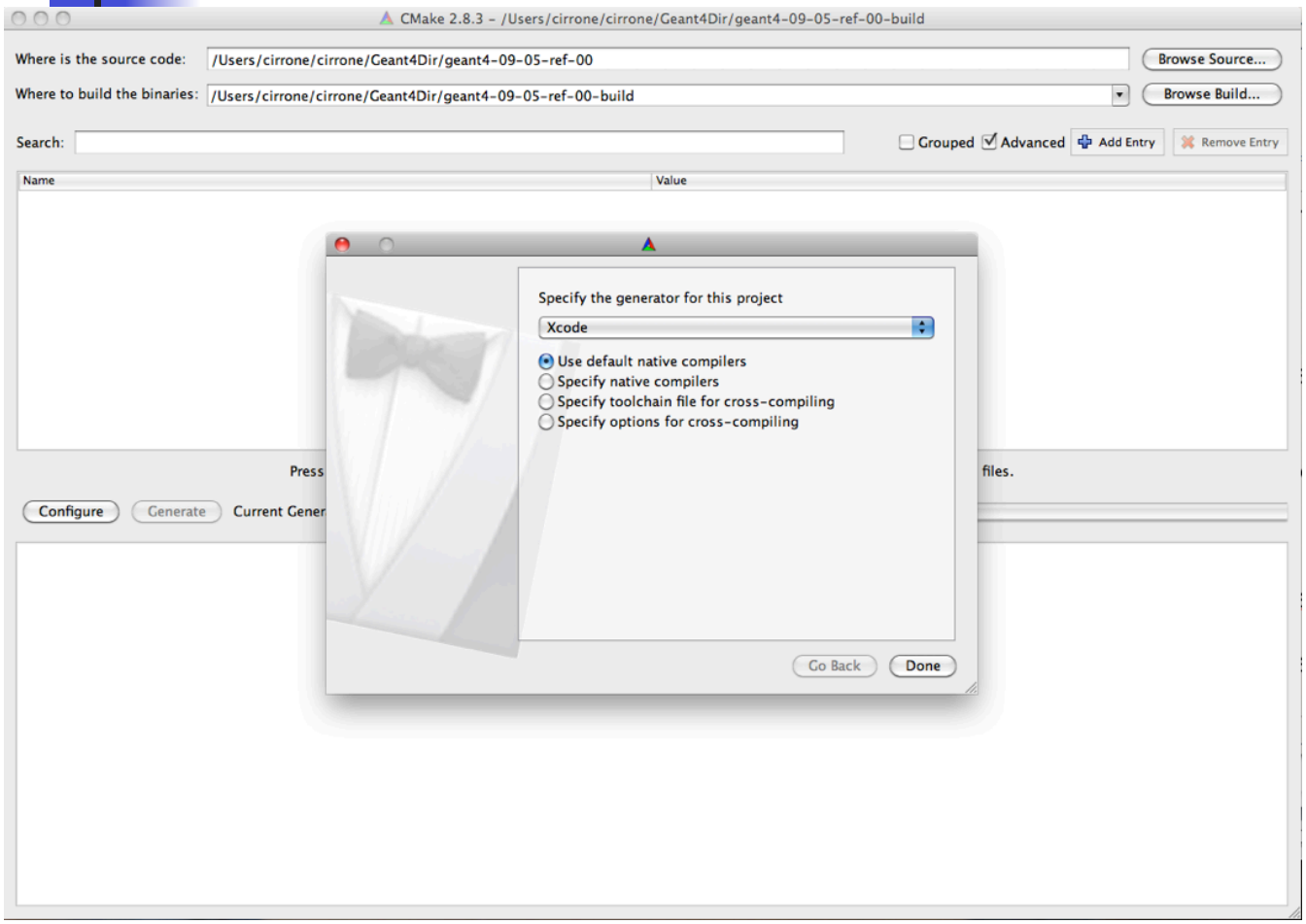


- If GEANT4_INSTALL_DATA is ON
  the additional external data libraries are automatically downloaded (e.g. –DGEANT4_INSTAL_DATA=ON)
- If GEANT4_INSTALL_EXAMPLES is ON
  Examples are installed
- If GEANT4_USE_SYSTEM_CLHEP is ON
  external CLHEP are searched
- See documentation for details for the complete variables list and explanation

cmake Geant4 installation

```
Linking CXX shared library ../../outputs/library/Darwin-g++/libG4physicslists.dylib
[100%] Built target G4physicslists
geant4-09-05-ref-00-build Lavora! > make -j2
Scanning dependencies of target G4ABLA
Scanning dependencies of target G4EMLOW
[  0%] [  0%] Creating directories for 'G4ABLA'
Creating directories for 'G4EMLOW'
[  0%] [  0%] Performing download step (download, verify and extract) for 'G4EMLOW'
Performing download step (download, verify and extract) for 'G4ABLA'
-- downloading...
     src='http://geant4.cern.ch/support/source/G4EMLOW.6.23.tar.gz'
     dst='/Users/cirrone/cirrone/Geant4Dir/geant4-09-05-ref-00-build/Externals/G4EMLOW-6.23/src/G4EMLOW.6.23.tar.gz'
     timeout='none'
-- downloading...
     src='http://geant4.cern.ch/support/source/G4ABLA.3.0.tar.gz'
     dst='/Users/cirrone/cirrone/Geant4Dir/geant4-09-05-ref-00-build/Externals/G4ABLA-3.0/src/G4ABLA.3.0.tar.gz'
     timeout='none'
-- [download 0% complete]
-- [download 2% complete]
-- [download 10% complete]
-- [download 34% complete]
-- [download 81% complete]
-- [download 100% complete]
```

GUI version of cmake



A friendly way to do the same things (on Windows and Mac)

If everything is ok…

```
+- CMAKE_INSTALL_PREFIX
   +- bin/
   |  +- geant4-config    (UNIX ONLY)
   |  +- geant4.csh       (UNIX ONLY)
   |  +- geant4.sh        (UNIX ONLY)
   |  +- G4global.dll     (WINDOWS ONLY)
   |  +- ...
   +- include/
   |  +- Geant4/
   |     +- G4global.hh
   |     +- ...
   |     +- CLHEP/        (WITH INTERNAL CLHEP ONLY)
   |     +- tools/
   +- lib/                (MAY BE lib64 on LINUX)
   |  +- libG4global.so   (AND/OR .a, OR G4Global.lib ON WINDOWS)
   |  +- ...
   |  +- Geant4-9.5.0/
   |     +- Geant4Config.cmake
   |     +- Geant4ConfigVersion.cmake
   |     +- Geant4LibraryDepends.cmake
   |     +- Geant4LibraryDepends-Release.cmake
   |     +- UseGeant4.cmake
   |     +- Linux-g++     (OR Darwin-g++ UNIX ONLY SOFTLINK -> ..)
   +- share
      +- Geant4-9.5.0
         +- data/         (IF GEANT4_INSTALL_DATA WAS SET)
         +- geant4make/
            +- geant4make.csh
            +- geant4make.sh
            +- config/
```