The LHCb simulation application, Gauss: design, evolution and experience

M. Clemencic (CERN), G. Corti (CERN),
S. Easo (RAL), C. Jones (Cambridge),
S. Miglioranzi (CERN),
M. Pappagallo (Bari), P. Robbe (LAL)

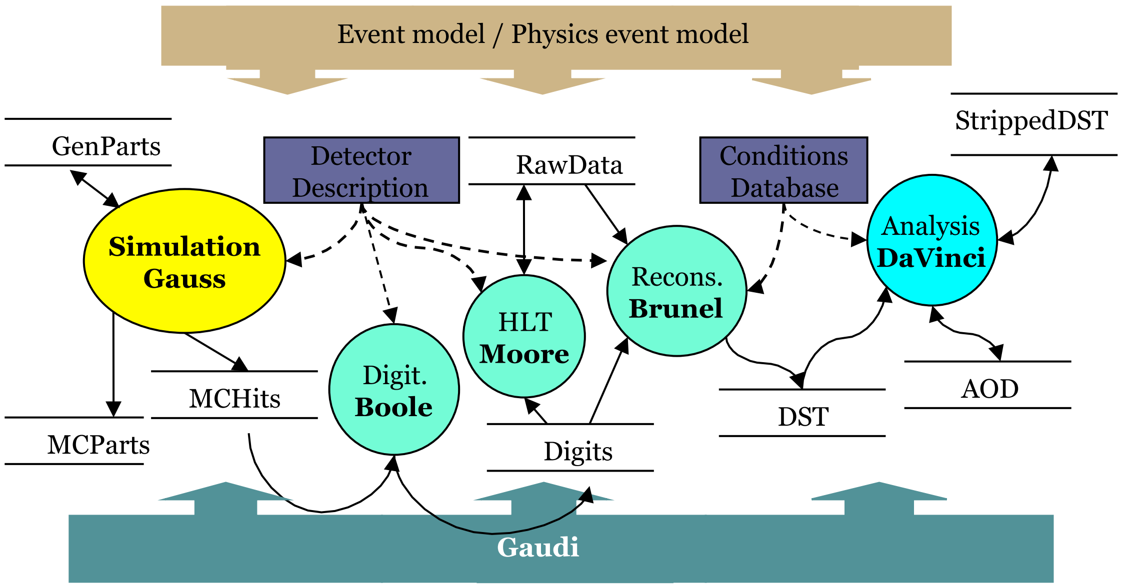
The Gauss simulation application

Gauss, the LHCb simulation application, mimics what happens in the spectrometer to understand experimental conditions and performance.

It provides:

- generation of proton-proton collisions
- decays of particles with special attention to B decays
- tracking of particles in the detector and interactions with the material
- production of “hits" when particles cross sensitive detectors

Data produced can be studied directly or in further processing (digitization, reconstruction,…)

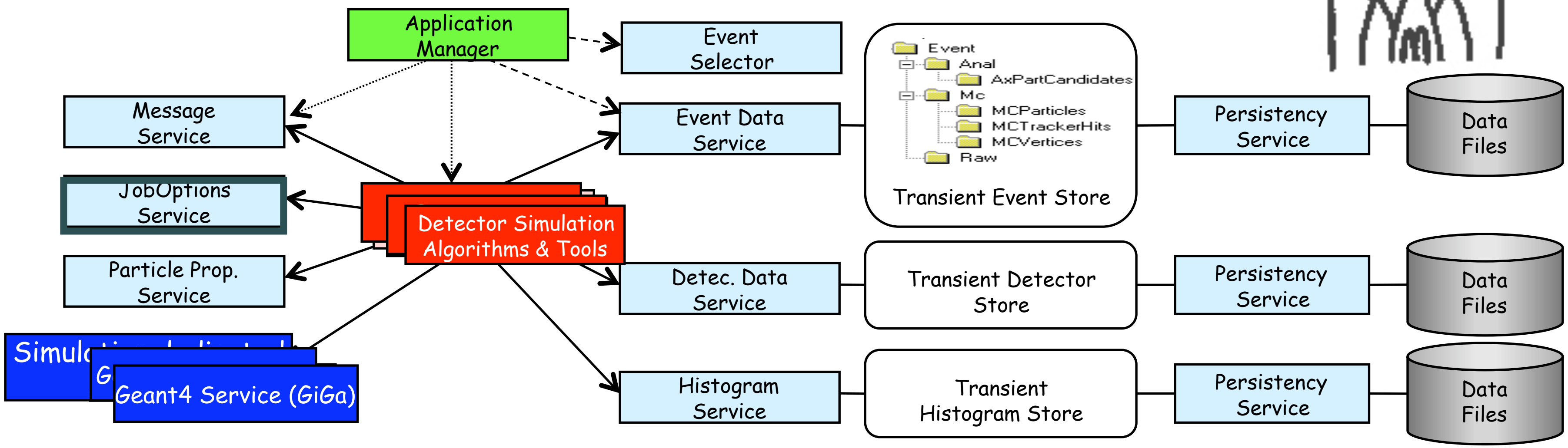


Gauss as a Gaudi Application

Gauss is built on top of the Gaudi framework and follow its architectural design:

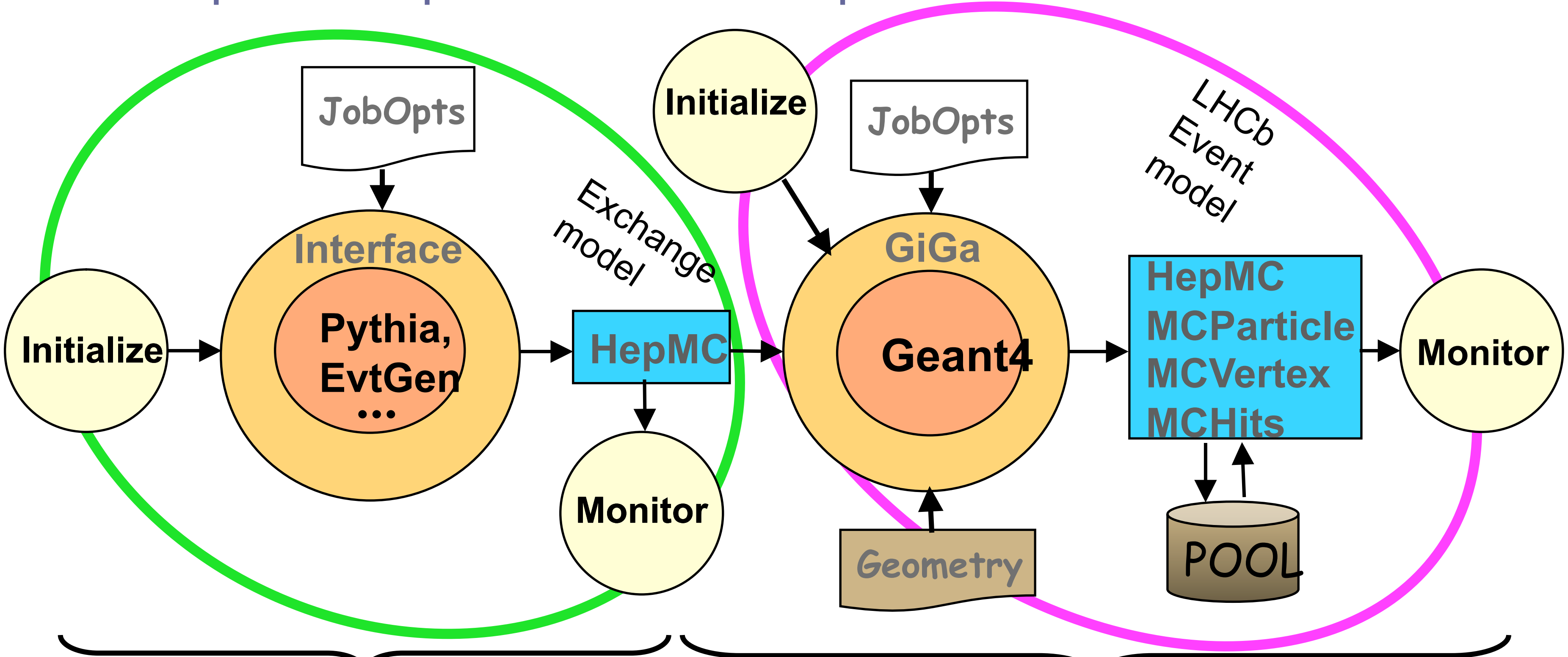
- Separation between "data" and "algorithms"
- Separation between "transient" and "persistent" representations of data
- Three basic categories of "data": event data, detector data, statistical data
- "User code"(Gauss) encapsulated in few specific places (Algorithms, Tools)
- Well defined component "interfaces"

http://cern.ch/proj-gaudi/welcome.html



The Gauss Project

Two independent phases run in sequence:



Event Generation

primary event generator
specialized decay package
pile-up generation

Detector Simulation

geometry of the detector (LHCb → Geant4)
tracking through materials (Geant4)
hit creation and MC truth information (Geant4 → LHCb)

The simulation sequence

Random number reset

Pile-up number determination (veto empty events)

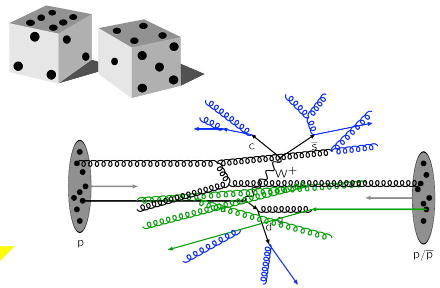
Signal Generation in /Event/Gen/HepMCEvents t=0

Pythia

EvtGen

Vertex Smearing

Generator Sequence



Random number reset

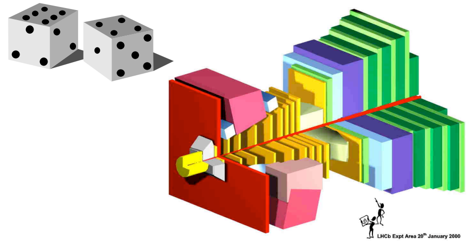
Generation -> G4 Primary Vertex

Simulation inside the detector

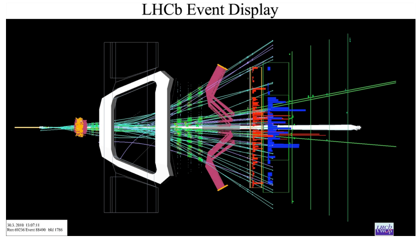
Geant4

Fill MCParticle/MCVertex/MCHits in /Event/MC/Particles...

Simulation Sequence



LHCb Event Display



Digitization

Gauss Python Configuration

In summer 2009 moved to a Python Configuration of the Gauss application:

- Basic option validation (types, names, etc.)
- Programming language (loops, logic, modularization, etc.)
- High level configuration (simple command for complex configuration tasks)

Job option files

- C++-like syntax without checking
- simple parser

↓

Python files

- OOP language
- the Python configuration is based on Configurables
- Configurables are special python classes built from the C++ components (Services, Algorthms, Tools…)
- each Configurable instance has got a name that is unique by construction

Gauss Python Configuration

High level configuration with Gauss Configurables:

consistency checks
can be performed
e.g.
monitoring algorithms run only
for simulated subdetectors

```
# /***** User Gauss/Gauss ******************************************
# |-DatasetName          = 'Gauss'
# |-DetectorGeo          = {'VELO': ['Velo', 'PuVeto'], 'MUON': ['Muon'], '
# |                        (default: {'VELO': ['Velo', 'PuVeto'], 'MUON': ['M
# |-Histograms           = 'DEFAULT'
# |-BeamEmittance        = 1.0100000000000001e-06  (default: 7.0400000000000
# |-Phases               = ['Generator']  (default: ['Generator', 'Simulatic
# |-Production           = 'PHYS'
# |-TotalCrossSection    = 9.1099999999999988e-24  (default: 9.7199999999999
# |-BeamCrossingAngle    = -0.00027  (default: 0.00032900000000000003)
# |-InteractionPosition = [0.0, 0.0, 0.0]  (default: [0.0, 0.0, 0.0])
# |-PhysicsList          = {'Em': 'Opt1', 'GeneralPhys': True, 'Hadron': 'LH
# |                        (default: {'Em': 'Opt1', 'GeneralPhys': True, 'Hadr
# |-DetectorMoni         = {'VELO': ['Velo', 'PuVeto'], 'MUON': ['Muon'], '
# |                        (default: {'VELO': ['Velo', 'PuVeto'], 'MUON': ['M
# |-BeamSize             = [0.044999999999999998, 0.044999999999999998]
# |                        (default: [0.037999999999999999, 0.0379999999999999
# |-DeltaRays            = True
# |-EnablePack           = True
# |-BeamBetaStar         = 2000.0  (default: 2000.0)
# |-Luminosity           = 1.23e+18  (default: 1.16e+18)
# |-CrossingRate         = 1.1245000000000002e-05  (default: 1.1245000000000
# |-DataPackingChecks    = True
# |-InteractionSize      = [0.032000000000000001, 0.032000000000000001, 38.1
# |                        (default: [0.027, 0.027, 38.199999999999996])
# |-Output               = 'SIM'
# |-BeamMomentum         = 3500000.0  (default: 5000000.0)
# |-DataType             = ''
# |-DetectorSim          = {'VELO': ['Velo', 'PuVeto'], 'MUON': ['Muon'], '1
# |                        (default: {'VELO': ['Velo', 'PuVeto'], 'MUON': ['Mu
```

New Generator classes structure

The Generation algorithm is now using tools, *i.e.* modules, which can be plugged in the main algorithm steering the sequence of the event generation:

- Generation of number of pile-up events, IPileUpTool
- Generate N pp-collisions, ISampleGenerationTool
    - Generation of beam parameters, IBeamTool
    - Generation of p-p interactions, IProductionTool
    - Decay of unstable particles, IDecayTool
    - Cut at generator level, IGenCutTool
    - Cut on full event properties, IFullEventCutTool
- Smearing of primary vertex. ISmearingTool

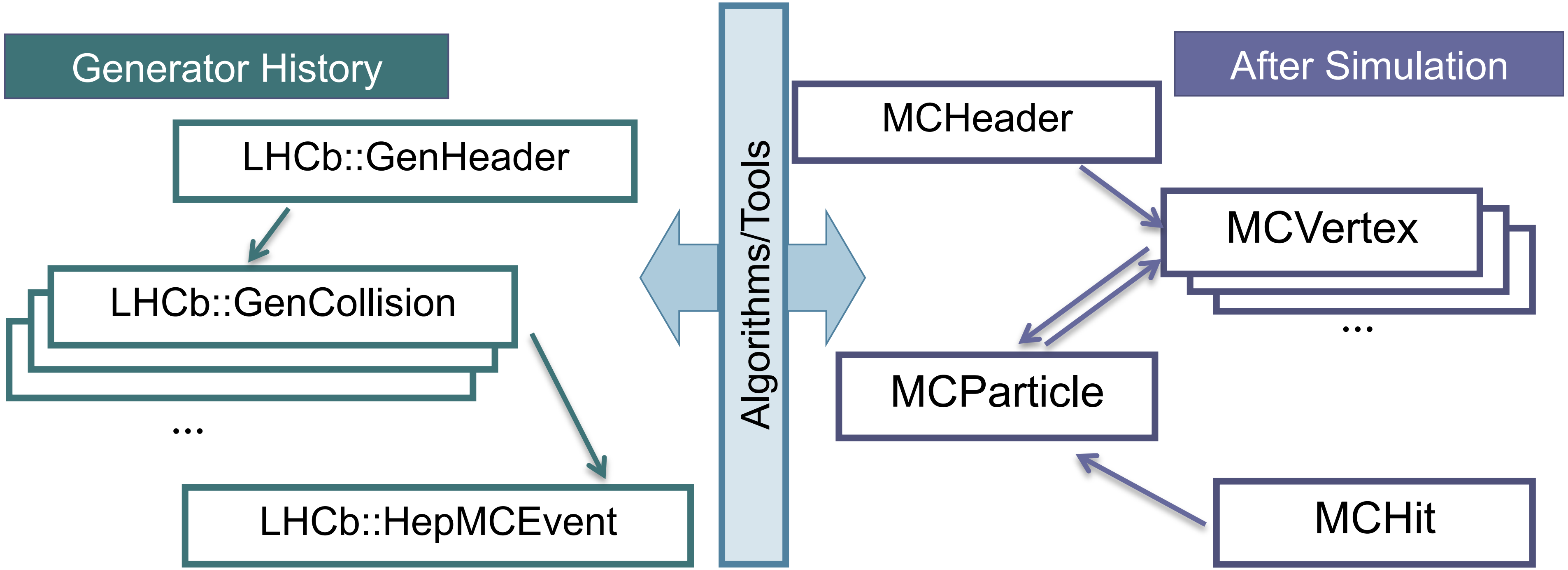
different implementations for various tools:

- large amount of common code
- easier addition of new generator packages (p-p interactions by Pythia or
- HERWIG)
- better flexibility to introduce new ideas, for generator level cuts for example

details about Primary Events Generation in Monday Poster Session (PO-MON-091)

Event model

In 2005 complete revision of the LHCb Event Model



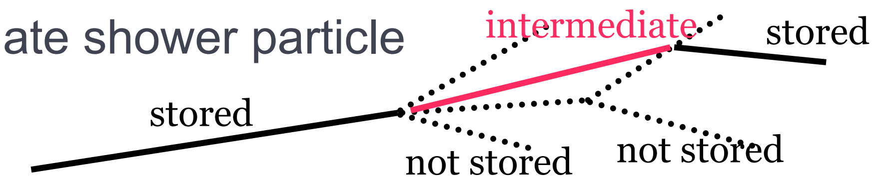
- All classes inherit from LHCb DataObject and containers (key/contained object)
- LHCb::HepMCEvent is the LHCb wrapper around HepMC

MC Event History

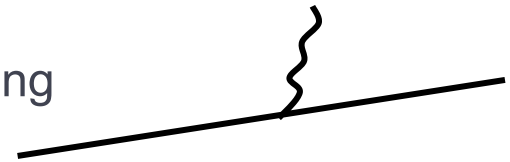
MCHistory (i.e. what happened during the tracking of particles) is essential to understand efficiencies and physics effects

- LHCb event model MCHistory (i.e. MCParticle, MCVertex and their relation) filled at the end of the Geant4 event processing (not optimal). Need to find a way to have a clear picture.
  - ✓ Geant4 does not have a tree structure to keep history (the only way to interact with tracks in G4 when a process occurs is in StepAction, unfeasible)
  - ✓ introduced use of HepMC internally to Geant4 to provide such a tree structure
  - ✓ we access a G4track to decide to "keep it" either when it is created to track it or when Geant stops tracking it.
  - ✓ can decide a-priori what to store through job options (e.g. skip the optical photons, keep all products of decays in detector,…)

- introduced intermediate shower particle



- dealing with particles emitted by particles not disappearing
  - ✓ HepMC::GenParticle can only have one end vertex
  - ✓ can decide in algorithm transferring to MCParticles if to keep the split or not

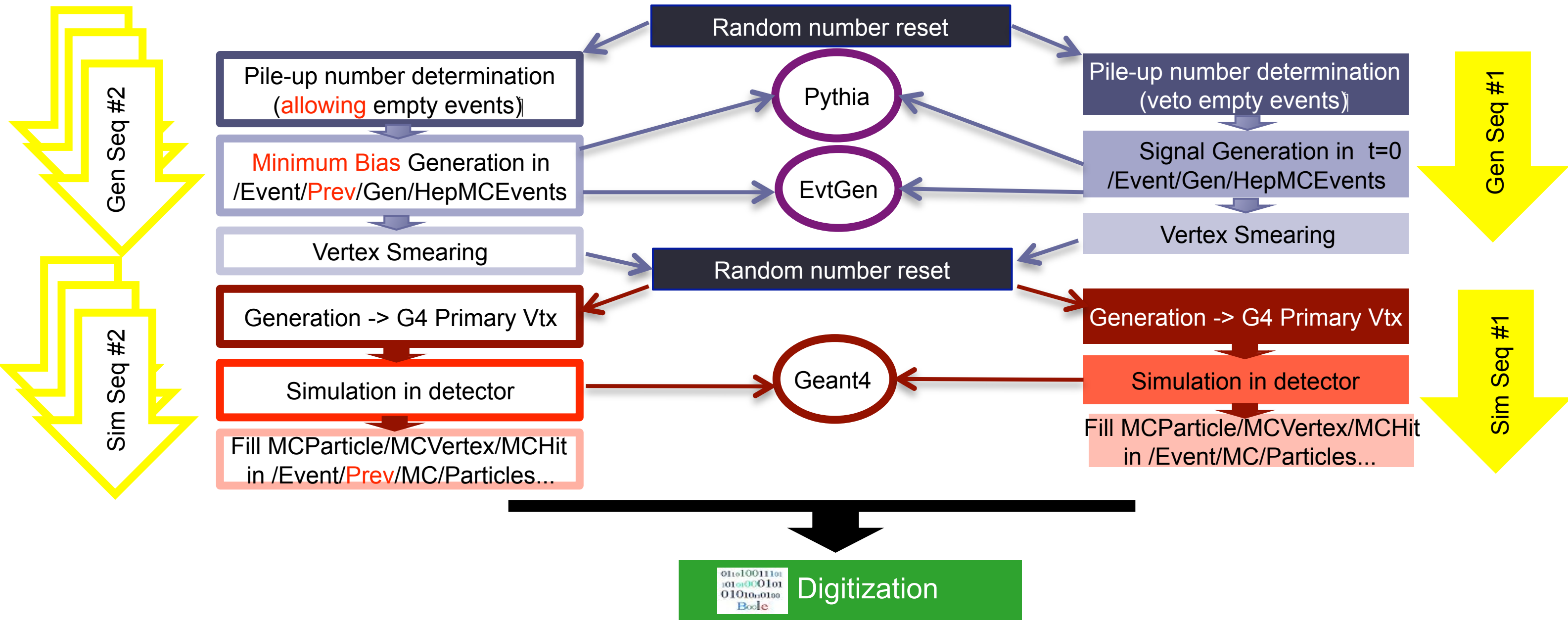


- kept the ProcessType info

SpillOvers (SO) in Gauss

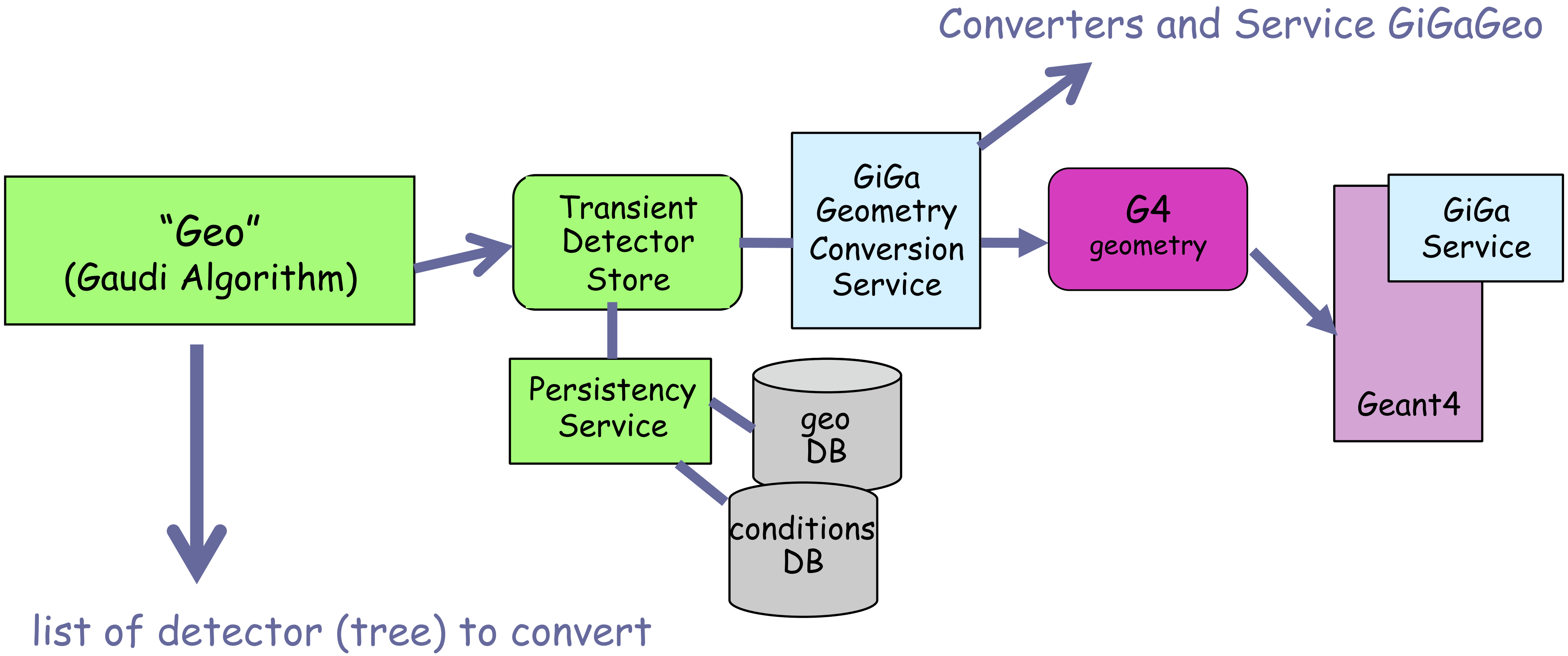
In 2009 the treatment of SO changed:

- previously treated in digitization application (Boole) using different input files (signal input file from Gauss and a minimum bias file merged)
- modified in order to generate spillover events in Gauss in a single file and a single job
- there is a single instance of Pythia, EvtGen, and Geant4 handling main event and SO events
- not a problem for Geant4 and EvtGen (decays): do not need to reconfigure the SO events



LHCb geometry in the simulation

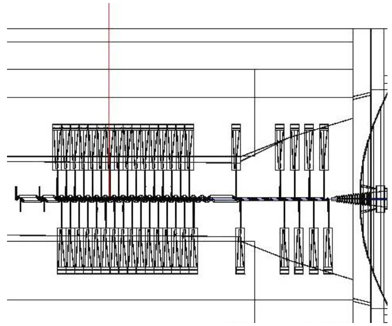
Gauss converts the LHCb geometry to the Geant4 description:




```
Geo.StreamItems += {"/dd/Structure/LHCb/BeforeMagnetRegion/Velo"};
Geo.StreamItems += {"/dd/Structure/LHCb/BeforeMagnetRegion/Rich1"};
Geo.StreamItems += {"/dd/Geometry/BeforeMagnetRegion/Rich1/Rich1Surfaces"};
...
```

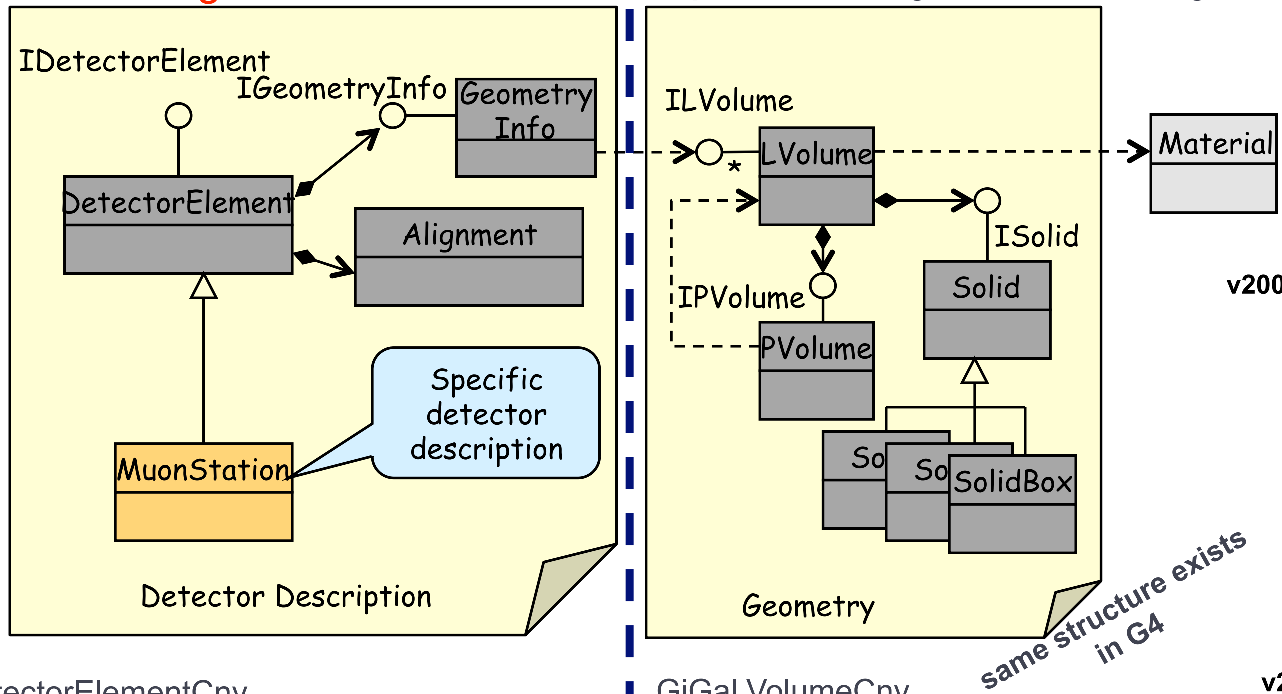
Geometry in the Simulation

v200601-450GeV-CloseVelo-BOff.opts

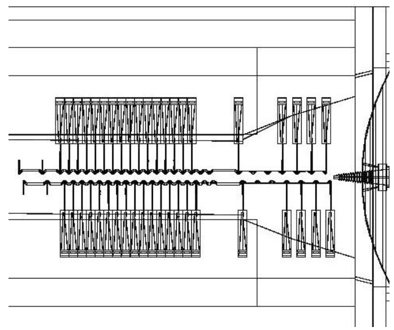


info about misalignment

position as given in LHCb geo DB



v200601-450GeV-SemiOpenVelo-BOff.opts



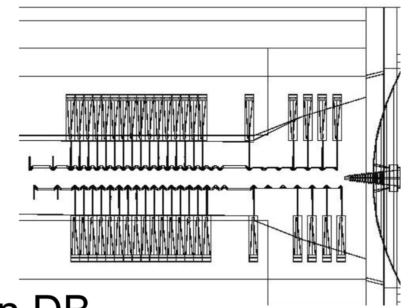
GiGaDetectorElementCnv

only called when given as StreamItems

GiGaLVolumeCnv

convert to G4 all its geometry tree

v200601-450GeV-OpenVelo-BOff.opts



Limitation:

- Pass all detectors (elements) to mis-align in options of Geo.StreamItems
- Pass all detectors as same level to what to misalign
- Cannot apply mis-alignment to parent and children if information is in condition DB

Solution (work in progress): re-engineering the conversion to G4 geo to take correctly into account the Detector Element structure

Gauss Monitoring

When adopting a new version of Gauss, a complete set of tests is performed to ensure the quality of the simulation:

- nightly build tests (QMTests)
- data quality
- geometry validation (overlaps checks, material budget, hit multiplicities,…)
- physics validation
- specific generator tests (generation efficiencies,…)

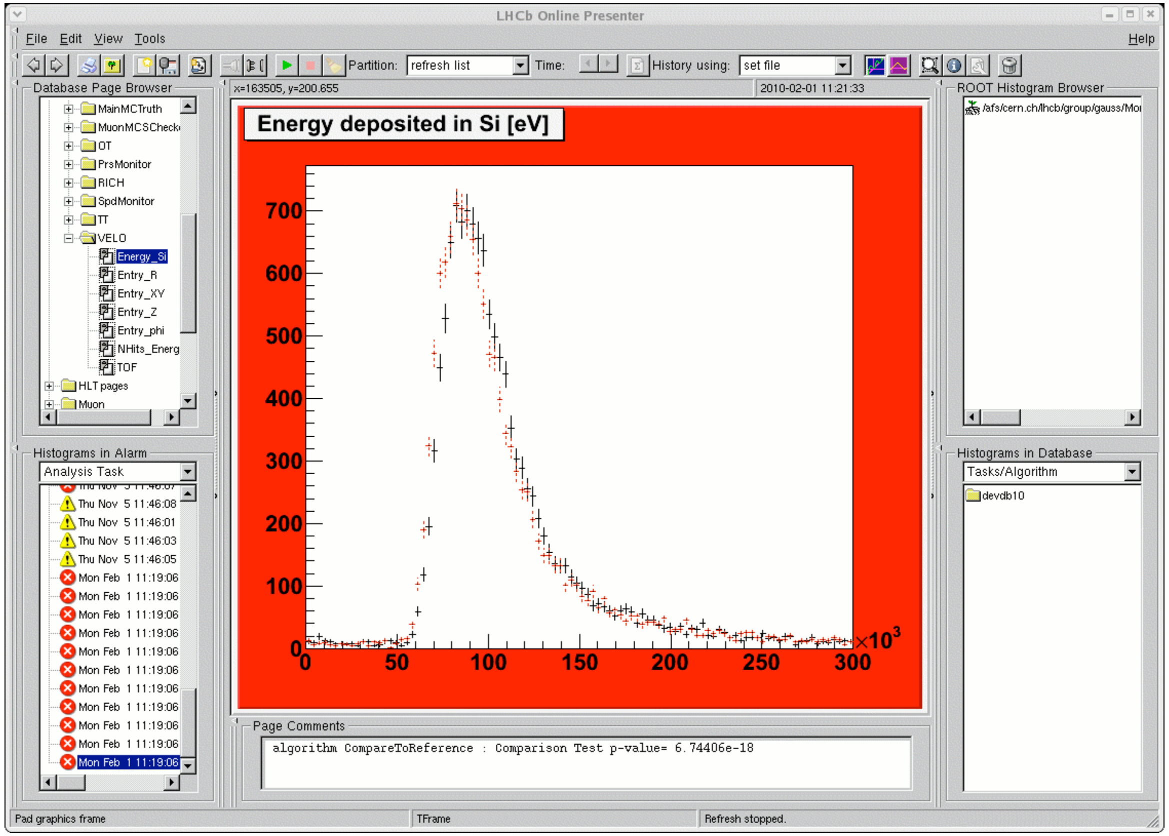
Gauss automatic software testing

- New releases are built with about ~1/month frequency, plus patches for production when needed
- upcoming releases and development versions are tested in the LHCb nightly system
- set of Run Time tests (~10 QMTests) for specific simulation benchmark samples
    - fast debug of detector or physics problems
    - generator-only (signal events, different generators samples) and full chain tests (minbias)
    - run both development and standard configurations

Project	Version	slc4_amd64_gcc34 (Thu Oct 7 06:58 2010)		x86_64-slc5-gcc43-opt (Thu Oct 7 04:29 2010)		slc4_ia32_gcc34 (Thu Oct 7 06:36 2010)		slc4_amd64_gcc34_dbg (Thu Oct 7 08:12 2010)		slc4_ia32_gcc34_dbg (Thu Oct 7 09:47 2010)		x86_64-slc5-gcc43-dbg (Thu Oct 7 04:50 2010)	
Geant4	GEANT4_v92r3p2	build (63)	tests	build (1)	tests	build (63)	tests	build (62)	tests	build (62)	tests	build	tests
Gauss	GAUSS_HEAD	build (21)	tests (11)	build (95)	tests (11)	build (21)	tests (11)	build (21)		build (21)		build (98)	tests (9)

Data quality

Integrating the complete set of Online Tools to monitor the Gauss output histograms:

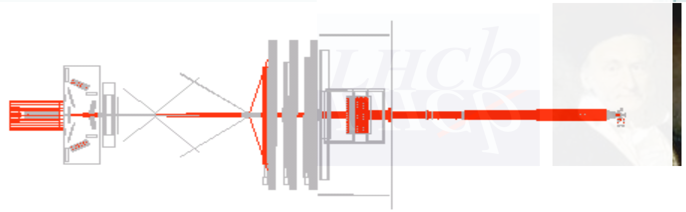


- OnlineHistogramDB (sql-based) storing the display settings and configurations
- Online Presenter (GUI to display ROOT-based histos)
- Tools for Automatic Analysis can be run on the MC histos

Set of Python scripts runs on logs and histos to produce statistic tables (generators efficiencies, comparison plots between versions…) updating html pages.

Geometry Validation

Every time the XML geometry description changes…

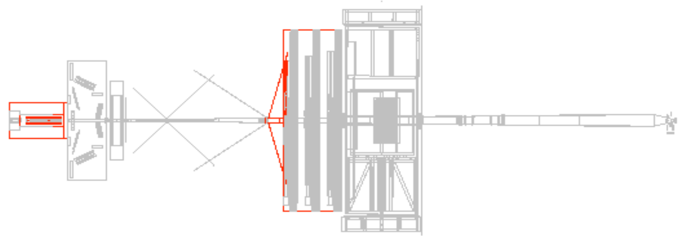


Overlaps check

Geant4 does NOT like volumes overlaps:

- Particle get stuck and event is (not) aborted
- Possible crashes

Must ensure when all mis-alignments are taken into account no overlaps are present

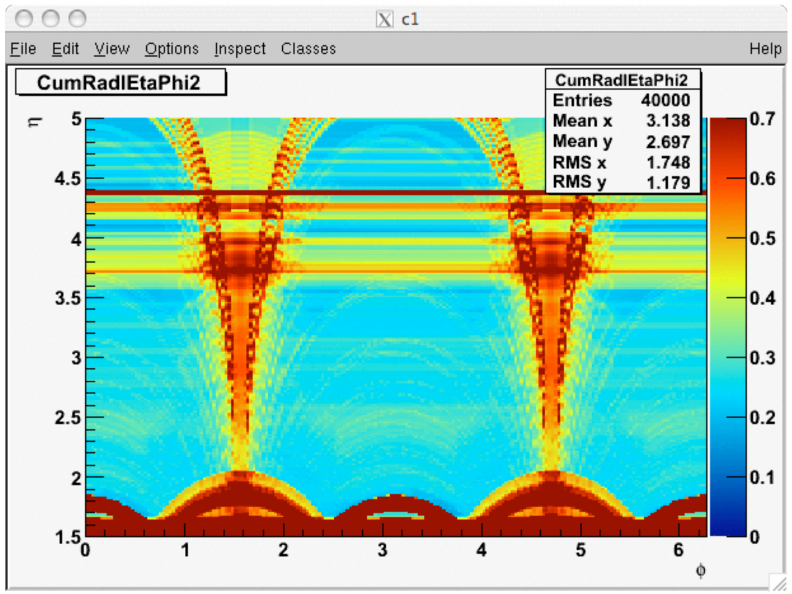


- Geant4 DAVID visualization tool was used to detect the overlaps between the volumes and converted to a graphical representation for visualization purposes (95% overlaps turned out to be due to precision problems)

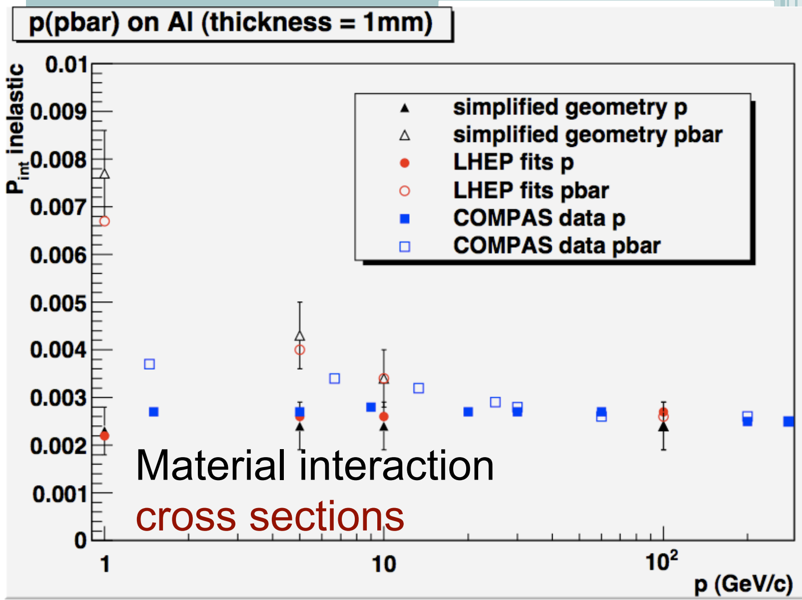
Material Budget evaluation

Radiation length evaluations performed periodically to compare the updates in the detector description:

- amount of material as seen by the particles at Geant4Step-level
- comparison with material as seen by the LHCb detector description



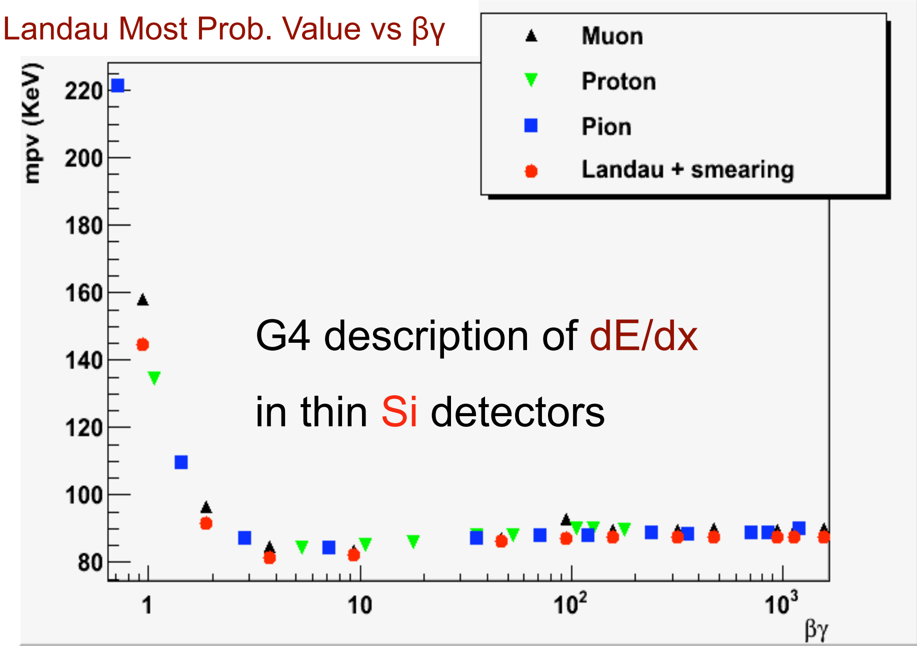
Geant4 Physics Validation

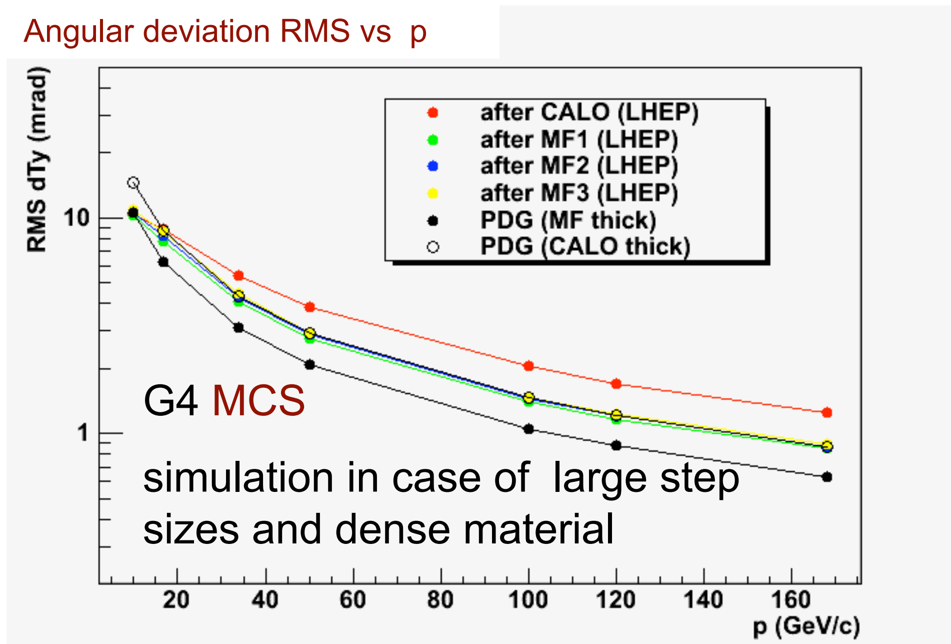


Every time a Geant4 version is changed:

- main physics quantities are tested
- process-related simulation issues are kept under control

Validation done with Geant4.9.2.p03
(PLs: LHEP, QGSP_BERT, FTFP_BERT).

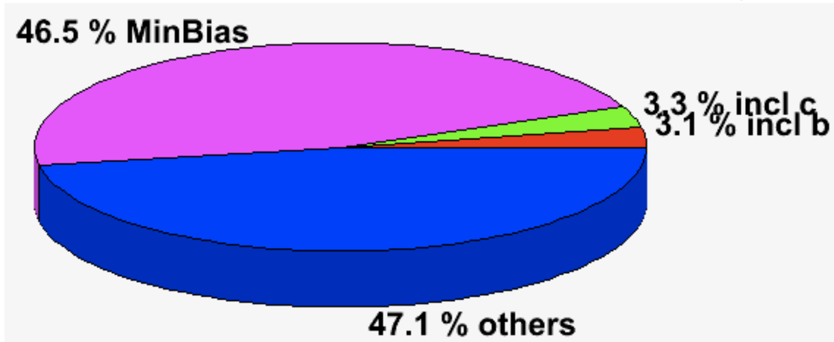




Gauss in Production

	Total Events	Event Types	Total file size
DC06	598M	155	129TB
MC09	2791M	165	198TB
2010	663M	165	99TB

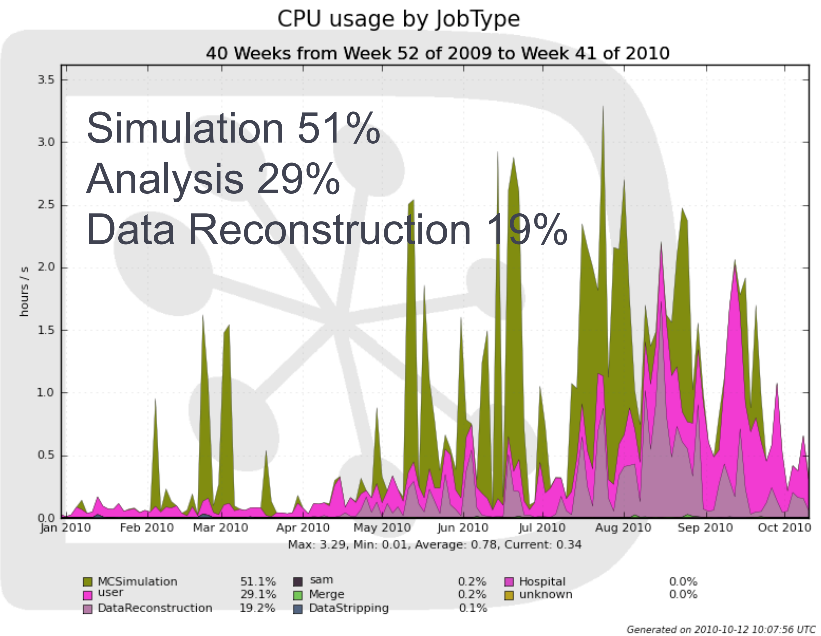
2010 Event Types

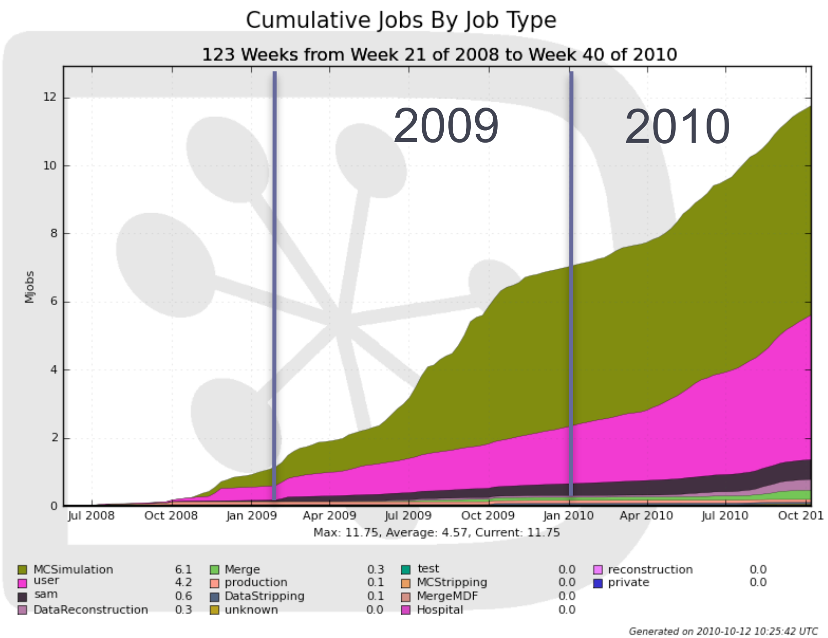


CPU time on slc5 dedicated machine (2.8 GHz Xeon)

- MinBias 0.47 min/evt
- inclusive b 1.26 min/evt
- Bs->mumu 1.32 min/evt

Stable memory consumption (~1.2 GB on slc5 64)





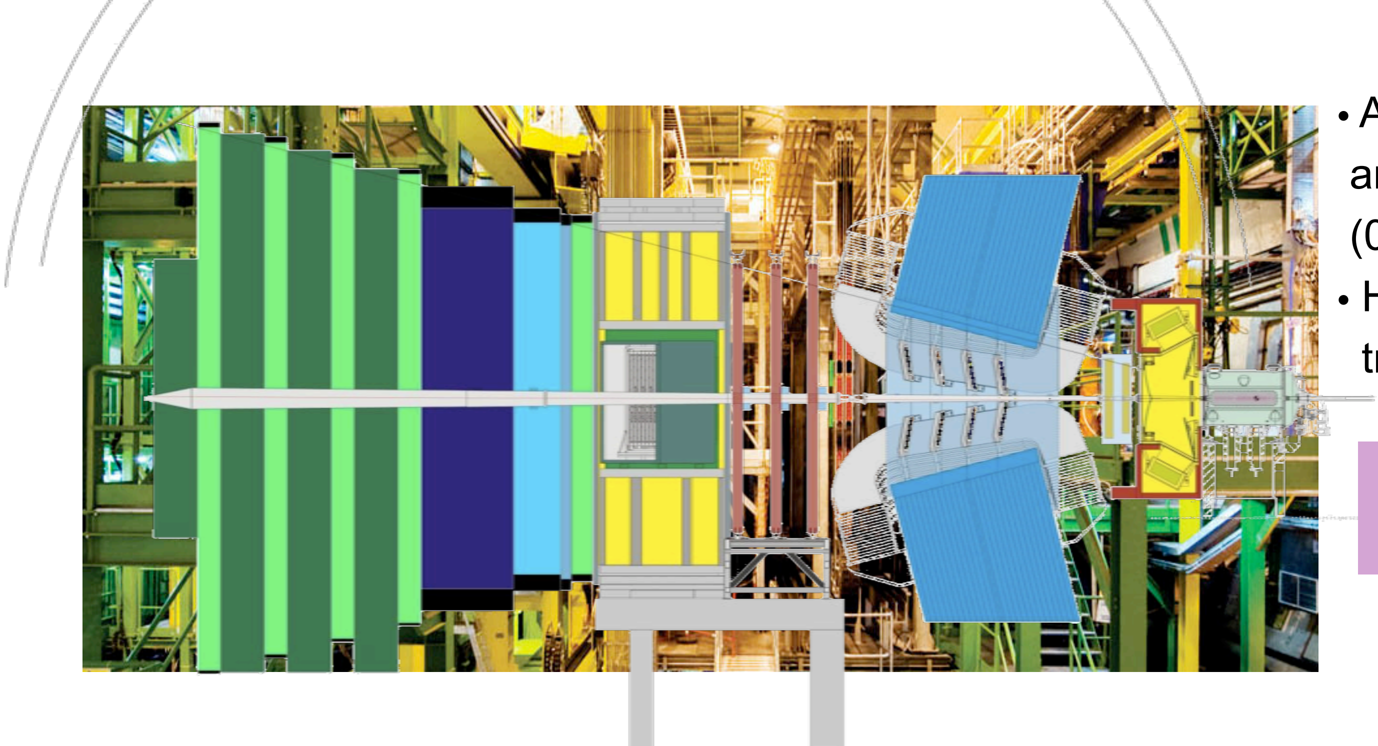
Summary

- Gauss, the LHCb simulation package, is a Gaudi-based application
- used in production and by users for their studies since 2004
- it has undergone several evolution steps in the last years to cope with crucial requirements as the handling of the MCHistory and the treatment of the SpillOvers
- MC productions on GRID allowed to successfully fulfill the needs and requirements of the LHCb Users
- different monitoring tools have been developed in order to guarantee the simulation quality

Backup

LHCb

- Designed to make precision measurement of CP violation and other rare phenomena in the b system at the LHC
- Trigger and reconstruct many different B decay modes to make independent and complementary measurements
- LHCb is a single arm forward spectrometer
- Forward production of bb, correlated



- Amount of material in tracker area kept as low as possible ($0.6X_0$ up to RICH2)
- HCAL used mainly for trigger purpose

12 mrad < θ < 300 (250) mrad

i.e. 2.0 < η < 4.9