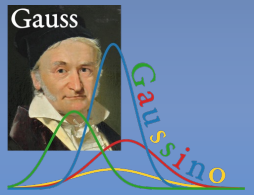


LHCb simulation

Monitoring simulation performance
& selected Geant4 related WIP to improve it

Prepared by
G. Corti, M. Mazurek, W. Pokorski, D. Popov, M. Veltri
on behalf of the LHCb Simulation Project

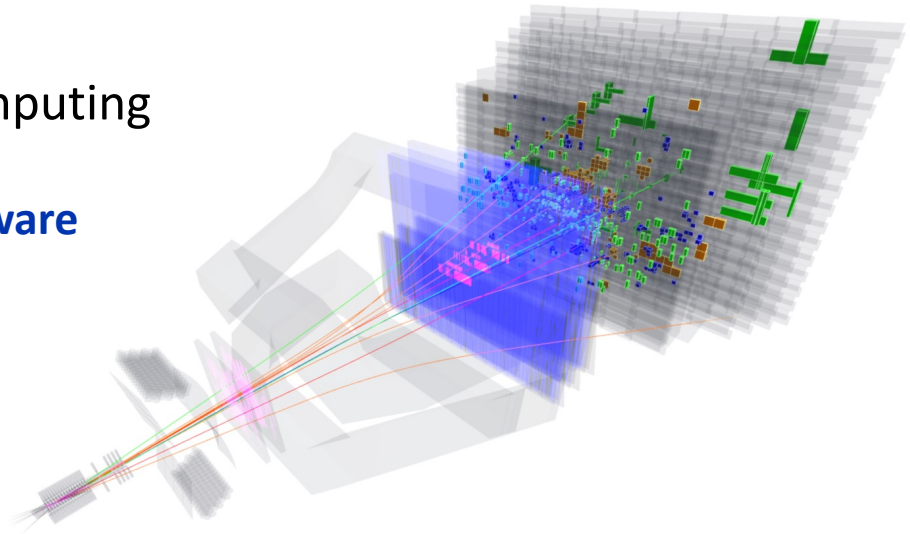
Setting the scene …

LHCb Upgrade in Run3

- full software trigger with high signal purity
- analysis directly on trigger output

L_{inst}: 4 10^{32} → 2 10^{33} cm^{-2} s^{-1}
μ: 1.1 → 7.6
L_{int}: 3(Run1) + 5(Run2) fb^{-1} → 50fb^{-1}

Very challenging for software & computing



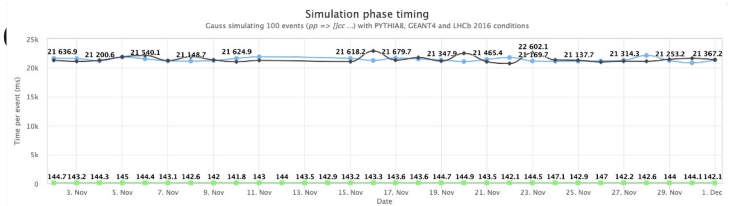
Modernization of the whole LHCb software

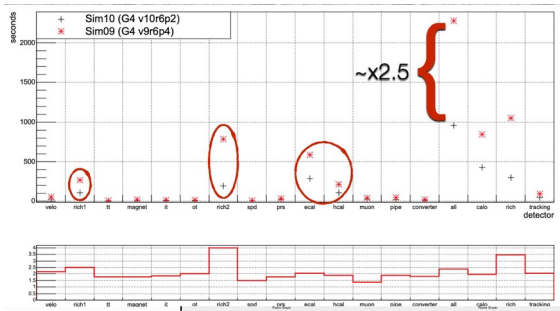
- Multi-threading
- Better use of multi-processor CPUs
- Reduce **memory usage**
- Optimize cache performance
- Remove dead code
- Modern data structures
- Enable **code vectorization**
- Enable **algorithmic optimization**
- HLT1 reconstruction on **GPUs**

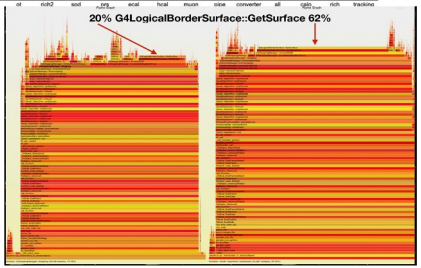
Infrastructure

LHCb Performance Regression Tests

- Infrastructure in place with nightly builds to che changes as they come in
  - CI -> Nightly builds -> QM/PyTest -> LHCbPR
- Run LHCb applications, various configurations
  - MC generators, LHC conditions, detector geometry
- Typical monitoring properties:
  - Application behaviour
    - Timings, CPU/memory profiling, stack traces sampling (planned)
  - And relevant physics distributions
    - Numbers of tracks/vertices, momentum, energy deposits
- One nightly build setup to adapt to and explore newer versions of Geant4
  - 10.6.4 in production, 10.7.3 in nightly,
  - Plan to add 11 in near future

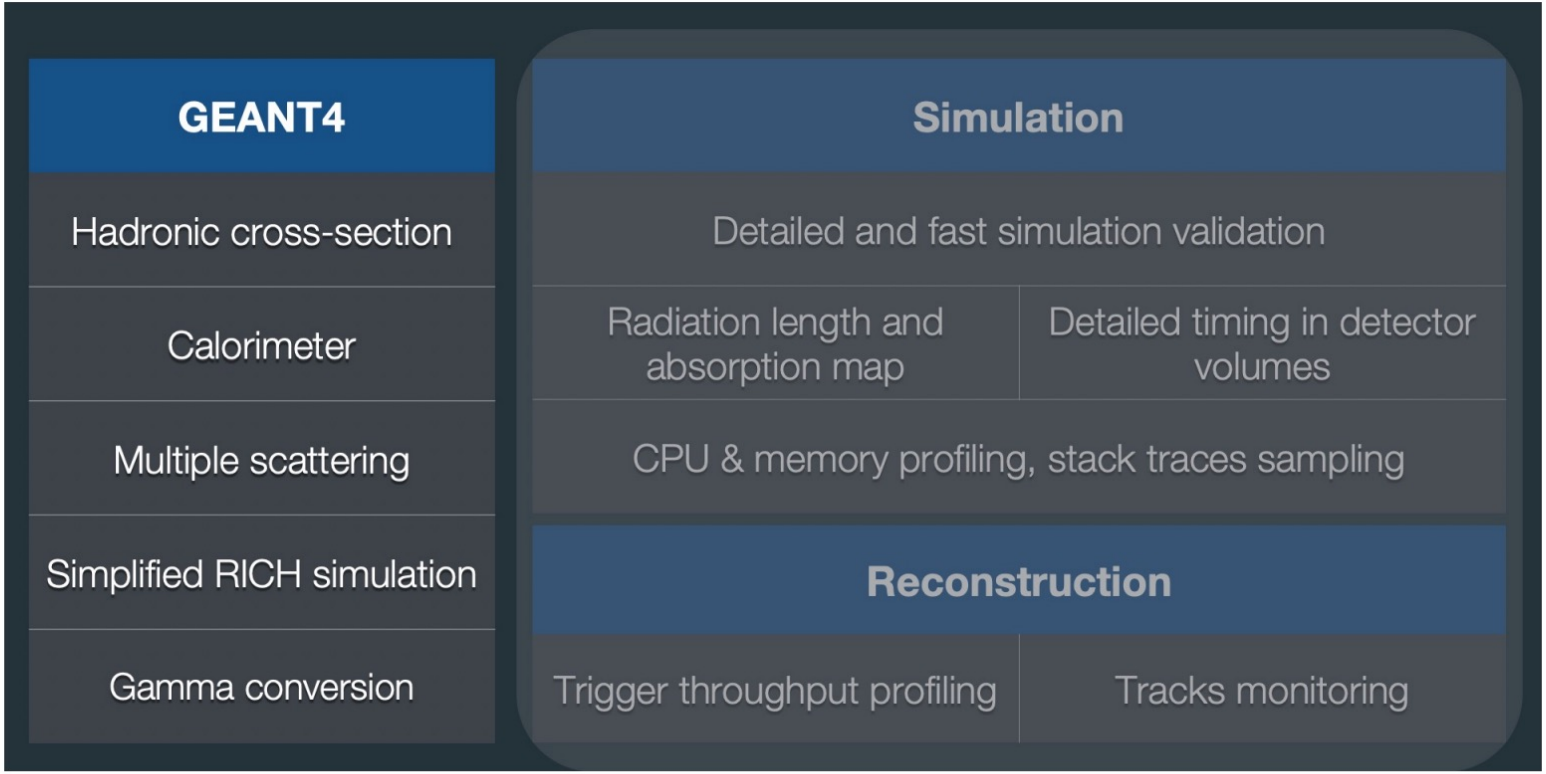






WIP to include profiling with flamegraphs to pinpoints CPU hotspots

LHCbPR Geant4 tests



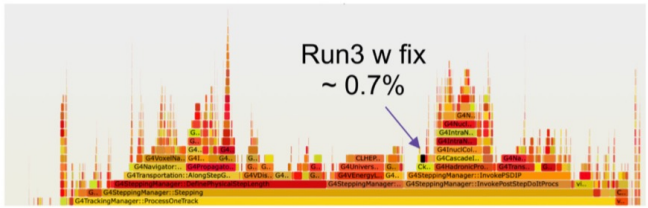
Some G4 tests also within Gauss.
Increase tests suite as necessary

LHCbPR Geant4 tests

Presented at G4 TF in Spring

New release of Gauss v56r3

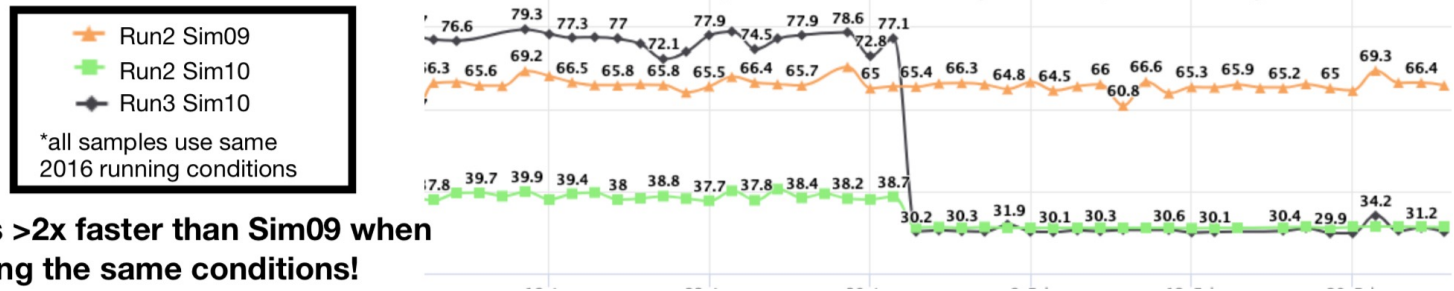
- Incorporates speed up of Geant4 simulations by A. Valassi
  - Lots of time spend on `G4LogicalBorderSurface::GetSurface`
  - More RICH PMTs in Run3 —> bigger impact



Fix: newer version of Grant4 performs a look up using `std::map` rather than `std::vector`

→ back-ported to current version

Impact of detector geometry on timing:



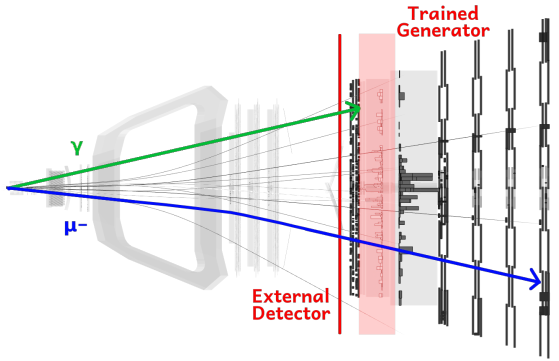
Sim10 is >2x faster than Sim09 when simulating the same conditions!

Fast Simulations

Fast Simulations and ML with Geant4

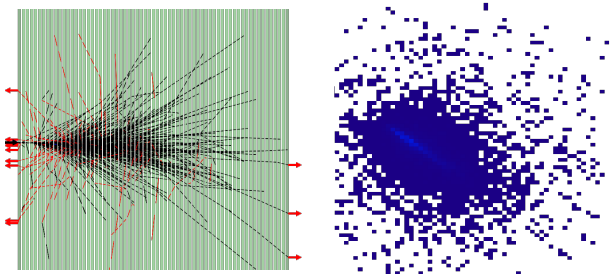
Fast simulations with Geant4:

- Stop detailed simulation in a particular region of the detector for a given type of particles
- Use machine learning to produce the output

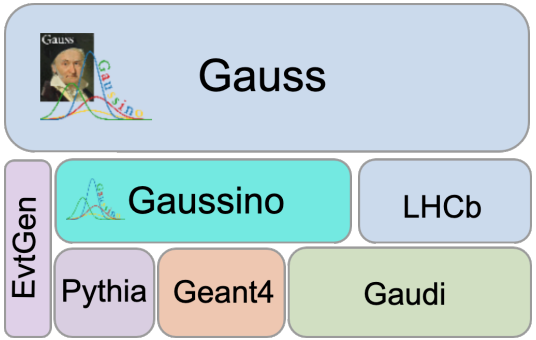


And machine learning

- **Train a ML** model to be able to produce the same output as Geant4
- Produce hits by running inference on the generator
- **Interface to machine learning libraries !!**



ML model serving with Gaussino



Gaussino new experiment-independent Core Simulation framework

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

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

Machine model serving using a Gaudi's Service:

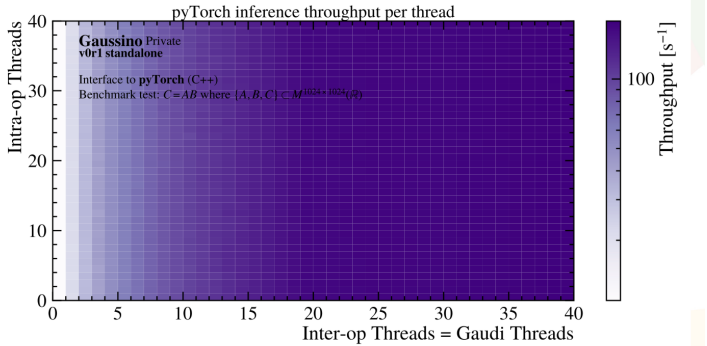
- Loading of the model
- Setting ML backend's general properties
- Handling multithreading

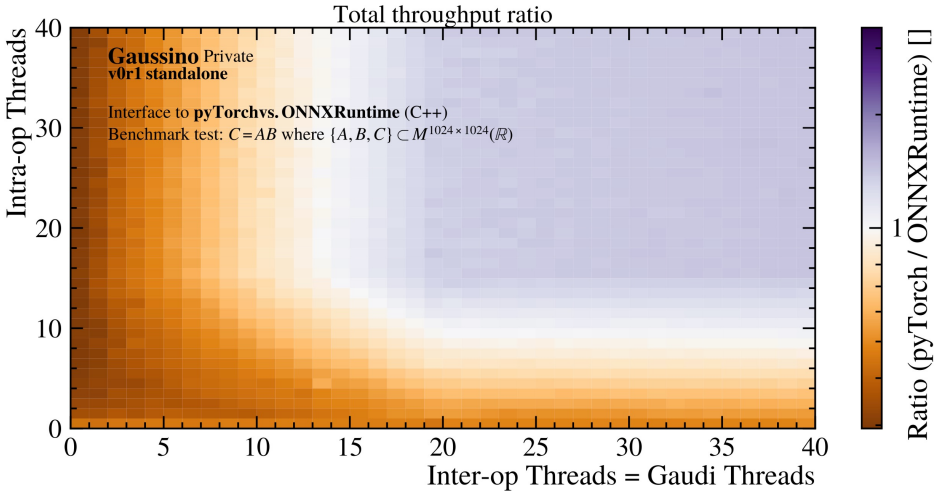
Machine model evaluation using Gaudi's tools and algorithms

- Provide the right input
- Handle the output

Machine Learning interface

Tested both pyTorch and ONNXRuntime in Gaussino

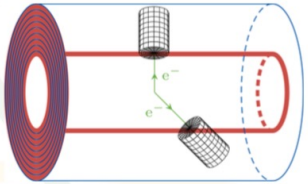




CaloChallenge

CaloChallenge idea

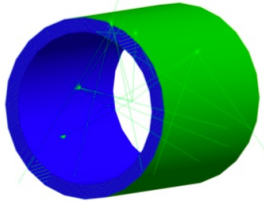
- new Geant4 initiative: **CaloChallenge** (more info)
- train on **experiment-agnostic training dataset**
- **compare** various **models objectively**,
- **retrain** the chosen model on the **target geometry**!

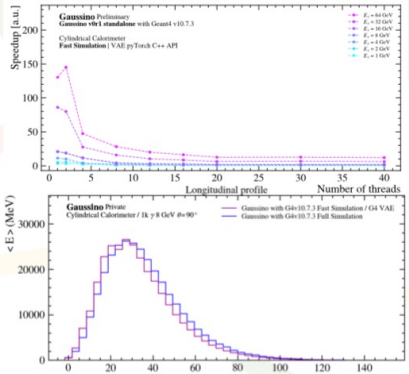


Credit: W. Pokorski (Geant4 Technical Forum)

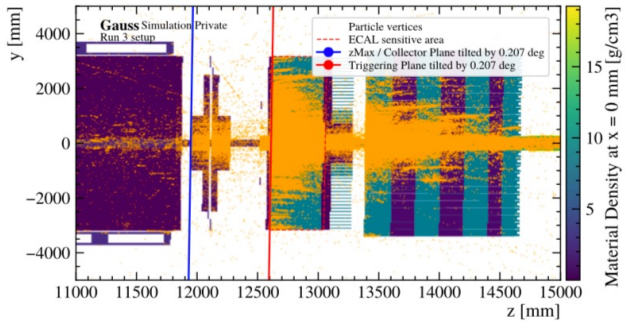
First benchmarks in Gaussino

presented at CHEP talk

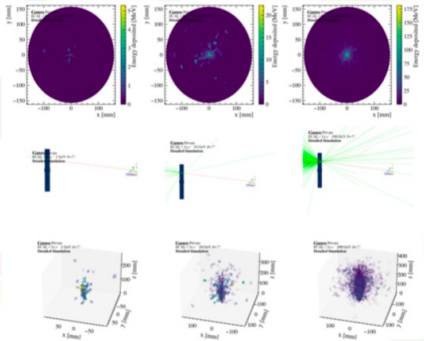




CaloChallenge for the LHCb calorimeters: solution



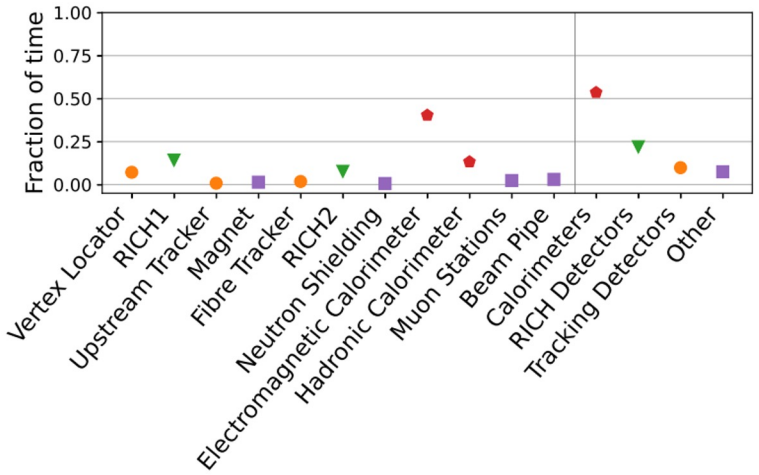
ECAL: detailed simulation



GPUs

GPU based simulation in LHCb

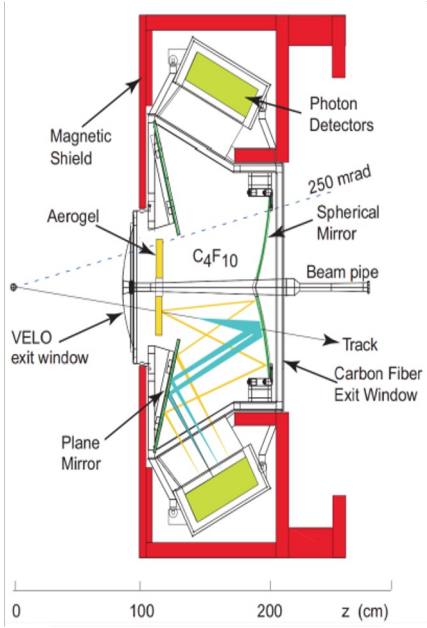
- In the LHCb experiment the detailed simulations, based on Geant4, dominate the use of computing resources
- In particular two are the processes responsible:
  - Electromagnetic showers inside the calorimeters
  - Optical photons transportation in the RICH detectors



Investigating a hybrid workflow scheme with the use of GPUs to tackle the issue for EM showers and optical photons leveraging on HEP R&D

GPU based optical photons simulation in LHCb

- Tests performed with **OPTICKS**, that provides an interface between Geant4 and the NVIDIA OptiX ray tracing engine to simulate photon propagation while maintaining the simulation of other particles on CPU



- Two RICH detectors in LHCb for PIDs
  - filled with different gases for different momentum ranges
- Equipped with a dual set of mirrors
  - Primary: Spherical and tilted → Light to secondary mirror
  - Secondary: Planar → Light to photodetectors

- Use a simplified version of RICH1 to validate the performances of *OPTICKS* and check its consistency with Geant4
  - No beam pipe, no exit window, only the upper part is used
  - Include mirror reflectivity and MaPMT quantum

GPU based optical photons simulation in LHCb

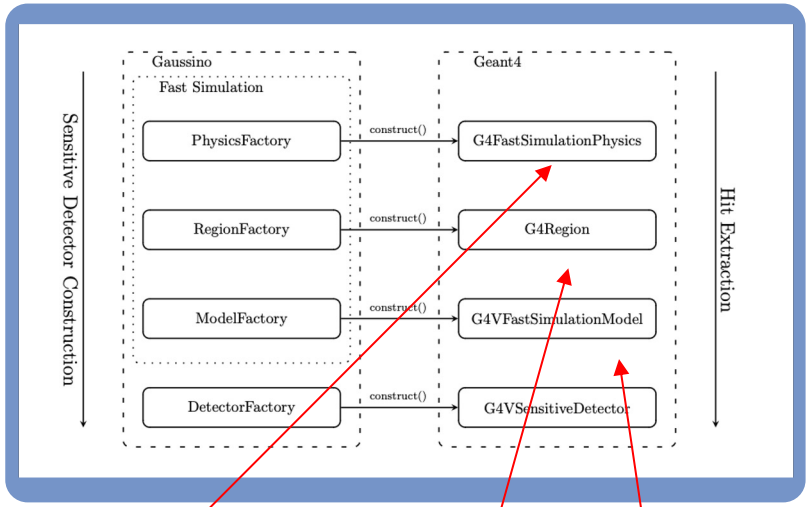
- The integration of OPTICKS into the LHCb simulation framework presents a considerably challenge which requires further effort and investigations, in particular
  - The use by OPTICKS of external packages which are required to be specific version to avoid conflicts for use in distributed computing resources
  - The necessity to develop an interface between OPTICKS and the LHCb core simulation framework

- Preliminary work with Mitsuba3 as an alternative
  - Proof-of-principle pipeline with the simplified settings

AdePT integration

- Ongoing effort with EP-SFT to integrate AdEPT into Gaussino via the Geant4 CustomSimulation (a.k.a. fast) simulation hook
  - Witek Pokorski working on it with LHCb student Juan Bernardo Benavides

- Gauss-on-Gaussino 'CustomSimulation' machinery configure and triggere a fast simulation model in Geant4
- AdePT example (17) demonstrates the use of the Geant4 fast simulation hook to call AdePT with LHCb GDML geometry
- Combine them to fill the AdePT pipeline with the particles entering the calorimeter region and then give back the whole simulated information to Gauss/Gaussino to generate Gauss hits
  - They should be equivalent to the ones generated in a plain Gauss (Geant4) simulation



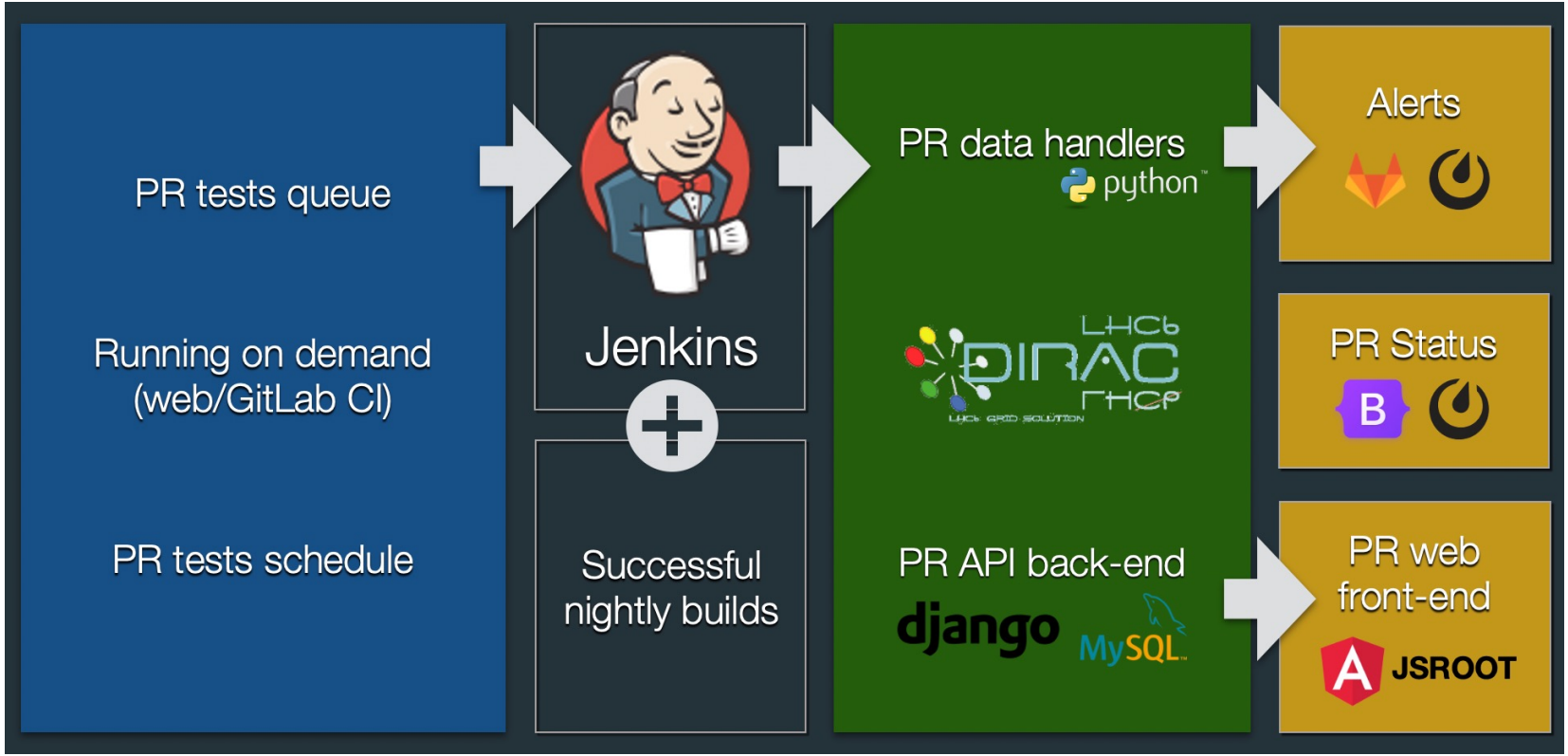
define the region where AdePT takes over the simulation

plug AdePT in

implement a special 'sensitive detector' processing AdePT output and creating Gauss hits

BACKUP

LHCbPR testing



Size of data samples are a compromise between time and statistics
Store results in various forms: basic types/JSON/files (e.g. ROOT)