INTERNATIONAL WORKSHOP
ON THE HIGH ENERGY CIRCULAR ELECTRON POSITRON COLLIDER
CEPC 2025
November 6-10, 2025 Guangzhou, China

Gaussino Simulation Framework

Introduction

Building Blocks and New Features

Soon to come and Further Plans

with some examples from its use in LHCb

Gloria Corti, CERN

Gsino Developers – M. Clemencic, M. Mazurek, A. Morris, D. Muller, W. Pokorski, G. Stewart

With contributions from students – F. Bianchi, B. Brown, S. Dobbs, A. Gomez, R. Pozzi, J. Rubio, W. Shi
and LHCb Simulation developers – B. Fang, N. McHugh, et al.

The idea of Gaussino

- SFT/FCC exploration of existing software solutions in 2015-2016
  - Gauss (LHCb simulation framework) identified as a potential base for a production quality implementation
    - Generation rather straightforward to use
    - Despite the work required (parallelism, fast simulations, specific FCC pileup…) interested in picking up simulation part, too
  - Experiments need to follow development of Gaudi and Geant4

“We should join forces for an experiment independent Gauss-core”

Investigation, in B. Hegner talk at 6th LHCb Computing Workshop, Nov 2015, LPHNE Paris
First ideas, in B. Hegner talk at 8th LHCb Computing Workshop, Nov 2016, LPHNE Paris

The idea of Gaussino in LHCb

- LHCb Upgrade in Run3 very challenging for software and computing!
  - large increase in luminosity, i.e. L_{inst} : $4 \times 10^{32} \rightarrow 2 \times 10^{33} cm^{-2} s^{-1}$ and pileup: $1.1 \rightarrow 7.6$
  - full software trigger with high signal purity
  - analysis directly on trigger output
  - **simulation will continue to dominate the CPU needs**

- Modernization of the whole software
  - **Multi-threading**
  - Better use of multi-processor CPUs
  - **Reduce memory usage**
  - Optimize cache performance
  - **Remove dead code**
  - Move to modern data structures
  - Enable code vectorization
  - Enable algorithmic optimization
  - HLT1 reconstruction on **GPUs**

LHCb Collaboration, Upgrade Software and Computing TDR, CERN-LHCC-2018-007

The idea of Gaussino in LHCb

- LHCb Upgrade in Run3 very challenging for software and computing!
  - large increase in luminosity, i.e. Linst : $4 \times 10^{32} \rightarrow 2 \times 10^{33} cm^{-2} s^{-1}$ and pileup: $1.1 \rightarrow 7.6$
  - full software trigger with high signal purity
  - analysis directly on trigger output
  - **simulation will continue to dominate the CPU needs**

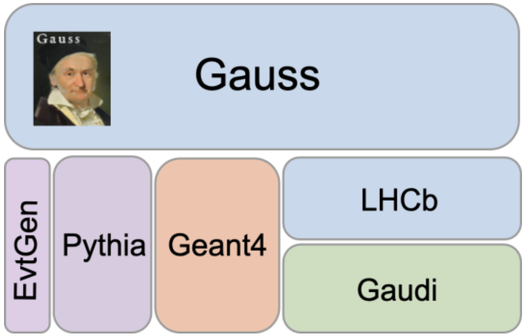
- Modernization of the whole software
  - **Multi-threading**
  - Better use of multi-processor CPUs
  - **Reduce memory usage**
  - Optimize cache performance
  - **Remove dead code**
  - Move to modern data structures
  - Enable code vectorization
  - Enable algorithmic optimization
  - HLT1 reconstruction on GPUs

Simulation software upgrade also needed!
- $\sim$ **15 years old**
- Adapt **to change** in LHCb common software, e.g. use of DD4Hep
- Exploit new feature of external HEP simulation software, e.g. in Geant4
- Combine multi-threaded Gaudi and Geant4
- Need of extensive palette of fast simulations

Separate simulation core functionality

LHCb Collaboration, Upgrade Software and Computing TDR, CERN-LHCC-2018-007

From Gauss to Gaussino …



Restructure the LHCb code introducing an **experiment-independent layer**

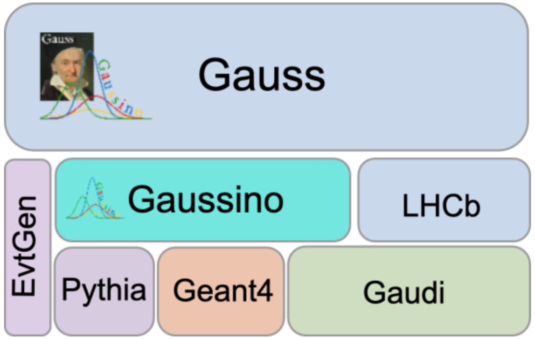
Gaussino Core Simulation framework

- uses Gaudi as core software framework
- run minimal functionality in stand-alone mode
- ideal test-bed for new developments
- make it available in Key4Hep Turnkey software stack

… back to Gauss[-on-Gaussino]

Gaussino Core Simulation framework

- provides the structure and the hooks
- provides components to use HEP-wide software, e.g. for Pythia8 and Geant4
- an experiment layer can be built on top



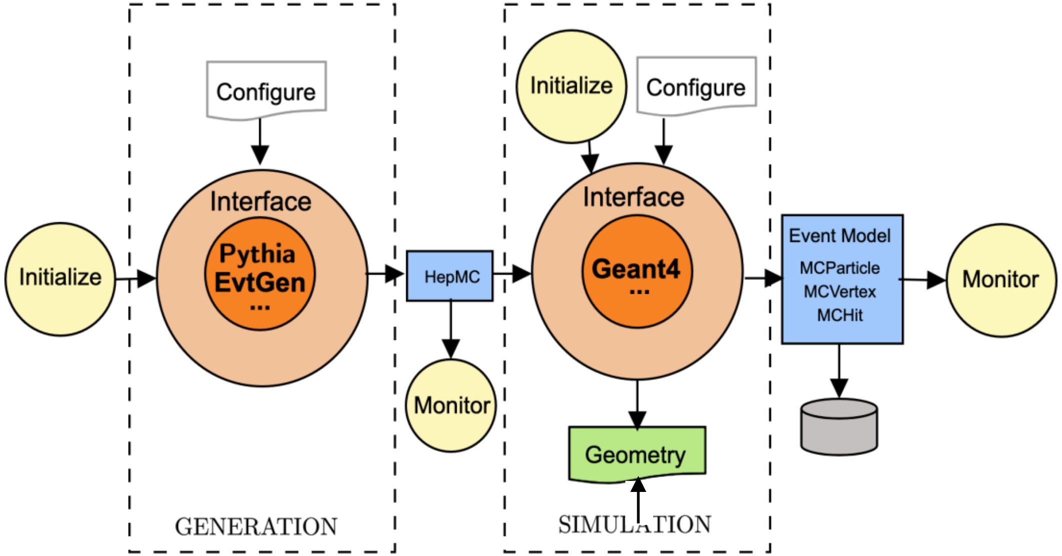
Gauss[-on-Gaussino] is the new version of the LHCb simulation framework

- based on Gaussino's core elements
- adds LHCb-specific components and configurations

Gaussino architecture

A complete simulation framework following the basic Gauss architecture

- Modularity
- Integrated generation and simulation phases
- Gaudi based
- MC truth output



Generation phase kept ***mostly* as-is**
Pythia8 adaptor available

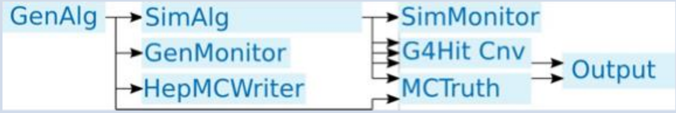
Simulation phase **redesigned**
following review of key elements

Underlying core elements

Execution structure

Use Gaudi functional framework

- Every algorithm as a 'task'
- Constant execution
  - Random engines created per call
- Fixed input/output



Random numbers

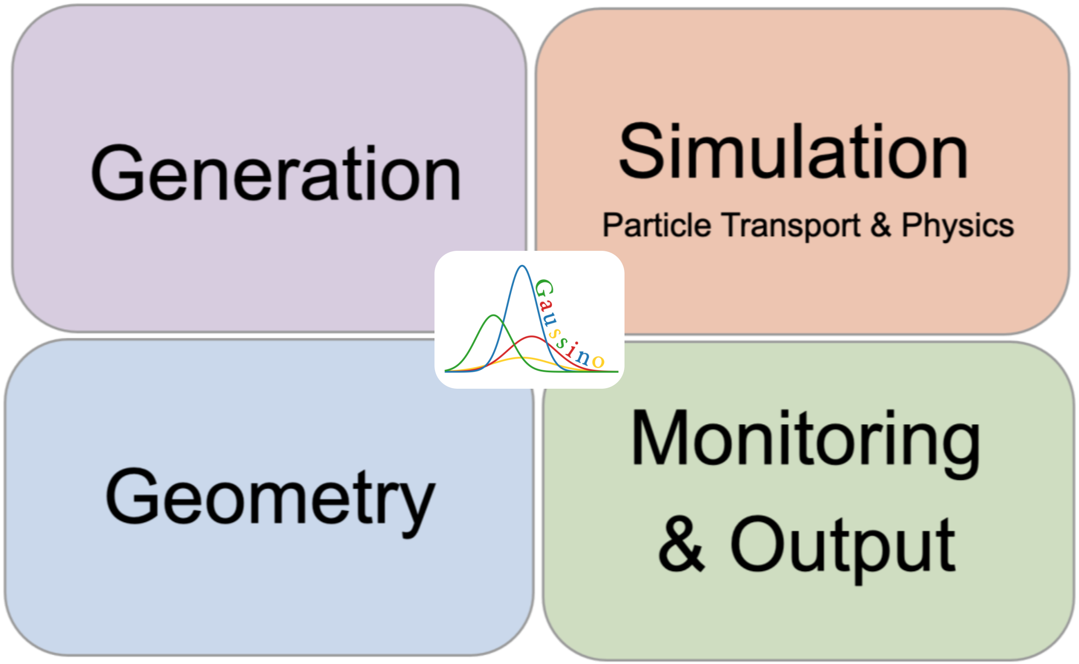
Must ensure reproducibility

- Adapt to multi-threading framework
  - A global singleton random engine want work!
- Create random engine on the stack
  - Pass it as reference
- Seed initialized with:
  - run#, event #, algorithm instance name, i.e. largest predictable unit

and passed to external libraries

Configurable building blocks

High level configuration in python



Python configurables steering C++ classes

A main configurable for each module

Can run simple setups in **stand-alone** mode with the **Gaussino configurable(s)**

Can **build** experiment specific configurable **on** the Gaussino configurable(s)

Upcoming change in Gaussino Configuration

- Migration to GaudiConfig2 in steps, starting with Generator phase
  - Gaudi **objects** (algorithms, tools, services) are configured by setting values of **properties** through **python bindings.**
  - Users interact with **high-level configurables** which translate options to low-level properties.

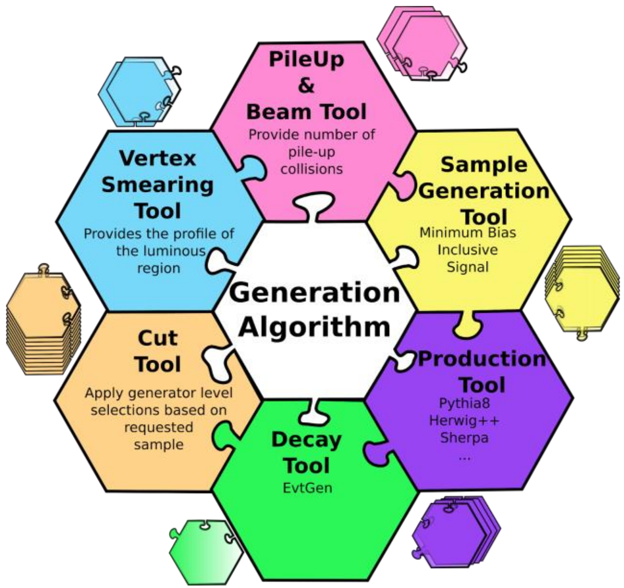
Old Configurables
- Global singletons (name-based)
- Implicit and silent overrides
- Patchy type-checking
- Unset properties do not exist

GaudiConfig2
- Local objects + functions that return configurables
- Overrides must be explicit by wrapping functions
- Precise and easily-extensible type-checking
- Unset properties return default values

- Implementation in Gaussino
  - Keep the same modular structure
  - Store user options in Pydantic object
  - Collect configurables in custom mapping (ConfigDict)

High-level options with Pydantic
- Pydantic provides an easy way to do type-checking and set default options
- Custom types
- Validator functions
- Dynamic default values *e.g.* based on other options
- Defines a YAML schema

Generation Phase



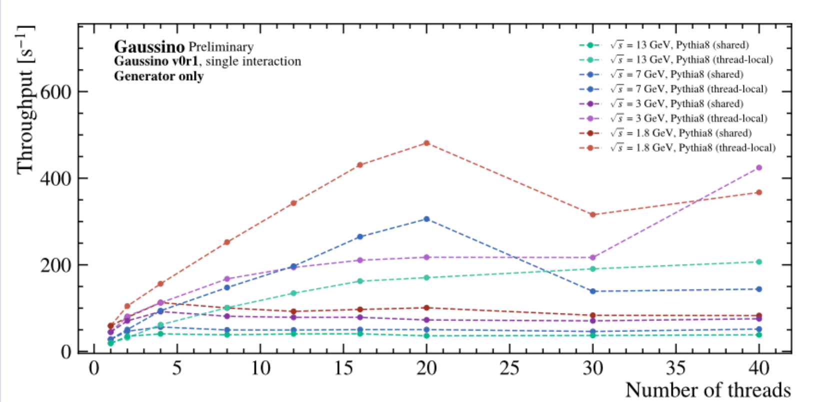
Extracted as-is from Gauss
Highly modular
Thread safety of generators
HepMC3 as exchange format

LHCb-PROC-2010-056, NSS2010

Shared and thread local Pythia 8
comparible with 8.2.4, next version also 8.3
Some Particle Guns

Experiments implement specific settings and generators, e.g. EvtGen in LHCb

Performance with Pythia8 as generator engine
shared (P8) vs thread local (P8MT) , pp at various energies



CERN-THESIS-2024-01, M. Mazurek

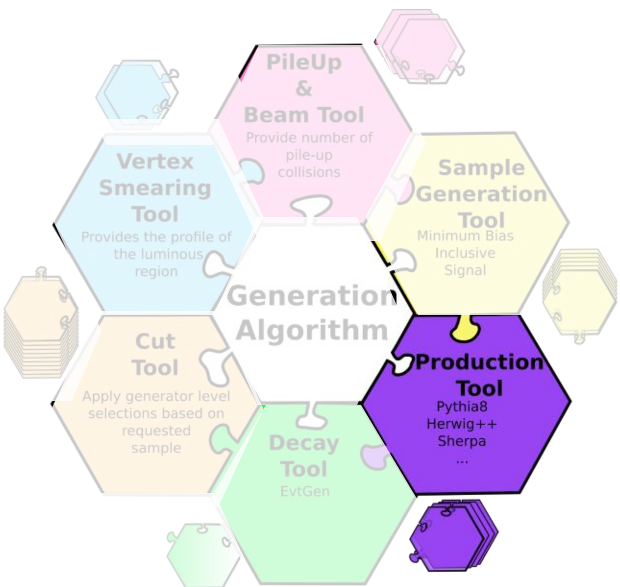
Production tools for Matrix Element Generators

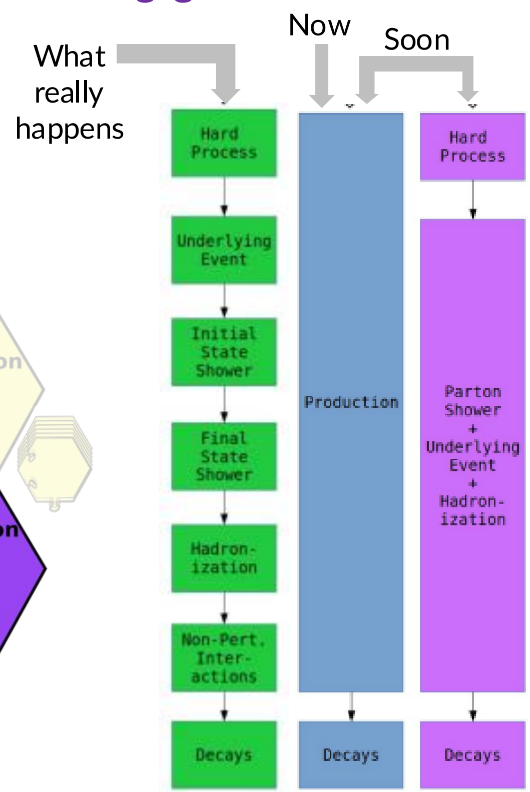
Refactor interfaces to introduce flexibility and easier combination of a
Hard Process ME and **Showering generators**

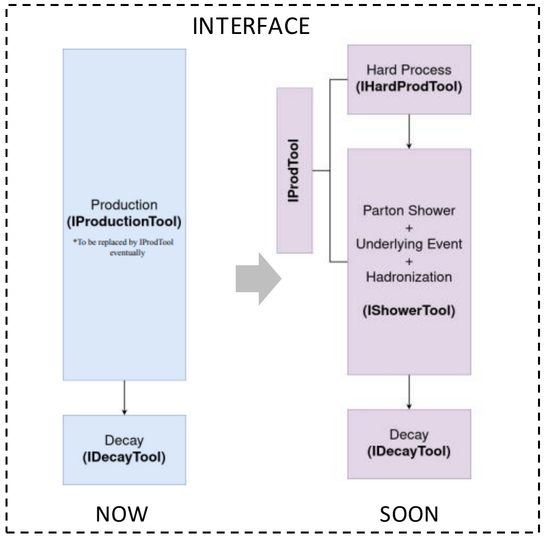
POWHEG-BOX
Madgraph5@NLO
SuperChic
...

\+

Pythia 8
Herwig 7
Sherpa

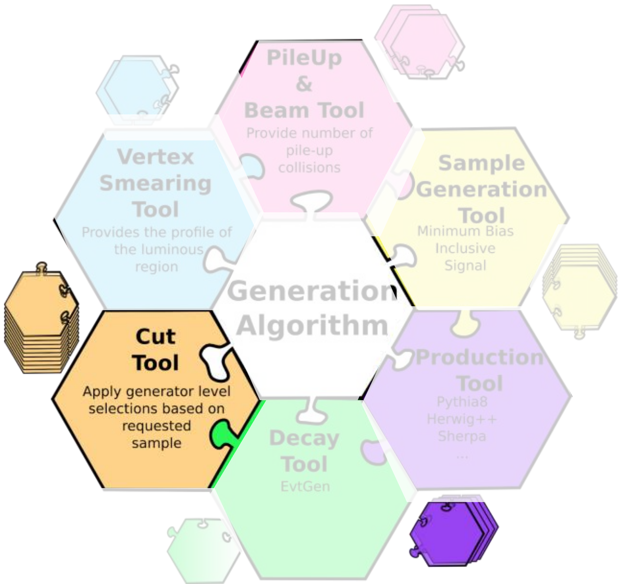






Implementation of POWHEG-BOX as HardProcess and Pythia8 as Shower working.
MadGraph to be tested

Generators Cuts with HepMC3



Generator-level cuts save time by rejecting events before detector simulation

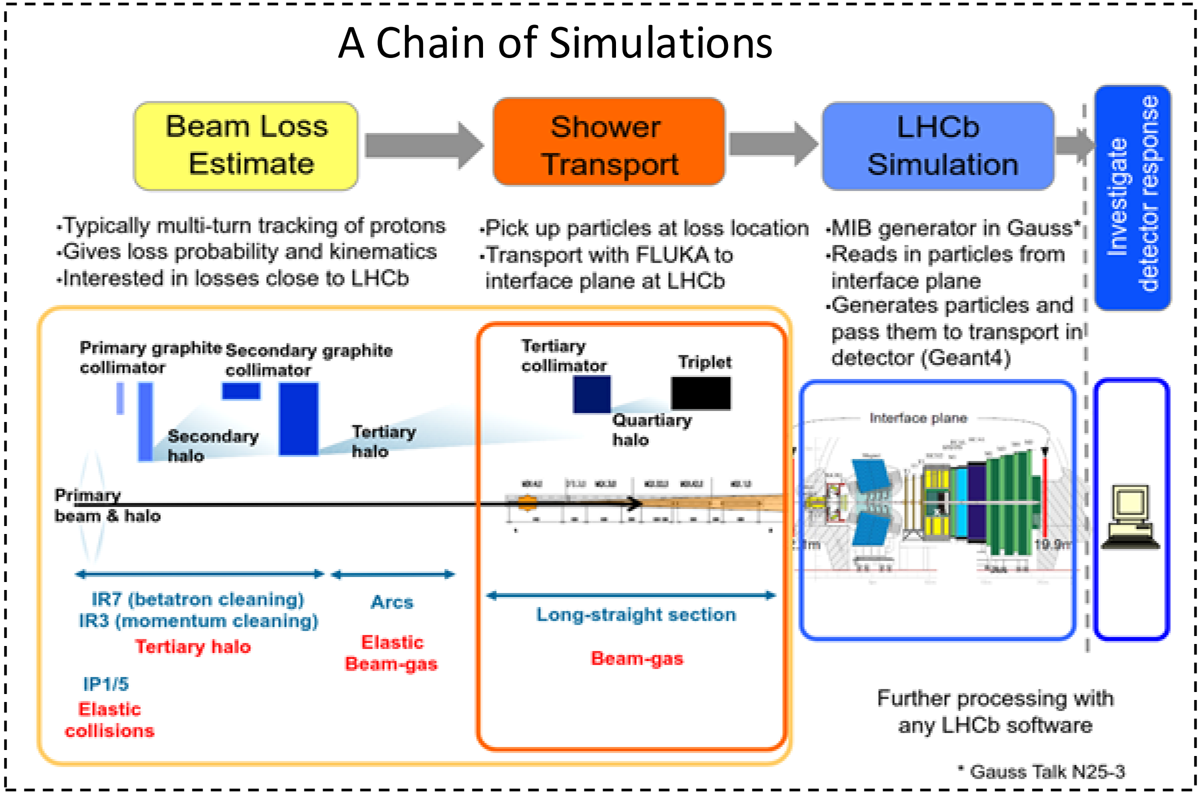
Signal vs Full Event

- A **signal** sample has a particle of a desired type
- The GenCutTool class applies cuts to the signal particle
- Cuts on the **full event** are applied with the FullGenEventCutTool

Generic cut tool

- Configure arbitrary cuts at run-time. In Gauss it used an LHCb custom software
- **Soon** in Gaussino
  - exploit native functionality for cuts of **HepMC3 Filters**
  - define a standard list of named Features in Gaussino's HepMCUser package

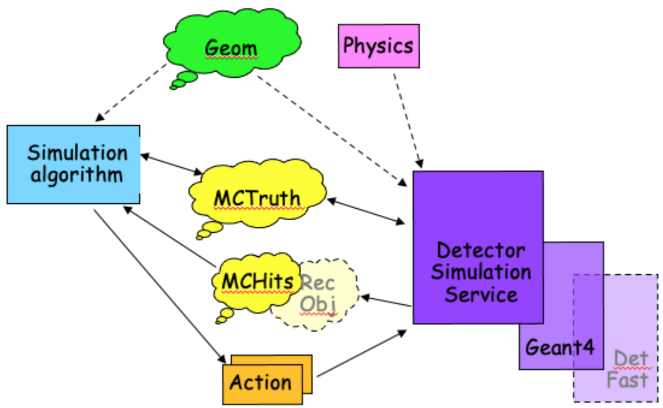
Machine-Induced Background



Ported to Gaussino LHCb MIB Generator

- Loads beam loss events from a ROOT
- Samples a Poisson-distribution for number of MIB interactions in event
- For each interaction:
  - Generate a loss event based on *LossW*
  - Generate particles based on *PartW*
- Assign spatial and kinematic properties
- Store particles in a HepMC3 event to be transported through the detector via Geant4

Simulation Phase



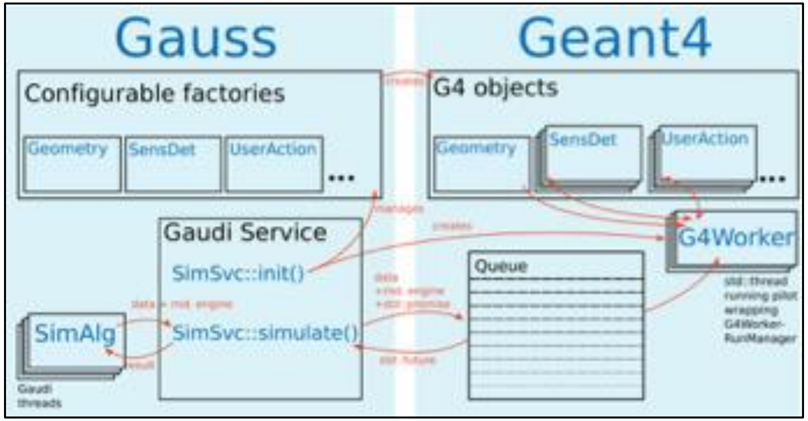
Geant4 multi-threading

- Currently based on event tasking and 10.7.4
- Move to 11.3.2 in the next+1 release

Gaudi tools as factories for Geant4 objects

- Geant4 manages its objets

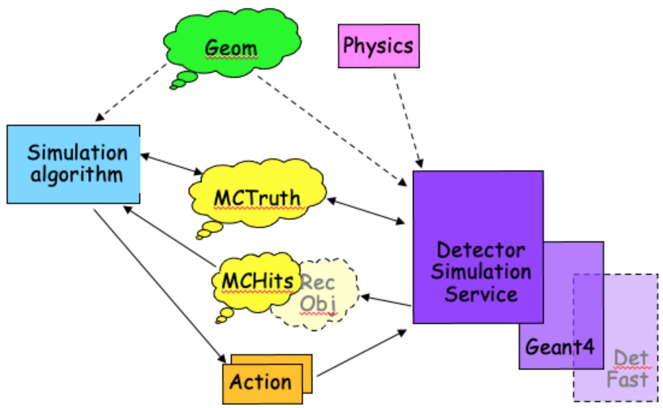
Experiments choose geometry and Geant4 setting, and implement fast simulations



Simulation & Geometry services to steer different backends

Flexible python configuration to combine different setting, e.g. for in time/out of time pileup, fast simulations, Geant4 physics, …

Simulation Phase



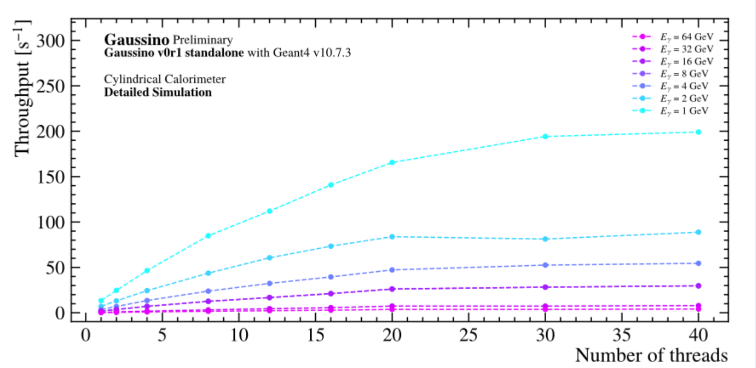
Simulation & Geometry services to steer different backends

Flexible python configuration to combine different setting, e.g. for in time/out of time pileup, fast simulations, Geant4 physics, …

Geant4 multi-threading
- Currently based on event tasking and 10.7.4
- Move to 11.3.2 in the version after next

Gaudi tools as factories for Geant4 objects
- Geant4 manages its objets

Performance for a generic cylindrical calorimeter geometry with γ of different energies



CERN-THESIS-2024-01, M. Mazurek

Exploring different tasking, e.g. inter-event and G4 TBB in the plan

MC Truth

- Consistent MC truth for particles from
  - Generator level only
  - Generator level but modified by Geant4
  - Created in Geant4

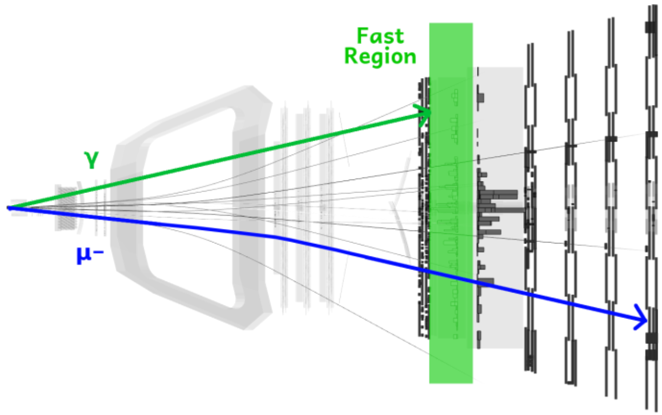
```
omega(782) HepMC|G4Primary|G4Truth = 1|0|0 CONV   = MC
|--- pi- HepMC|G4Primary|G4Truth = 1|1|1 CONV   = G4
|---|--- e- HepMC|G4Primary|G4Truth = 0|0|1 CONV   = FROMG4
|--- pi+ HepMC|G4Primary|G4Truth = 1|1|1 CONV   = G4
|--- pi0 HepMC|G4Primary|G4Truth = 1|0|0 CONV   = MC
|---|--- gamma HepMC|G4Primary|G4Truth = 1|1|1 CONV   = G4
|---|---|--- e+ HepMC|G4Primary|G4Truth = 0|0|1 CONV   = FROMG4
|---|---|--- e- HepMC|G4Primary|G4Truth = 0|0|1 CONV   = FROMG4
|---|--- gamma HepMC|G4Primary|G4Truth = 1|1|1 CONV   = G4
```

- Keep the G4 history in HepMC3 structure while processes occur
- Linking of hits to particles
- Conversion to final event model
  - LHCb::MCParticles, MCVertices, MCHits

Gaussino native event model planned, with experiments to chose persistency

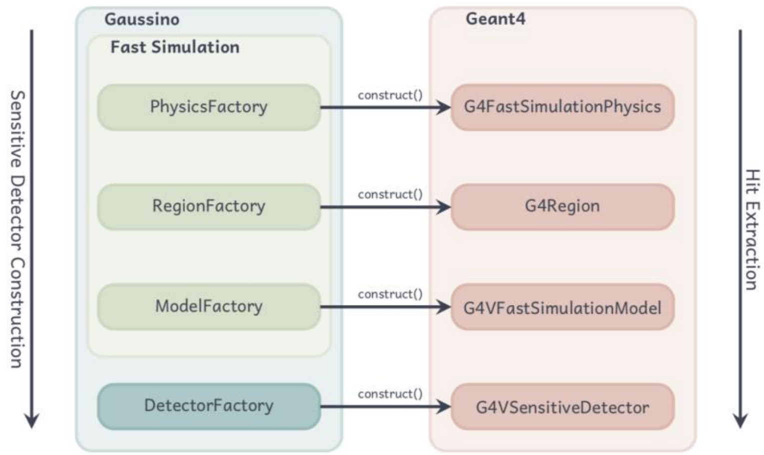
Interface for custom physics

The interface steers where to use custom physics instead of Geant4



Dedicated high-level configurable

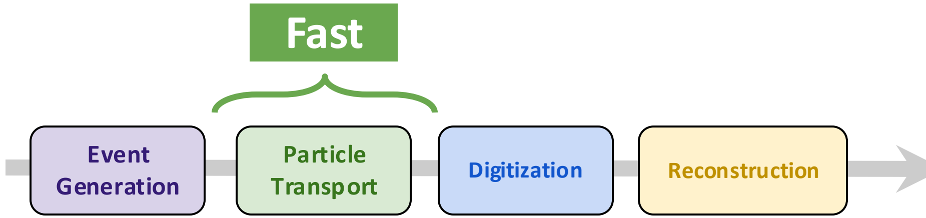
Factories in C++ using Gaudi tools that configure Geant4 objects



- for which particle to do it
- in which region to do it
- how to do it
  - particle and track conditions
  - hit generation algorithm

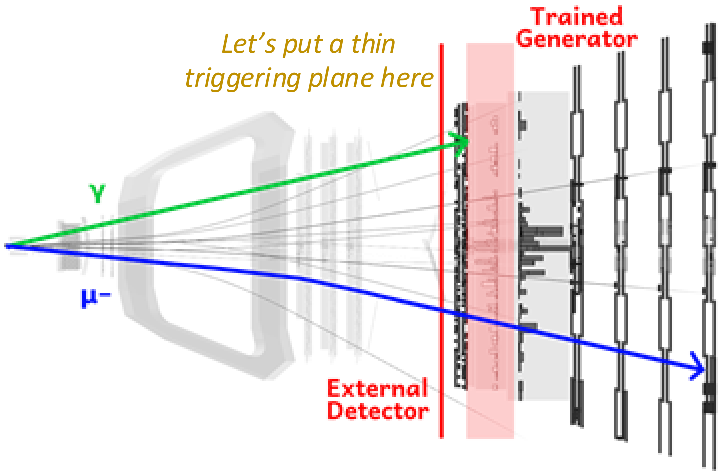
Allow use of fast simulation models in a given detector

Machine Learning Fast Simulation, the LHCb example



ML fast simulation in Calorimeter

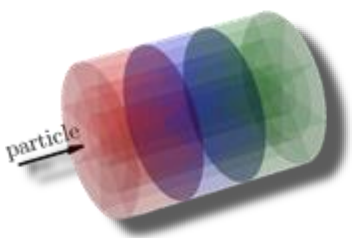
- **stop detailed simulation** in a particular region of the detector
- use a **parametric model** to produce **similar output**

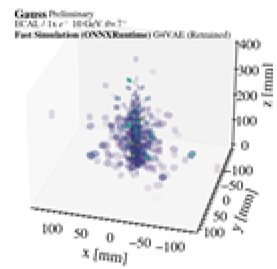


- **train an AI model** on the output
- produce hits by running **inference** on the **generator** part

Generic showers in LHCb with CaloChallenge

Build a cylinder of virtual hits
around a particle shower



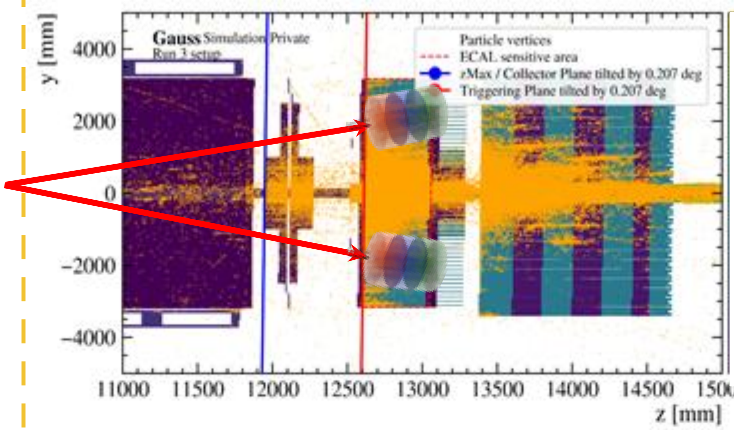


making things generic = minimizing the work later

- **compare** models **objectively** across geometries
- **adapt** easily by **retraining** on LHCb's calo data

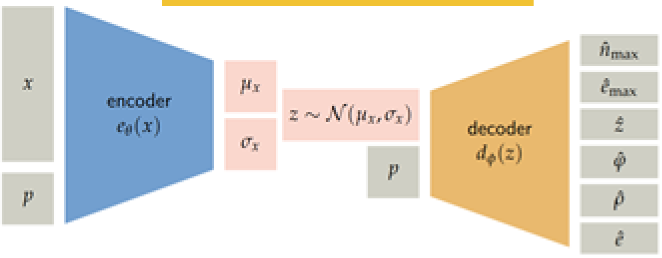
arXiv:2410.21611
MetaHEP, A. Zaborowska et al.

LHCb Training Data

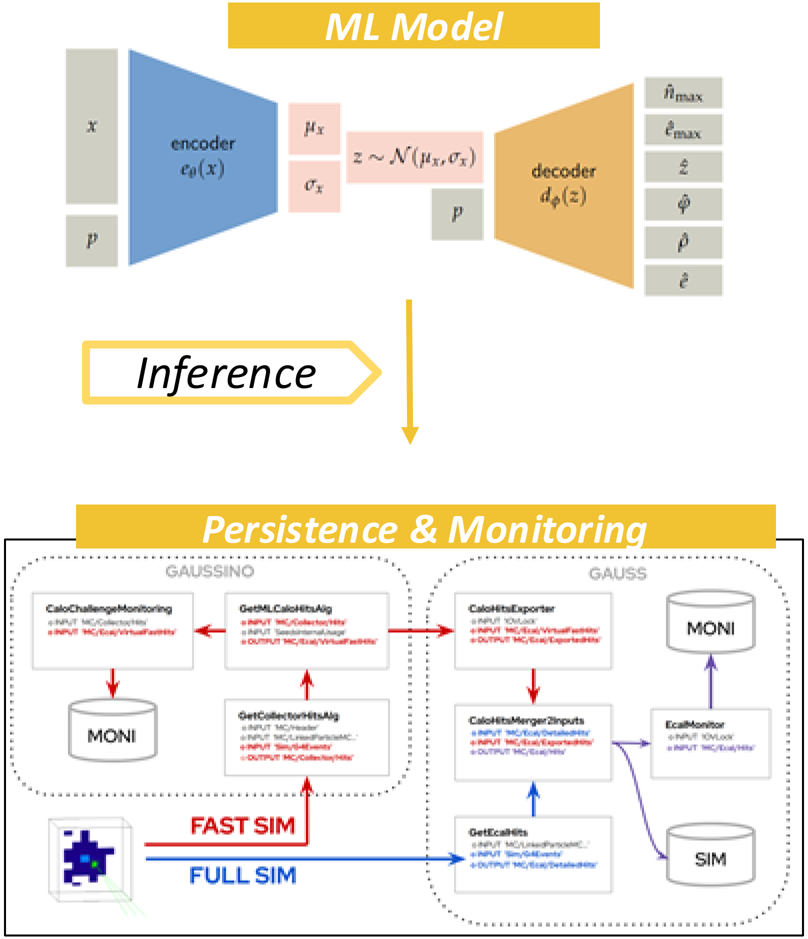


Training

ML Model



Inference workflow



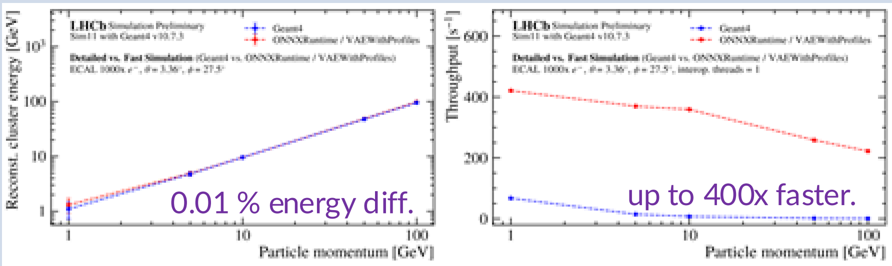
Gaussino

- interface to **fast simulations** to a given detector
  → via Geant4
- interface to **ML libraries** CHEP 2023 talk →
  pyTorch and ONNXRuntime

Gauss

- introduces the **LHCb geometry & conditions**

Performance for single e^-

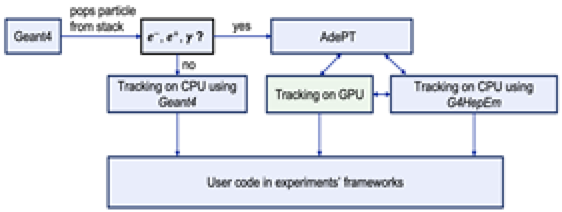


First official productions planned this year

AdePT

- AdePT - a lightweight plugin to Geant4 allowing to offload the simulation of **EM particles to GPU**
  - no changes in the user code needed (caveat: tracks are processed in parallel)
  - But it has to be integrated in the experiments' frameworks
- It offers a **flexible CPU-GPU workflow** as required by the experiment frameworks

Seamless offload of electrons, positrons, and gammas



Project components

- Core infrastructure, scheduling and transport: AdePT
- Physics: G4HepEm
  - Compact rewrite of EM processes, focusing on performance and targeted at HEP detector simulation applications adapted for GPU
- Geometry: VecGeom
  - GPU adaptation built on top of the original VecGeom GPU/CUDA support
  - Includes several GPU-focused improvements, like an optimised navigation state system, and a BVH navigator
  - New surface model now available as alternative to the solid model
- Magnetic field map: covfie
  - 3D field map, we are using a Runge-Kutta propagator on top of it

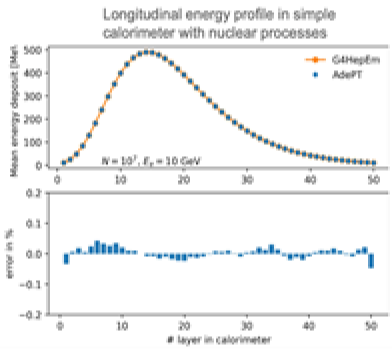
Physics correctness

AdePT relies fully on **G4HepEm physics*** in custom tracking loop including $e^{-/+}$- and γ-nuclear processes

* does not mean that we get the same physics results due to custom tracking loop, VecGeom, and B field propagation

Fallback to CPU to G4HepEm:
- Nuclear processes
- Avoiding certain regions (custom physics, FastSim, slow geometry)

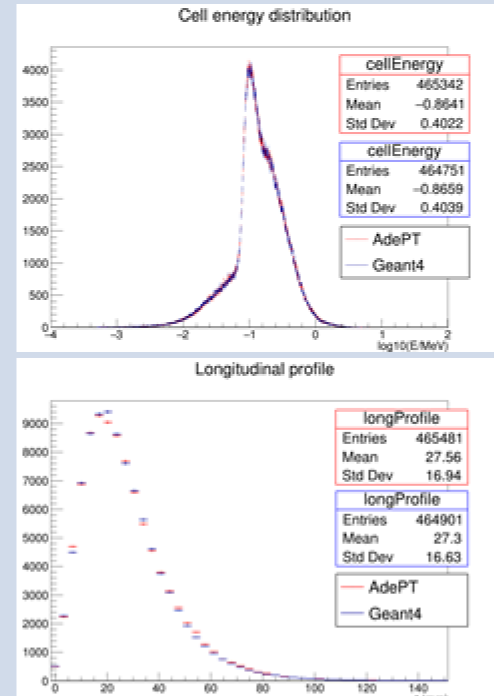
Longitudinal energy profile in simple calorimeter with nuclear processes



AdePT in Gaussino

- Fully integrated in Gaussino
  - Requires additional external packages with build dependencies and access to GPUs
    - VecGeom and G4HepEM with Geant4
  - One line in **configuration** to switch it on
  - To be deployed in next+1 Gaussino release

- CaloChallenge setup
  - Physics results showed the expected agreement with Geant4
  - For enough particles sent to the GPU, the gains can be significant
  - Achieved 5x speedup with 4 CPU threads in initial tests with gamma-only (1000 particles/event) events
    - Disclaimer: many improvements have been made in AdePT since that study

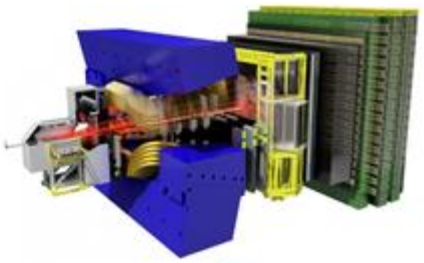


AdePT in LHCb

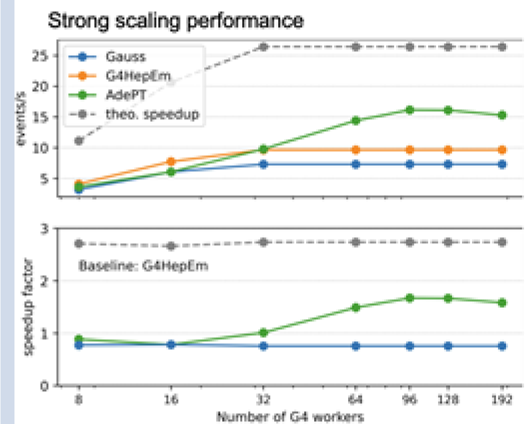
- Use as out-of-the box as Geant4 from Gaussino
  - Geometry and configuration in Gauss

```
Geometry: full LHCb detector
Field: 3D field map (v6r1 down)
Input: 20000 min bias events
GPU: Nvidia RTX4090
CPU: AMD Ryzen 9 16 cores

Full production settings except
MC truth
```



Computing Performance

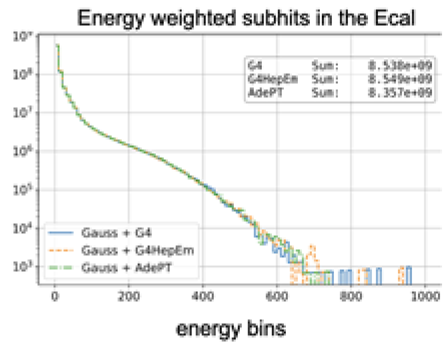


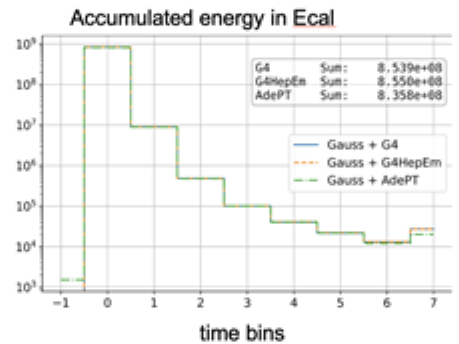
The GPU is **underutilized** in minimum bias events in Gauss

By oversubscribing the CPU, the GPU can be filled!

1.7x speedup w.r.t. G4HepEm, **2.1x** w.r.t. Gauss

Physics Performance

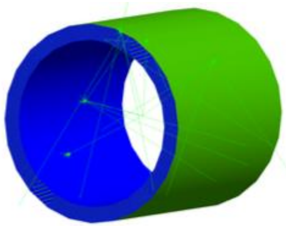


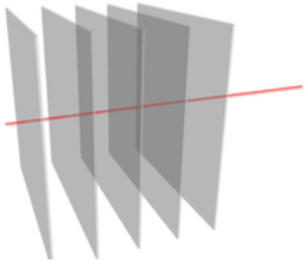


Geometry

Generic service to steer passing the information to Geant4 from different backends

- **Backend** service for **DD4Hep**
- Import & export of **GDML** files
- Custom service for '**internal**' volumes of simple shapes
  - works **stand-alone**
  - can be **mixed** with other geometry services
  - supports Geant4 **parallel worlds**
  - used for **fast simulations**



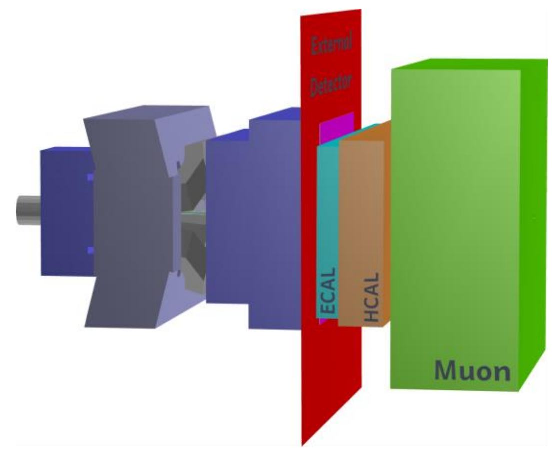


Geometry

Experiments can use or extend it

In LHCb, **Gauss[-on-Gaussino]** takes care of

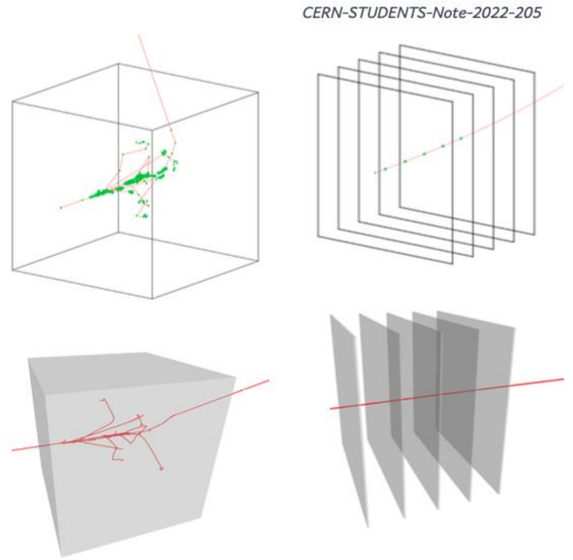
- high level service to **configure** the geometry to simulate and sensitive detectors
- extension for **legacy** Detector Description for Run 1&2 and 3
- extension for **new** description using DD4Hep for Run3 and beyond



Visualization

Geant4 visualization drivers
- Available at **runtime**
- Volume **overlap checks** possible
- **G4 data** only
- Drivers: ASCIITree, OpenGL, DAWN, HepRep

Geant4 Users' Guides



Phoenix event display
- Available as **external** tool
- Geometry to be converted from **GDML** to a dedicated format
- both **Geant4 or experiment-specific** simulated data
- Data exported to **JSON**
- **Simulated vs. Reconstructed** data comparison possible

Dedicated steering due to Gaudi & Geant4 multi-threading interplay
- visualisation has its own thread
- information exchange at the right time

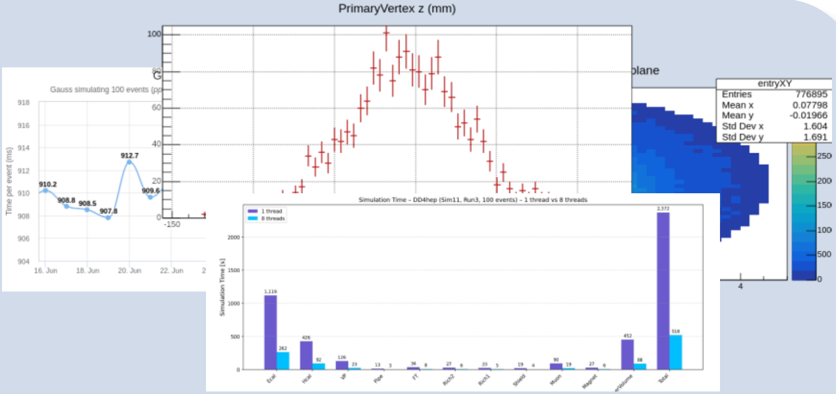
Monitoring & Output

Various persistent output formats possible with predefined contents

- conversion to in memory event model with consistent Monte Carlo ‘truth’
  - from generators
  - from Geant4 choosing what to keep
- ROOT tuples and histograms
- HepMC3 generator output and EDM4hep sometimes in the future

Continous **monitoring** of the produced simulation **samples** and software **performance** via the **LHCbPR** automatic tool.

Gaussino stand-alone tests in the plan



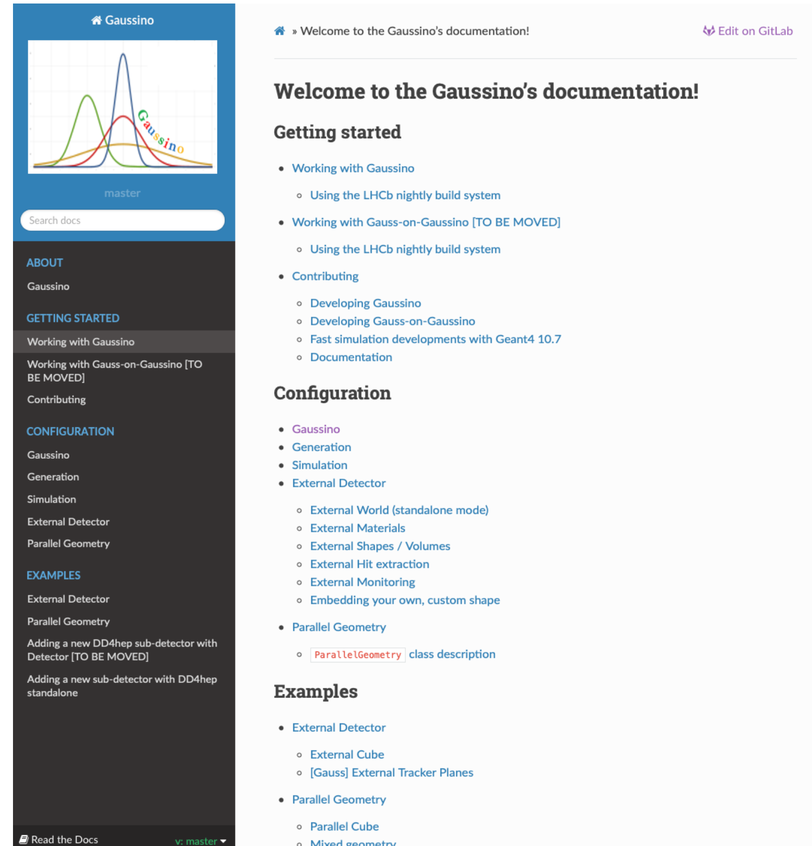
Simulation Quality in ACAT2022 Poster by D. Popov

Documentation

https://gitlab.cern.ch/Gaussino/Gaussino

https://gaussino.docs.cern.ch/gaussino

- New developments in Gaussino are documented
- Versioning of the documentation
- Provides
  - how to install and run
  - description of high-level python configurations
  - simple examples



https://mattermost.web.cern.ch/gaussino

gsino-users@cern.ch

Conclusions and outlook

- The experiment-independent simulation framework, Gaussino, is **becoming** mature for **general use**
  - Removing remaing dependecies from LHCb software and clean-up slowly starting
- It can be used **stand-alone** and as an **intermediate layer** for experiments specific simulation frameworks
  - Gaussino is built on the **Gaudi** framework and provides an infrastructure for **generators** and **Geant4-based** detailed simulation
  - It now provides hooks for **Machine Learning** fast simulation and **AdePT**
  - The **generator** interfaces are **evolving** for better support of ME generators
- LHCb will soon deploy Gauss[-on-Gaussino] for production
  - The evolution of the LHCb simulation based on Gaussino providing LHCb-specific additions and configurations