

Hands-On Session: Accelerator Physics (Simulation Lab)

Gabriela Hoff - IIP/UFRN



Geant4 - GEometry ANd Tracking



Objectives of the Lecture

To introduce the **Geant4** toolkit, its applications and basic limitations, and to provide hands-on experience by running examples.

1. KNOW THE GEANT4 TOOLKIT



- Understand what Geant4 is and what it is not.
- Overview of its architecture and main components.
- Key concepts: geometry, physics processes, tracking, events, hits.
- Software structure and main classes.



Goal: gain a solid understanding of the Geant4 toolkit and how it works.

2. EXPLORE ITS APPLICATIONS



- High Energy Physics (detector simulation)
- Medical Physics (radiotherapy, imaging)
- Radiation Protection and Dosimetry
- Space and Astroparticle Physics
- Industrial and other applications



Goal: recognize the wide range of areas where Geant4 is used and its impact.

3. UNDERSTAND BASIC LIMITATIONS



- Physics models have validity ranges and approximations.
- Accuracy depends on models, data and physics list choices.
- Computational cost and memory requirements.
- Statistical (Monte Carlo) nature of the results.
- Geometry and input simplifications affect the outcome.



Goal: be aware of the main limitations and how they affect simulations and results.

4. RUN EXAMPLES



- Build and run official Geant4 examples.
- Explore detector construction, physics lists and primary generators.
- Visualize results.
- Modify parameters and observe the impact.



Goal: acquire practical experience by running and analyzing simulations.

TAKE-HOME MESSAGE

By the end of this lecture, you will be able to:



Describe the Geant4 toolkit and its main components.



Identify key application areas and real-world uses.



Recognize basic limitations and their implications.



Run and explore examples to start your own simulations.

This is the first step toward developing robust and reliable simulation applications with Geant4.

Geant4 General toolkit

AboutDownloadDocumentationUser Forum⁴²Bug Reports⁴²EventsContact Us

Geant4

Toolkit for the simulation of the passage of particles through matter. Its areas of application include high energy, nuclear and accelerator physics, as well as studies in medical and space science.

[Getting started](#)

Get started

Everything you need to get started with Geant4.

[I'm ready to start!](#)

Download

Geant4 source code and installers are available for download, with source code under an [open source license](#).

Latest: 11.4.2

Docs

Documentation for Geant4, along with tutorials and guides, are available online.

[Read documentation](#)

News

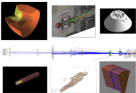
[26 Jun 2026](#)
[Release 11.5 beta](#)

[12 Jun 2026](#)
[Release 11.4.2](#)

[17 Mar 2026](#)
[2026 Planned Features](#)

[13 Mar 2026](#)
[Release 11.4.1](#)

[05 Dec 2025](#)
[Release 11.4](#)



About us

Collaboration

Contribute

☐ Highlight all ☐ Match Case ☐ Match Operators ☐ Ignore Words 1 of 10 matches



Geant4 General toolkit: Capabilities

GEANT4 – GENERAL CHARACTERISTICS

GENERAL CHARACTERISTICS



Comprehensive toolkit: provides all the components needed to build a wide range of simulations.



Geometry: flexible system for describing complex detector and world geometries.



Physics processes: large collection of models for electromagnetic, hadronic, optical and decay processes.



Tracking: precise simulation of particle transport and interactions.



User control: highly configurable and extensible by the user.



Performance: optimized for speed and memory usage (multi-threading supported).



Open source: freely available under the Geant4 Software License.

MAIN PHYSICS CAPABILITIES

PHYSICS DOMAIN	DESCRIPTION
 Electromagnetic	ionization, bremsstrahlung, multiple scattering, gamma interactions, optical photons...
 Hadronic	elastic, inelastic, capture, spallation, fission, evaporation...
 Radioactive Decay	α , β , γ and isomeric decays
 Optical Physics	Cherenkov, scintillation, absorption, reflection, refraction...
 Extensible	Users can add their own physics models and processes



GEANT4 PROVIDES:

- ✓ A unique combination of accuracy, flexibility and performance.
- ✓ A wide choice of physics models and high-precision particle transport.
- ✓ Support for a very broad range of energies (from eV to TeV and beyond).
- ✓ Tools for simulation, analysis and visualization integrated in a powerful framework.



Geant4 is a versatile and reliable toolkit used in high energy physics, nuclear physics, astroparticle physics, medical physics, space science and many other fields.

Geant4 General toolkit: Ecosystem

Roles in the Geant4 Ecosystem

From toolkit development to user applications

Geant4 is a **toolkit**. It provides the simulation engine, **physics models** and **tools**.
Different developers contribute and use Geant4 for their specific needs.

1. USER DEVELOP APPLICATION



- Defines the problem
- Builds the detector geometry
- Chooses physics list
- Generates primary particles
- Runs simulations
- Analyzes results

Focus: application to a specific experiment or study

2. SOURCE CODE DEVELOPER



- Develops or extends Geant4 software components
- Implements new features
- Optimizes performance
- Fixes bugs
- Contributes to the toolkit

Focus: software implementation and framework development

3. PHYSICS MODEL DEVELOPER



- Develops and improves physics models
- Implements new processes
- Validates models with data
- Extends physics capabilities
- Works across energy domains

Focus: physical accuracy and model validation

4. GEANT4 TOOLKIT DEVELOPER (CORE TEAM)



Maintains the overall architecture and quality of Geant4
Integrates contributions (code and physics models)
Ensures reliability, performance and documentation
Defines roadmap and releases



Quality Assurance



Integration & Build



Documentation & Examples



Performance & Scalability



Community & Support

Feedback loop
Users report issues and needs
Developers improve models and software



Open source (C++)
Cross-platform
Modular and extensible
Used in science, industry and medicine

GEANT4 TOOLKIT (CORE)

GEOMETRY



Describe the experimental setup: volumes, materials, positions, etc.

PHYSICS PROCESSES



Electromagnetic, hadronic, optical, decay, etc.
Wide range of models.

TRACKING



Transport of particles through matter, handling interactions and secondaries.

SERVICES & UTILITIES



Magnetic field, random numbers, units, data management, parallelization, etc.

PERSISTENCY & DATA MANAGEMENT

Input/Output, event data, hits, conditions, etc.



VISUALIZATION & ANALYSIS TOOLS

Visualization (OpenGL, HepRep), analysis (root, CSV, HDF5, ...)



APPLICATIONS & IMPACT



High Energy Physics
Collider experiments, detector design, performance studies



Nuclear Physics
Detector response, activation, shielding, dosimetry



Medical Physics
Radiotherapy, imaging, proton therapy, treatment planning



Space & Astroparticle Physics
Cosmic rays, space radiation, instrumentation



Radiation Protection & Industry
Shielding design, safety, material analysis, semiconductor, etc.

Users build applications using Geant4. Source code developers extend and improve the toolkit.

Physics model developers provide accurate models. Geant4 toolkit developers integrate and deliver a robust, reliable and powerful simulation toolkit for all.

Geant4 General toolkit: General Structure

GEANT4 – TYPICAL WORKFLOW AND APPLICATION AREAS

1. TYPICAL WORKFLOW

- 1  **Define the geometry** of the detector and materials.
- 2  **Choose or build a physics list** (select processes and models).
- 3  **Generate primary particles.**
- 4  **Geant4 transports particles,** simulates interactions and produces secondaries.
- 5  **Hits and energy deposits** are recorded.
- 6  **Results are stored** and visualized.



PHYSICS RESULTS AND ANALYSIS

Interpret, validate and extract physics quantities.

2. APPLICATION AREAS



High Energy Physics

Experiments at particle colliders (ATLAS, CMS, ALICE, LHCb, ...), detector design and performance studies.



Nuclear Physics

Detector response, nuclear reactions, shielding, dosimetry and activation studies.



Medical Physics

Radiotherapy (proton, ion, photon), imaging (CT, PET, SPECT), dose calculation and treatment planning.



Space Science

Simulation of cosmic rays, space radiation environment, and spacecraft instrumentation.



Radiation Protection

Shielding design, environmental studies, dose assessment and risk evaluation.



Other Applications

Material science, accelerator design, electronics reliability, cultural heritage, homeland security and more.



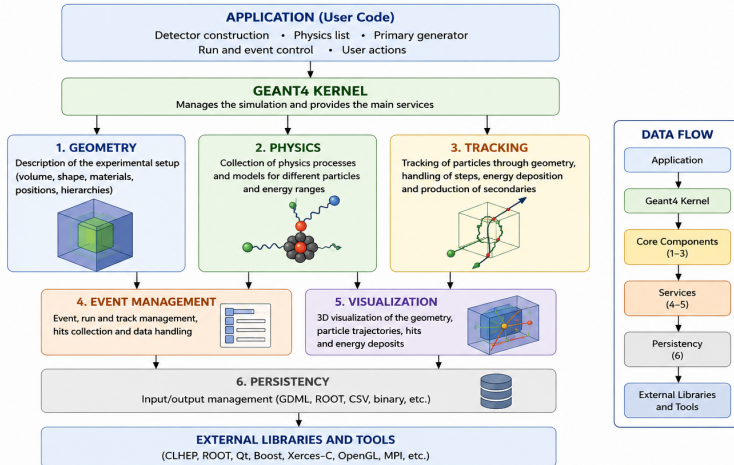
Geant4 is a powerful and versatile toolkit for the simulation of particle-matter interactions in a wide range of scientific and technological applications.

Geant4 General toolkit: General Structure

GEANT4 – GENERAL SOFTWARE STRUCTURE

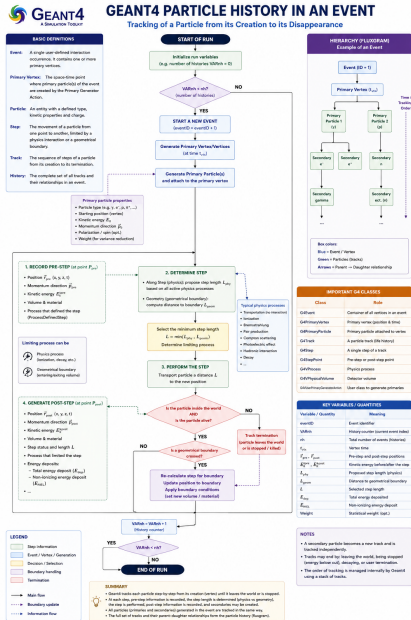
Geant4 is organized in a modular architecture.

The user application uses the Geant4 kernel and its components to perform a simulation.



The modular structure of Geant4 provides flexibility, extensibility and high performance for a wide range of particle-matter interaction simulations.

Defining the "source"



Geant4: The simulation beginning



GEANT4 INITIAL STEP QUANTITIES

Quantities that define the particle state and context
at the beginning of each step



1. Position (x, y, z)

Coordinates of the particle
at the beginning of the step.

$$\vec{r}_0 = (x_0, y_0, z_0)$$



2. Momentum Direction ($\vec{p}$)

Unit vector giving the
direction of motion.

$$\hat{u} = (u_x, u_y, u_z)$$



3. Kinetic Energy (E_k)

Particle kinetic energy
at the beginning of the step.

$$E_k$$



4. Global Time (t)

Time since the beginning
of the event.

$$t$$



5. Step Number

Step index in the track.

$$n_{\text{step}}$$



6. Volume and Material

Logical volume and material
of the current touchable.

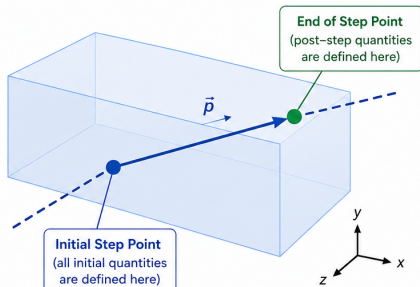
(Volume, Material)



7. Process Defined Step

The physical process that
limited the previous step.

*Process**



These initial quantities are used by Geant4 to:

- transport the particle,
- evaluate physics and geometry interactions,
- determine the proposed step length,
- and compute the outcome at the end of the step.

They are stored in the *G4StepPoint (PreStepPoint)*.

Geant4: processing the transportation

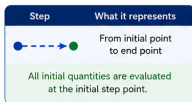
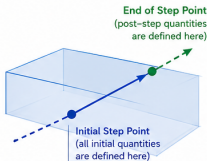


Geant4 Initial Step Quantities and How Geant4 Deals with Them

At the beginning of each step, Geant4 defines a set of initial step quantities that describe the particle state and the geometry/physics context **before** any interaction or boundary crossing is processed.

INITIAL STEP QUANTITIES

- **Position (x,y,z)**
Coordinates of the particle at the beginning of the step.
- **Momentum Direction ($\vec{p}$)**
Unit vector giving the direction of motion.
- **Kinetic Energy (E_k)**
Particle kinetic energy.
- **Global Time (t)**
Time since the beginning of the event.
- **Step Number**
Step index in the track.
- **Volume and Material**
Logical volume and material of the current touchable.
- **Process Defined Step**
The physical process that limited the previous step.



HOW GEANT4 DEALS WITH INITIAL STEP QUANTITIES



1. Initialization of the Step

Geant4 creates a G4Step object and copies the current Track information to define the initial step point.

All initial step quantities are stored in G4StepPoint (the PreStepPoint).



2. Transportation

Geant4 determines the proposed step length to the next interaction or geometry boundary.

The particle is transported along its direction without change.



3. Physics & Geometry Evaluation

At the end of the step, Geant4 selects the winning process or boundary and computes the outcome.

New particle state (energy, direction, position, time, etc.) is computed at the end point.



4. Post-Step Point

The final state is stored in the PostStepPoint.

Post-step quantities are available for user actions and scoring.



5. Ready for Next Step

The PostStepPoint becomes the initial point for the next step.

The cycle repeats until the track is terminated.



Key Points

- Initial step quantities describe the particle state before any change.
- They are taken from the Track at the initial step point (PreStepPoint).
- Geant4 transports the particle, applies physics/geometry, and produces post-step quantities.
- The post-step point becomes the initial point of the next step.



Why it matters?

Accurate definition of initial step quantities ensures correct physics interactions, energy deposition, and detector response simulation.



Geant4: processing the transportation



GEANT4 VERTEX

Definition and Quantities

A Geant4 Vertex represents a point in space-time where a particle track is created. It is the origin of a track and stores the information needed to define that point and its context.

VERTEX DEFINITION

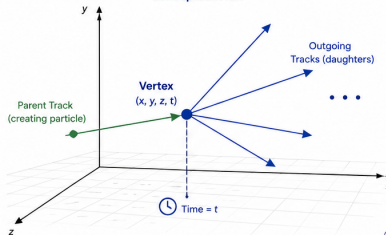
In Geant4, a vertex is an object of `G4VtxPoint` class.

It defines the position and time where a new track starts, together with the particle that created it (if any).

QUANTITIES ASSOCIATED WITH A VERTEX

- 1 Position**
Spatial coordinates of the vertex in the global reference frame. (x, y, z)
- 2 Time**
Global time of the vertex (time since the beginning of the event). t
- 3 Creating Particle**
Track ID of the parent particle that created this vertex. $parentID$
- 4 Event ID**
Identifier of the event to which this vertex belongs. $eventID$
- 5 Vertex ID**
Unique identifier of the vertex within the event. $vertexID$
- 6 Number of Outgoing Tracks**
Number of daughter tracks created at this vertex. N_{out}

Conceptual View



KEY POINTS

- Every primary particle starts from a primary vertex ($parentID = 0$).
- Secondary vertices are created when interactions (e.g., decay, hadronic interaction) produce new particles.
- The vertex stores the space-time point and the link to the particle that created the new tracks.
- All quantities are stored in the `G4VtxPoint` object and can be accessed during tracking and analysis.

Summary of Vertex Quantities

Quantity	Type (C++)	Description
Position (x, y, z)	<code>G4ThreeVector</code>	Spatial coordinates of the vertex (mm)
Time t	<code>G4double</code>	Global time of the vertex (ns)
Creating Particle ($parentID$)	<code>G4int</code>	Track ID of the parent particle
Event ID ($eventID$)	<code>G4int</code>	Identifier of the event
Vertex ID ($vertexID$)	<code>G4int</code>	Unique identifier of the vertex
Number of Outgoing Tracks (N_{out})	<code>G4int</code>	Number of daughter tracks

Typical Access in Geant4

```
const G4VtxPoint* vertex = track->GetVertex();  
// Position  
G4ThreeVector pos = vertex->GetPosition();  
// Time  
G4double time = vertex->GetGlobalTime();  
// Creating particle (parent track ID)  
G4int parentID = vertex->GetParentID();  
// Vertex ID  
G4int vertexID = vertex->GetVertexID();  
// Number of outgoing (daughter) tracks  
G4int nOut = vertex->GetNumberOfDaughters();
```

 The vertex provides the start point in space-time and the genealogy context for all tracks.

Geant4: processing the transportation



GEANT4 STEP: DEFINITION, STAGES AND QUANTITIES

A Step is the propagation of a track between two points: **Pre-Step Point** → **Post-Step Point**

STEP DEFINITION

In Geant4, a step represents the movement of a particle from one point to another along its trajectory.

It is the basic unit for transportation and interaction processes.

STEP STAGES

1. Initialize Step

Create G4Step and set the pre-step point.

2. Along Step (Transportation)

Propagate the particle along its current direction up to a proposed length.

3. At End of Step

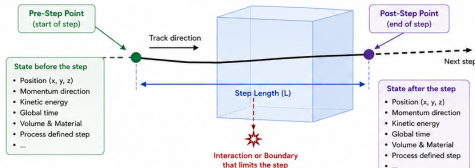
Determine the physical interaction or boundary that limits the step.

4. Define Post-Step

Set the post-step point and update position, time and other quantities.

5. Finalize Step

Store post-step quantities and process status.



STEP VARIABLES (G4Step)

G4Track*	fTrack	Pointer to the track
G4StepPoint*	fPreStepPoint	Pre-step point
G4StepPoint*	fPostStepPoint	Post-step point
G4double	fStepLength	Step length
G4double	fTotalEnergyDeposit	Total energy deposit
G4double	fNonionizingEnergyDeposit	Non-ionizing energy deposit
G4double	fDeltaEnergy	Energy change (= Pre.E - Post.E)
G4VProcess*	fPostStepProcess	Process limiting the step
G4StepStatus	fStepStatus	Status of the step
G4int	fPointOfNoReturn	Flag for geometry safety
G4double	fWeight	Statistical weight

STEP STATUS (G4StepStatus)

fGeomBoundary	: Step limited by geometry boundary
fAfterStepProc	: Particle is at rest
fAlongStepDoItProc	: Step limited by along-step process
fPostStepDoItProc	: Step limited by post-step process
fWorldBoundary	: Left the world volume
fUndefined	: Undefined / not set



A step is completely described by its pre-step and post-step points, the step length, the process that limited it, and the energy deposited.

PRE-STEP POINT QUANTITIES (G4StepPoint)

Quantity	Description
Position (x, y, z)	Coordinates at the beginning of the step
Momentum direction ($\hat{p}$)	Unit vector of particle direction
Kinetic energy (Ek)	Kinetic energy of the particle
Global time (t)	Time since the beginning of the event
Volume	Logical volume at the pre-step point
Material	Material at the pre-step point
Process defined step	Process that limited the previous step
Weight	Statistical weight

STEP QUANTITIES

Quantity	Description
Step length (L)	Distance traveled during the step
Total energy deposit	Total energy deposited in the step
Non-ionizing energy deposit	Energy lost to non-ionizing processes
Delta energy	Change in kinetic energy (Pre.E - Post.E)
Post-step process	Process that limited the step
Step status	Status of the step (see list on right)

$$\Delta E = E_k^{\text{pre}} - E_k^{\text{post}} \quad (\text{energy change during the step})$$

POST-STEP POINT QUANTITIES (G4StepPoint)

Quantity	Description
Position (x, y, z)	Coordinates at the end of the step
Momentum direction ($\hat{p}$)	Direction after the step
Kinetic energy (Ek)	Kinetic energy after the step
Global time (t)	Time after the step
Volume	Logical volume at the post-step point
Material	Material at the post-step point
Process defined step	Process that will limit the next step
Weight	Statistical weight

LEGEND



Pre-step point (start of step)



Post-step point (end of step)



Step length (L) (traveled distance)



Limiting interaction or boundary



Next step (continuation)

Geant4: processing the transportation



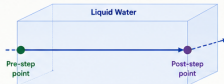
GEANT4 STEP COMPUTATION: EXAMPLES

EXAMPLE 1: TRANSPORTATION WITHOUT A BOUNDARY

(Particle moves inside the same volume)

Scenario

A 10 MeV electron moves in a homogeneous volume of liquid water. The proposed step length is limited by a physics process (e.g., ionization energy loss). No boundary is encountered.



Stages of the Step Computation

Stage	What Geant4 does	Main Quantities / Results
1 Initialize Step (Pre-step point)	Create G4Step and copy the current track information to define the pre-step point (G4StepPoint).	- Position $\vec{r}_0$ - Direction $\vec{\beta}_0$ - Kinetic energy $E_0 = 10 \text{ MeV}$ - Global time t_0 - Volume: Liquid Water
2 Along Step (Transportation)	Compute proposed step length by transportation manager and physics processes. The shortest proposed length is selected.	- Proposed step length L_{proposed} (limited by ionization process) - No boundary found within L_{proposed}
3 At End of Step (Physics & Geometry Evaluation)	Evaluate discrete processes and geometry at the end of the proposed step. No boundary is crossed.	- Process: Ionization - Energy loss ΔE - Final energy $E_1 = E_0 - \Delta E$ - Step status: fGeomBoundary = false
4 Define Post-step Point	Store the post-step point with updated quantities.	- Position $\vec{r}_1 = \vec{r}_0 + L \vec{\beta}_0$ - Direction $\vec{\beta}_1$ (unchanged) - Energy E_1 - Global time $t_1 = t_0 + \Delta t$ - Volume: Liquid Water
5 Finalize Step	Store step quantities, update track, and prepare for the next step.	- Step length L - Total energy deposit ΔE - Next step starts from post-step point

Summary

The particle was transported inside the same volume. The step length was limited by a physics process (ionization). No boundary was encountered.

Pre-step point → Transportation → End of step (physics) → Post-step point

EXAMPLE 2: TRANSPORTATION WITH A BOUNDARY IDENTIFICATION

(Particle reaches a volume boundary)

Scenario

A 10 MeV gamma photon moves in liquid water and reaches the boundary with a lead (Pb) absorber. The step is limited by the geometry boundary.



Stages of the Step Computation

Stage	What Geant4 does	Main Quantities / Results
1 Initialize Step (Pre-step point)	Create G4Step and copy the current track information to define the pre-step point.	- Position $\vec{r}_0$ - Direction $\vec{\beta}_0$ - Kinetic energy $E_0 = 10 \text{ MeV}$ - Global time t_0 - Volume: Liquid Water
2 Along Step (Transportation)	Compute proposed step length by transportation manager and physics processes. Also compute distance to nearest geometry boundary.	- Proposed step length by processes L_{proc} - Distance to boundary L_{geom} - Selected step length $L = \min(L_{\text{proc}}, L_{\text{geom}})$ → here: $L = L_{\text{geom}}$ (boundary wins)
3 At End of Step (Physics & Geometry Evaluation)	At the end of the step, a geometry boundary is found. Do not cross the boundary.	- Step status: fGeomBoundary = true - Process at end: Transportation - Energy deposit in water: ΔE_{water} - New volume: Lead (Pb) identified
4 Define Post-step Point	The post-step point is set on the boundary surface (just before entering the next volume).	- Position $\vec{r}_1$ on boundary surface - Direction $\vec{\beta}_1$ (unchanged) - Energy $E_1 = E_0 - \Delta E_{\text{water}}$ - Global time $t_1 = t_0 + \Delta t$ - Volume: Lead (Pb) (next volume)
5 Finalize Step	Store step quantities, update track and prepare for the next step in the new volume.	- Step length $L = L_{\text{geom}}$ - Boundary identified - Next step starts in Lead (Pb)

Summary

The particle reached a geometry boundary. The step length was limited by the distance to the boundary. The post-step point is located on the boundary and the next step will start in the new volume.

Pre-step point → Transportation → Boundary found → Post-step point (on boundary)

Legend

- Pre-step point (start of step)
- Post-step point (end of step)
- Track direction
- ★ Boundary encountered
- Current volume
- Next volume

Quantities in G4Step

- StepLength
- TotalEnergyDeposit
- PreStepPoint and PostStepPoint
- StepStatus (e.g., fGeomBoundary)

Key idea

At each step, Geant4 transports the particle up to the closest limit: a physics interaction or a geometry boundary, and updates the particle state accordingly.

Geant4: borders recognizance and treatment

Geant4 – HOW TO TREAT TRANSPORT NEAR TO GEOMETRICAL BORDERS

Geant4 uses geometrical tolerances and navigator options to decide if a particle is inside, outside or on a surface and how to move it safely across boundaries.

GENERAL PRINCIPLES

- Geometry boundaries are not infinitely sharp.
- Numerical tolerances are used to classify positions and to avoid stuck or repeated crossings.
- Navigator “pushes” the particle when it is found too close to a surface.
- Zero-step protection prevents endless loops at boundaries.
- Relocation and re-navigation ensure a consistent volume after every step.

GEOMETRICAL TOLERANCE (SURFACE ACCURACY)

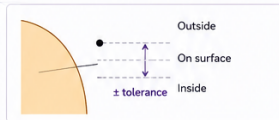
Defines the numerical tolerance used to decide if the particle is considered on/inside/outside a surface.

Parameter (UI)	Default
/geometry/tolerance	1e-9 mm

1 GEOMETRY TOLERANCE (SURFACE ACCURACY)

Defines the basic tolerance used by the geometry package to decide if a particle is on, inside or outside a surface.

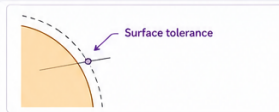
Parameter (UI)	Default
/geometry/tolerance	1e-9 mm



2 SURFACE TOLERANCE

Additional tolerance used by solid models to classify points near surfaces.

Parameter (UI)	Default
/geometry/surfaceTolerance	1e-9 mm

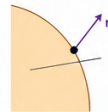


Geant4: borders recognizance and treatment

3 TREATMENT NEAR BOUNDARY (PUSH / RELOCATE)

Navigator slightly moves the point along the surface normal to avoid being stuck at borders or re-crossing the same surface.

Parameter (UI)	Default	Description
/geometry/navigator/accurate	true	Use accurate calculations (slower but more robust).
/geometry/navigator/push	true	Push the point if it is too close to a surface.
/geometry/navigator/pushValue	1e-7 mm	Distance used to push the point along the normal.
/geometry/navigator/relocate	true	Relocate the point if it is numerically outside.

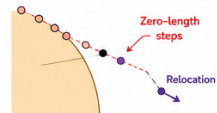


- Original point (too close)
- After push (inside volume)

4 ZERO-STEP PROTECTION

Parameters that help to recover from repeated zero-length steps.

Parameter (UI)	Default	Description
/geometry/navigator/zeroStepLimit	10	Maximum number of consecutive zero-steps before forcing relocation.
/process/transportation/(optional)	0	Additional limits to detect stuck situations.

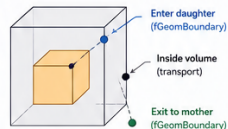


Geant4: borders recognizance and treatment

5 BOUNDARY IDENTIFICATION (POST-STEP)

How to recognize that a step has stopped at a geometry boundary.

Check	Meaning
<code>postPoint->GetStepStatus() == fGeomBoundary</code>	Step stopped exactly at a geometry boundary.
<code>postPoint->GetPhysicalVolume() != prePoint->GetPhysicalVolume()</code>	Particle entered another physical volume (crossed a border).
<code>postPoint->GetPhysicalVolume() == nullptr</code>	Particle exited the world volume (StepStatus = fWorldBoundary).



6 RECOMMENDED PRACTICE

- Use default tolerances unless you have specific precision problems.
- If you observe stuck particles at borders:
 1. Check overlaps in geometry.
 2. Keep `/geometry/navigator/push = true`.
 3. If needed, increase `/geometry/navigator/pushValue` slightly.
 4. Avoid making tolerances much larger than the defaults.

TYPICAL DEFAULTS (ROBUST AND SAFE)

<code>/geometry/tolerance</code>	1e-9 mm
<code>/geometry/surfaceTolerance</code>	1e-9 mm
<code>/geometry/navigator/push</code>	true
<code>/geometry/navigator/pushValue</code>	1e-7 mm
<code>/geometry/navigator/zeroStepLimit</code>	10

✓ SUMMARY – HOW GEANT4 TREATS BORDERS



1. Classify position
Using tolerance and surface tolerance, the point is classified as inside, outside or on surface.



2. Push / Relocate
If too close or outside, the navigator pushes or relocates the point along the surface normal.



3. Re-navigate
Navigator recomputes the current volume consistently.



4. Transport
The step is performed safely without getting stuck or re-crossing the same boundary.



IMPORTANT

Proper treatment of transport near borders avoids:

- Stuck tracks
- Infinite zero-steps
- Wrong volume classification
- Unphysical geometry leaks

What do I need to run an application

After installing Geant4, add tools (if necessary)

- ▶ create the Geometry
- ▶ invoke the physics processes and models
- ▶ define the "source"

Creating the geometry



Geant4 – MAIN STRUCTURE OF A DETECTOR CONSTRUCTOR CLASS

In Geant4, the detector geometry is typically implemented in a class that inherits from `G4VUserDetectorConstruction`. This class is responsible for building and returning the world volume (top of the geometry).

1. CLASS DEFINITION

```
class DetectorConstruction : public G4VUserDetectorConstruction
{
public:
    DetectorConstruction();
    virtual ~DetectorConstruction();

    virtual G4VPhysicalVolume* Construct(); // Build and return world

private:
    // --- Geometry building helper methods ---
    void DefineMaterials(); // Define all materials
    G4VPhysicalVolume* DefineVolumes(); // Build the geometry
    void DefineSensitiveDetectors(); // (optional) SD

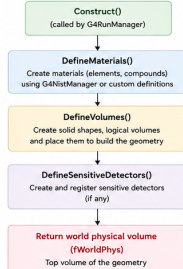
    // --- Member data ---
    G4NistManager* fNist; // Material manager
    G4LogicalVolume* fWorldLog; // World logical volume
    G4VPhysicalVolume* fWorldPhys; // World physical volume
};
```



PURPOSE

- Define all materials used in the detector.
- Build the geometry (volumes and placements).
- Assign visual attributes (optional).
- Register sensitive detectors (optional).
- Return the world physical volume to Geant4.

2. GEOMETRY CONSTRUCTION FLOW (Construct())



3. MAIN MEMBER DATA

Member	Purpose
<code>G4NistManager* fNist</code>	Access to the NIST material database.
<code>G4LogicalVolume* fWorldLog</code>	Logical volume of the world.
<code>G4VPhysicalVolume* fWorldPhys</code>	Physical volume of the world (returned by <code>Construct()</code>).
Other logical/physical volume pointers	(optional) Keep pointers if needed outside <code>Construct()</code> .
Parameters (<code>G4double</code> , <code>G4int</code> , etc.)	Geometry dimensions, thicknesses, positions, etc.

4. KEY NOTES



`Construct()` must build the complete geometry and return the world physical volume.



All volumes are built using: `G4VSolid` → `G4LogicalVolume` → `G4VPhysicalVolume` (placed in mother volume).



Visual attributes (colors, visibility) are optional and help in visualization.



Sensitive detectors are associated to logical volumes. They are not part of the geometry itself.



This class is registered in the main program as:

```
runManager->SetUserInitialization(new DetectorConstruction());
```

5. EXAMPLE OF THE CONSTRUCT() METHOD STRUCTURE

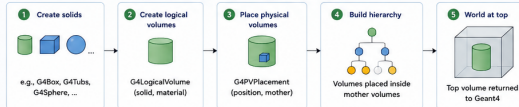
```
G4VPhysicalVolume* DetectorConstruction::Construct()
{
    // 1. Define materials
    DefineMaterials();

    // 2. Build geometry
    fWorldPhys = DefineVolumes();

    // 3. Sensitive detectors (optional)
    DefineSensitiveDetectors();

    // 4. Return world
    return fWorldPhys;
}
```

Inside DefineVolumes() – Typical steps



LEGEND

- Initialization / Entry
- Material definition
- Geometry building
- SD / Auxiliary
- Return / Output

Invoking physics list



GEANT4 PHYSICS LIST: MAIN STRUCTURE AND PHYSICS CLASSES

A Physics List defines all physics processes used in the simulation: particles, models, cross sections and production cuts.

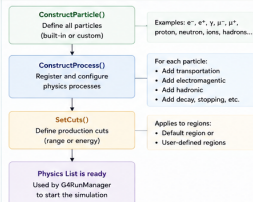
1. PHYSICS LIST CLASS – MAIN STRUCTURE

```
class MyPhysicsList : public G4VUserPhysicsList
{
public:
    MyPhysicsList();
    virtual ~MyPhysicsList();
    virtual void ConstructParticle(); // Define all particles
    virtual void ConstructProcess(); // Define all physics
    virtual void SetCuts(); // Set production cuts
};
```

MAIN RESPONSIBILITIES

- ✓ Define all particles used in the simulation.
- ✓ Register and configure the physics processes.
- ✓ Set production cuts for each region.

2. PHYSICS LIST CONSTRUCTION FLOW



TYPICAL MEMBERS (inherited)

- defaultCutValue
- particleIterator
- physicsListHelper
- processManager
- emConfigurator (opt.)
- hadronicConfigurator (opt.)
- decayPhysics (opt.)
- stoppingVerbose (opt.)
- ...

KEY NOTES

- ⚙️ Transportation is added automatically by Geant4.
- 📌 All other processes must be added in `ConstructProcess()`.
- 🔗 Production cuts define when secondaries are produced.

3. PHYSICS CLASSES IN GEANT4

Category	Main Base Class	Purpose	Examples of Concrete Classes
Electromagnetic (EM) 	G4VEmPhysics	EM interactions of leptons and photons (ionization, bremsstrahlung, Compton, pair production, scattering, ...)	• G4EmStandardPhysics • G4EmStandardPhysics_option1/2/3/4 • G4EmLivermorePhysics • G4EmPenelopePhysics • G4EmLowEPhysics • G4EmDNAPhysics_option2, G4EmDNAPhysics
Hadronic 	G4VHadronicPhysics	Hadronic interactions (elastic, inelastic, capture, fission, fragmentation, ...)	• G4HadronElasticPhysics • G4HadronPhysicsQGSP_BERT • G4HadronPhysicsQGSP_BIC • G4HadronPhysicsFTFP_BERT • G4HadronPhysicsLHEP • G4HadronPhysicsShielding • G4BinaryCascadePhysics • G4IonBinaryCascadePhysics
Decay 	G4VDecayPhysics	Radioactive and particle decays	• G4DecayPhysics • G4RadioactiveDecayPhysics
Stopping 	G4VStoppingPhysics	Stopping of charged particles at rest	• G4StoppingPhysics • G4LowEnergyStoppingPhysics
Neutron 	G4VNeutronPhysics	Neutron specific processes	• G4NeutronTrackingCut • G4NeutronTimeCut • G4NeutronHPPhysics
Ion 	G4VionPhysics	Ion ionization and multiple scattering	• G4IonPhysics • G4IonRangePhysics • G4IonCRUJ3Physics
Optical 	G4VOpticalPhysics	Optical photon processes (scintillation, Cherenkov, absorption, Rayleigh, ...)	• G4OpticalPhysics • G4ScintillationPhysics • G4CerenkovPhysics
Low-energy / Specialized 	— (various)	Low energy, DNA, medical, space, biasing, etc.	• G4EmDNAPhysics • G4EmLivermorePhysics • G4SpacePhysics • G4BiasingPhysics

4. EXAMPLES OF COMPLETE PHYSICS LISTS (PREDEFINED)

FTFP_BERT	QGSP_BIC	QGSP_BERT	Shielding	LHEP	Em Standard Option 4	Em Livermore (or Penelope)
Balanced for general purpose (High Energy)	Better for shielding and low-energy hadrons	High precision hadronic interactions	Optimized for shielding applications	Low energy (< 20 GeV) hadrons	Precision EM (medical, low E)	Low energy EM (< 1 GeV)

All these physics lists combine several physics classes internally.

5. HOW A PHYSICS LIST IS USED



KEY TAKEAWAYS

- ✓ A Physics List = set of physics processes. It defines particles, processes and cuts.
- ✓ Choose a list appropriate to your energy range and application.
- ✓ You can create your own Physics List by combining physics classes.

Defining the "source"

BASIC DEFINITION

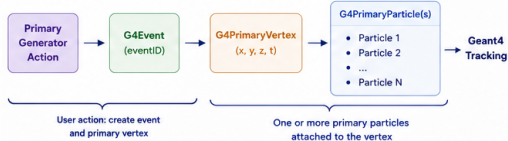
The Primary Generator Action (`G4VUserPrimaryGeneratorAction`) is a user-defined class responsible for generating the primary event.

It defines the primary vertex (position and time) and one or more primary particles (species, energy, direction, polarization, etc.) that start the simulation.

KEY OBJECTS

- `G4Event`: container of the primary vertices of the event
- `G4PrimaryVertex`: position, time and list of primary particles
- `G4PrimaryParticle`: particle definition and kinematic properties

CONCEPTUAL VIEW



STAGES OF THE PRIMARY GENERATOR ACTION

1



Initialization

- User initializes particle gun or other generator.
- Define particle properties and distributions.

Called once at beginning of run

2



GeneratePrimaries(`G4Event* anEvent`)

- Called once per event.
- Create a primary vertex and set its position and time.
- Generate one or more primary particles.
- Add the vertex (with its particles) to the event.

Called for every event

3



Add Vertex to Event

- `anEvent->AddPrimaryVertex(vertex);`

One vertex (or more)

4



Tracking Starts

- Geant4 takes the event and starts tracking from the primary particles.

After generation

Defining the "source"

TYPICAL WORKFLOW (PER EVENT)

1

Create a primary vertex

```
G4PrimaryVertex* vertex = new G4PrimaryVertex(vertexPos, vertexTime);
```

- vertexPos = (x, y, z), vertexTime = t



2

Define primary particle(s)

```
G4PrimaryParticle* particle = new G4PrimaryParticle(particleDef);
```

- particleDef = G4ParticleTable::GetParticleTable()->FindParticle("e-");
- Set momentum direction or momentum:
 particle->SetMomentumDirection(direction); // unit vector
 or particle->SetMomentum(momentum); // MeV/c
- Set kinetic energy:
 particle->SetKineticEnergy(energy); // MeV
- Set time (if needed): particle->SetGlobalTime(t);
- (Optional) Set polarization, weight, etc.



3

Attach particle to the vertex

```
vertex->SetPrimary(particle);
```

- Repeat for additional primary particles if needed.

4

Add vertex to the event

```
anEvent->AddPrimaryVertex(vertex);
```

Defining the "source"

EXAMPLE: USING G4PARTICLEGUN

```
// In constructor
fParticleGun = new G4ParticleGun(1); // one particle per vertex
fParticleGun->SetParticleDefinition(
    G4ParticleTable::GetParticleTable()->FindParticle("e-"));
fParticleGun->SetParticleEnergy(10.*MeV);
fParticleGun->SetParticleMomentumDirection(G4ThreeVector(0.,0.,1.));
fParticleGun->SetParticlePosition(G4ThreeVector(0.,0.,-100.*mm));

// In GeneratePrimaries(G4Event* anEvent)
fParticleGun->GeneratePrimaryVertex(anEvent);
```



G4ParticleGun is a convenient helper class to generate one or more primary particles with simple syntax.

IMPORTANT CLASSES AND VARIABLES

Class / Type	Variable / Method	Type	Description
G4Event*	anEvent	pointer	Current event
G4PrimaryVertex*	vertex	pointer	Primary vertex of the event
G4PrimaryParticle*	particle	pointer	Primary particle
G4ThreeVector	vertexPos	vector	Vertex position (x, y, z)
G4double	vertexTime	double	Vertex global time [ns]
G4ParticleDefinition*	particleDef	pointer	Particle type definition
G4double	kinEnergy	double	Kinetic energy [MeV]
G4ThreeVector	momentumDir	vector	Momentum direction (unit vector)
G4double	globalTime	double	Particle time [ns]
G4double	weight	double	Statistical weight (optional)

Random Number usage



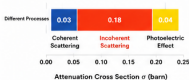
USING THE RANDOM NUMBER GENERATOR IN GEANT4

Process Selection Based on Total Cross Section

At each step, Geant4 selects which physical process will occur by sampling a random number based on the relative probabilities of all competing processes.

1. ATTENUATION CROSS SECTIONS (σ)

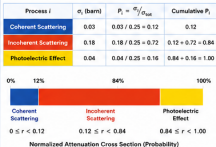
For the current particle, material and energy, each process i provides its cross section σ_i .



Total attenuation cross section
 $\sigma_{\text{tot}} = \sigma_{\text{coh}} + \sigma_{\text{incoh}} + \sigma_{\text{pe}} = 0.25 \text{ barn}$

2. NORMALIZATION (PROBABILITIES)

Cross sections are normalized by the total to obtain selection probabilities.



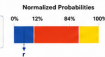
3. RANDOM NUMBER SAMPLING

Generate a random number r uniformly distributed in $[0, 1)$.

Example:
 $r = 0.0053721$

Locate r in the cumulative probability interval:
 $0 \leq r = 0.0053721 < 0.12$

Selected process:
 Coherent Scattering

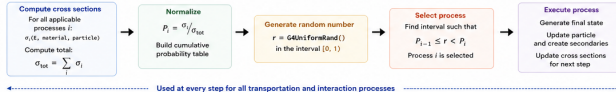


WHEN GEANT4 USES RANDOM NUMBERS IN AN EVENT (Typical moments)

- Primary vertex generation
Position, direction, time of primaries
- Randomization in primary generation
Energy, direction, polarization, etc.
- Process selection (this example)
Choose which physical process occurs
- Final state generation
Scattering angles, energies, secondary multiplicity, etc.
- Step limitation
Multiple scattering, range straggling, time of flight, etc.
- Decay / Nuclear / Hadronic processes
Decay times, channels, kinematics
- Optical photon processes
Absorption, re-emission, reflections, wavelength, etc.
- Biasing and variance reduction
Russian roulette, splitting, weights
- Output randomization (optional)
Smearing, digitization, etc.

✓ All random numbers in Geant4 come from the same engine (CLHEP Random) and are independent across threads.

HOW GEANT4 SELECTS A PROCESS (AT EACH STEP)



KEY FORMULAS

Total cross section:

$$\sigma_{\text{tot}} = \sum_i \sigma_i$$

Normalization (probability):

$$P_i = \sigma_i / \sigma_{\text{tot}}$$

Cumulative probability:

$$C_i = \sum_{j=1}^i P_j$$

(By definition, $C_n = 1$)

NOTES

- The random number r is uniformly distributed in $[0, 1)$.
- The probabilities are proportional to the cross sections.
- The selected process i is the one for which $C_{i-1} \leq r < C_i$.
- The same procedure is repeated independently at every step of every track.
- Cross sections depend on particle type, energy and material.

PSEUDOCODE (CONCEPTUAL)

```

// Assume: vector<Process> processes with cross sections sigma[i]
double sigmaTot = 0.0;
for(i in processes) sigmaTot += sigma[i];

double r = G4UniformRand(); // 0 <= r < 1
double cumulative = 0.0;
for(i in processes) {
    cumulative += sigma[i] / sigmaTot;
    if(r < cumulative) { select process i; break; }
}
  
```

LEGEND (COLORS)

Coherent Scattering (elastic)

Incoherent Scattering (inelastic)

Photoelectric Effect (absorption)

Flow / procedure

Random number usage

Random number position

SUMMARY

Geant4 uses the Random Number Generator to sample:

- which process occurs (based on cross sections),
- how the process develops (final state),
- and many other stochastic aspects of particle transport.

This ensures physically correct and unbiased Monte Carlo simulation.



Geant4 and Large Experiments

Experiment	Laboratory	Geometry Framework
ATLAS	CERN	GeoModel + Geant4
CMS	CERN	DD4hep / DDD + Geant4
ALICE	CERN	O2 + Geant4
LHCb	CERN	DD4hep + Gauss
FCC Detector Concepts	CERN	DD4hep + Geant4
CLIC Detector Concepts	CERN	DD4hep
ILD	ILC	DD4hep
SiD	ILC	DD4hep
Belle II	KEK	basf2 + Geant4
BESIII	IHEP	BOSS + Geant4
DUNE	Fermilab	LArSoft

Tabela 1: Large experiments and detector geometries implemented using Geant4.

Geant4 and Large Experiments

Experiment	Laboratory	Geometry Framework
MicroBooNE	Fermilab	LArSoft
SBND	Fermilab	LArSoft
ICARUS	Fermilab	LArSoft
JUNO	JUNO Colla- boration	SNiPER + Geant4
Hyper-Kamiokande	Kamioka	Geant4
Super-Kamiokande	Kamioka	Geant4
SNO+	SNOLAB	RAT-PAC
XENONnT	LNGS	Geant4
LZ	SURF	BACCARAT
DarkSide-20k	LNGS	G4DS
AMS-02	ISS	Geant4

Tabela 2: Large experiments and detector geometries implemented using Geant4.

Geant4 and Large Experiments

Experiment	Laboratory	Geometry Framework
DAMPE	China	Geant4
HERD	China Space Station	Geant4
Fermi-LAT	NASA	Geant4
BM	FAIR	FairRoot + Geant4
PANDA	FAIR	FairRoot + Geant4
R3B	FAIR	FairRoot + Geant4
Pierre Auger	Argentina	Offline + Geant4
CTA	International	sim_telarray + Geant4
Sirius Beamlines	CNPEM	Geant4

Tabela 3: Large experiments and detector geometries implemented using Geant4.

Hand's On

Open your VM, and let's have fun!