CEPC-on-Gaussino prototype: an application of Gaussino simulation framework for CEPC experiment

Tao Lin (IHEP)

lintao@ihep.ac.cn

ICHEP 2024

20 July 2024

Outline

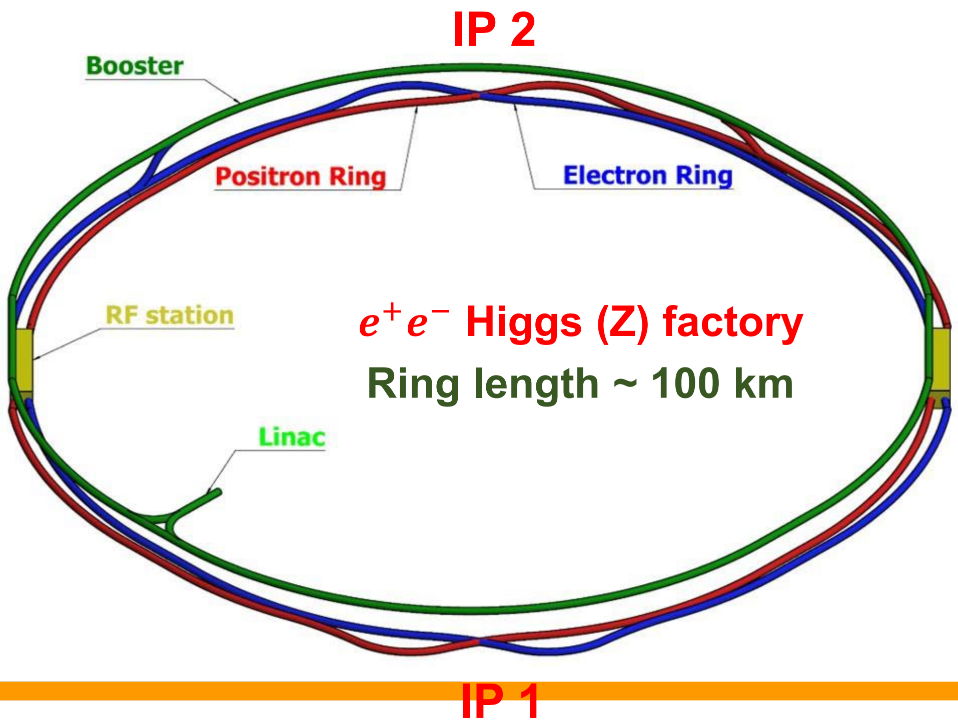
- Introduction
- Moving to a next simulation framework
- CEPC-on-Gaussino prototype
- Summary

Outline

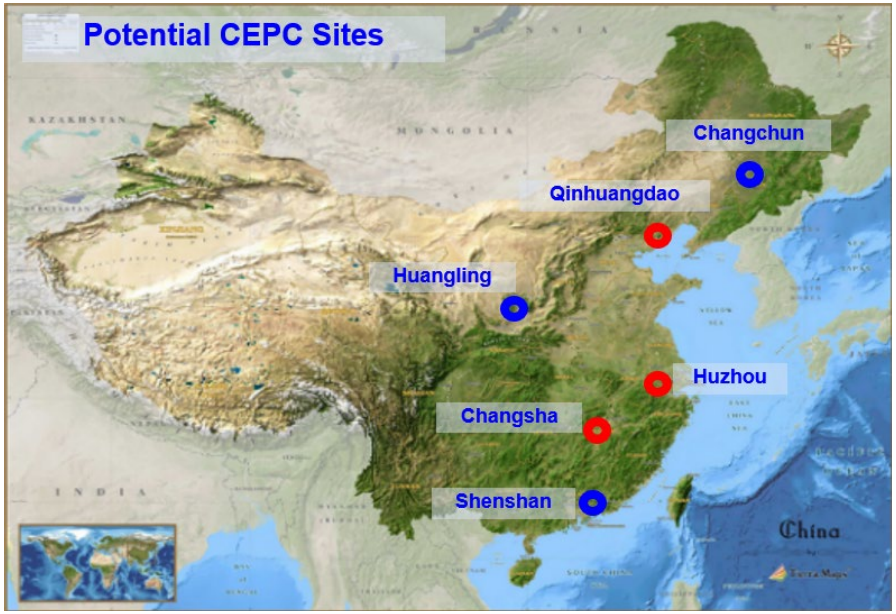
- Introduction
  - CEPC project and the status
  - CEPCSW and the current simulation framework
- Moving to a next simulation framework
- CEPC-on-Gaussino prototype
- Summary

CEPC: Circular Electron Positron Collider

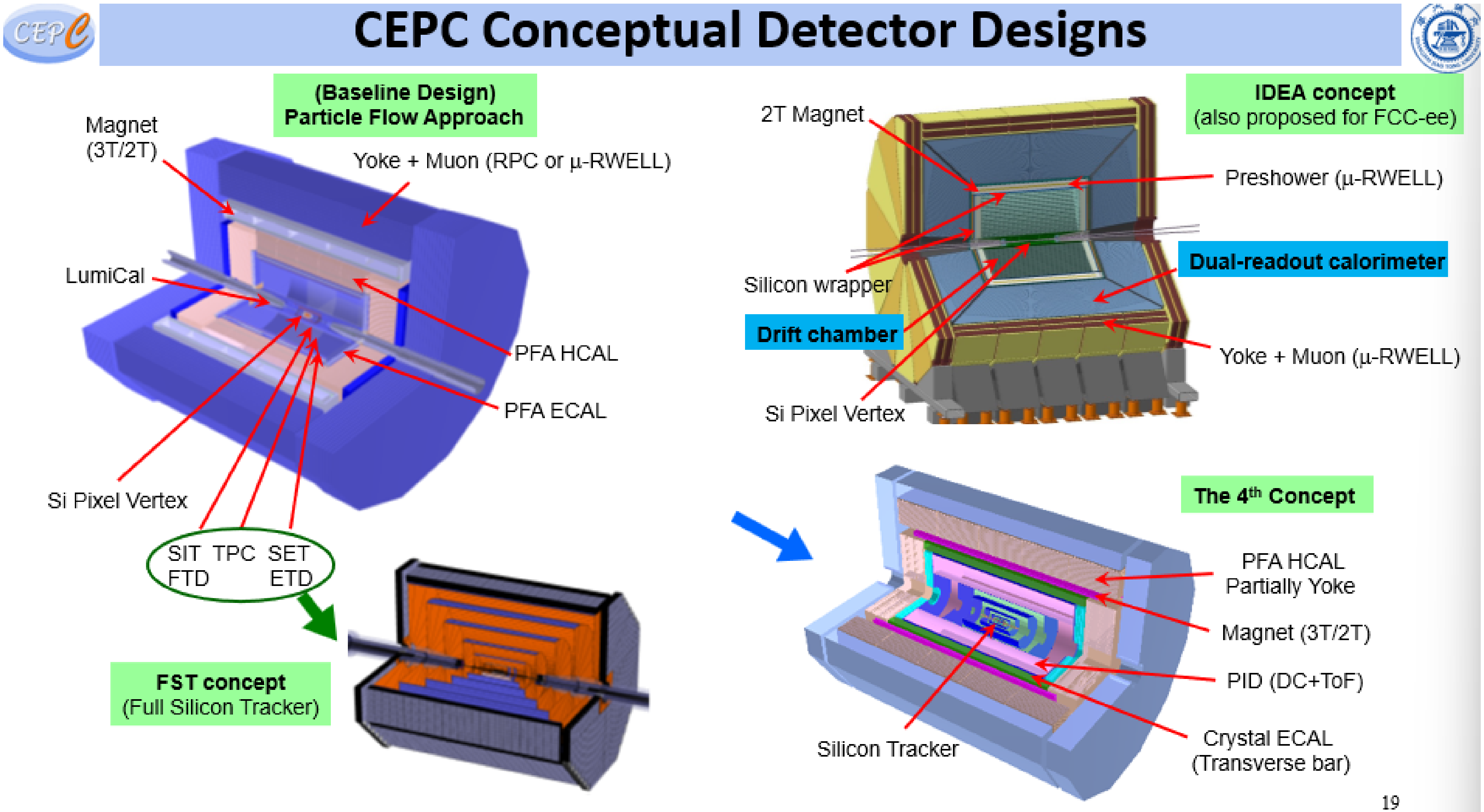
- CEPC is an e+e- Higgs factory producing Higgs/W/Z bosons and top quarks, aims at discovering new physics beyond the Standard Model.
- Milestones:
  - 2018.11: CEPC CDR released
  - 2023.12: CEPC Accelerator TDR released



Haijun Yang | Overview of the CEPC Project
CEPC International Workshop at Marseille

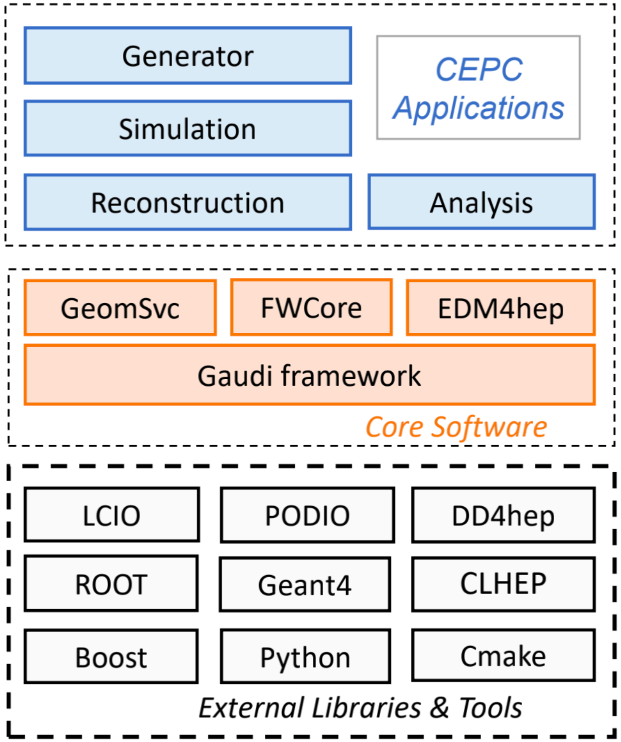


Towards Reference Detector TDR



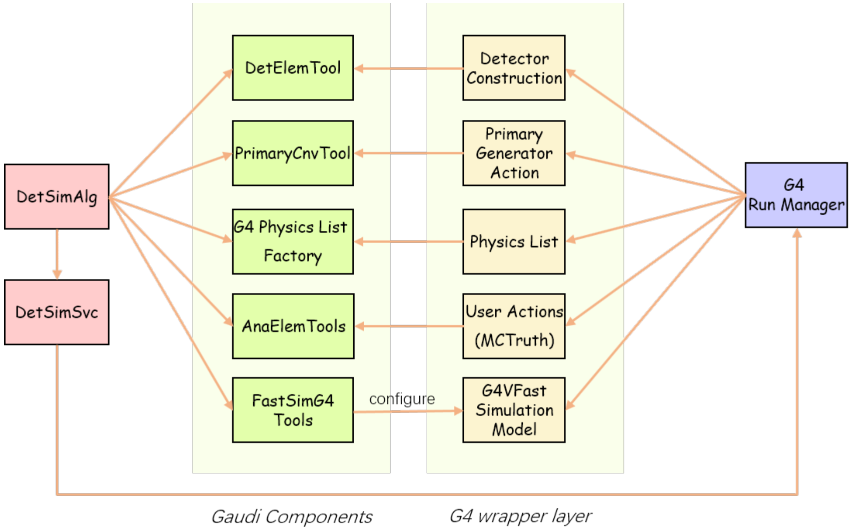
CEPCSW: software based on Key4hep

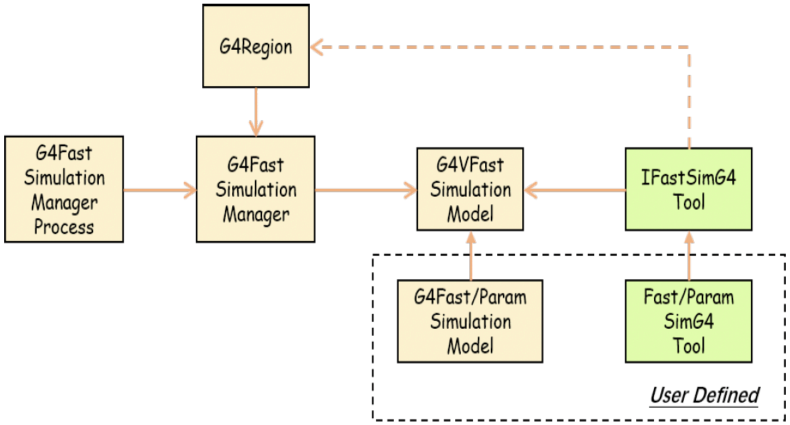
- CEPCSW is developed based on a **Common Turnkey Software Stack (Key4hep)**
  - The consensus among CEPC, CLIC, FCC, ILC and other future experiments was reached at the Bologna workshop in June 2019
  - Maximize the sharing of software components among different experiments.
- **Core software**
  - Gaudi (Hive): defines interfaces to all software components and controls their execution
  - EDM4hep: generic event data model
  - K4FWCore: manages the event data
  - DD4hep: geometry description
  - CEPC-specific framework software: generator, Geant4 simulation, beam background mixing, fast simulation, machine learning interface, etc.
- CEPCSW source code:
  - https://code.ihep.ac.cn/cepc/CEPCSW



Simulation framework in CEPCSW

- Full detector simulation has been developed based on Geant4.
  - A unified simulation framework to integrate Geant4 and Gaudi.
  - Detector description: convert from DD4hep geometry using DDG4.
  - Event data: SimTrackerHit and SimCalorimeterHit
  - Generator interfaces: HepMC, LCIO, StdHep, Beam background, Particle Gun
  - Detector responses: trackers, DC, TPC, calorimeter
  - Monte Carlo truth: association between hits and MC particles
  - Fast simulation interfaces: Geant4 Region based.



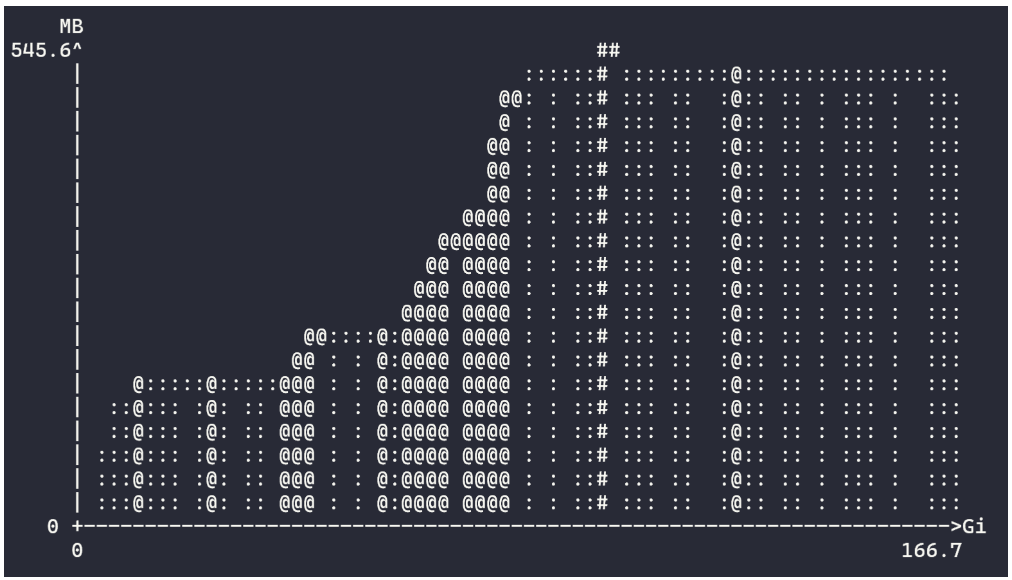


Outline

- Introduction
- Moving to a next simulation framework
  - Motivation
  - Gaussino project from LHCb
- CEPC-on-Gaussino prototype
- Summary

Moving to a next simulation framework (1)

- The current simulation framework in CEPCSW is not benefit from the multithreading because it does not support the Geant4-MT.
  - Sharing the geometries and physics lists could reduce the memory usage.



Command line:

```
gaudirun.py --profilerName=valgrindmassif sim.py
```

Memory usage (serial version)

Simulation setup:

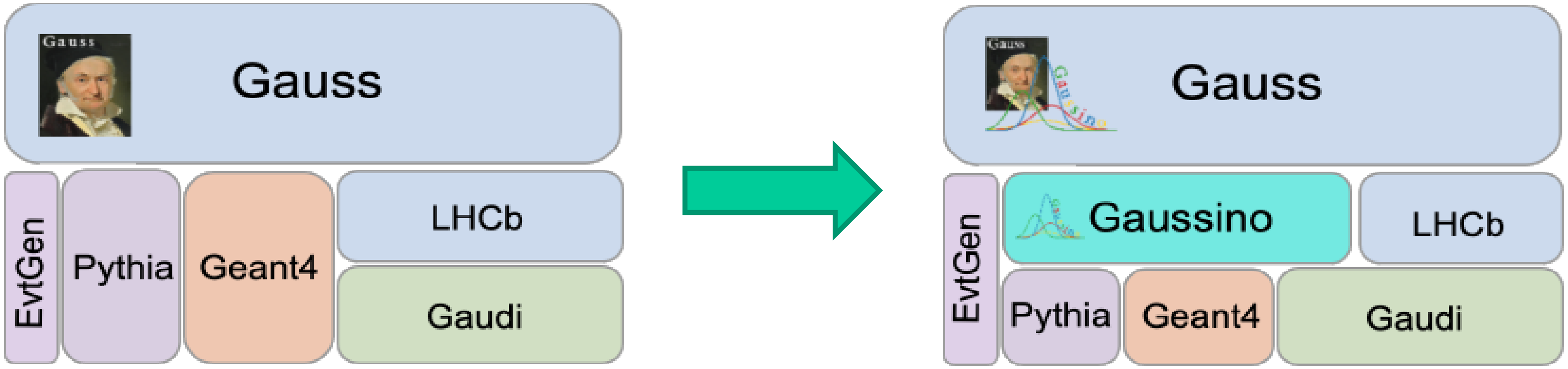
- Detector: TDR_o1_v01
- Physics list: QGSP_BERT
- Generation: single muons
- N events: 100

The RSS memory is about **950MB at initialization stage**.

The figure shows the **heap memory usage**.

Moving to a next simulation framework (2)

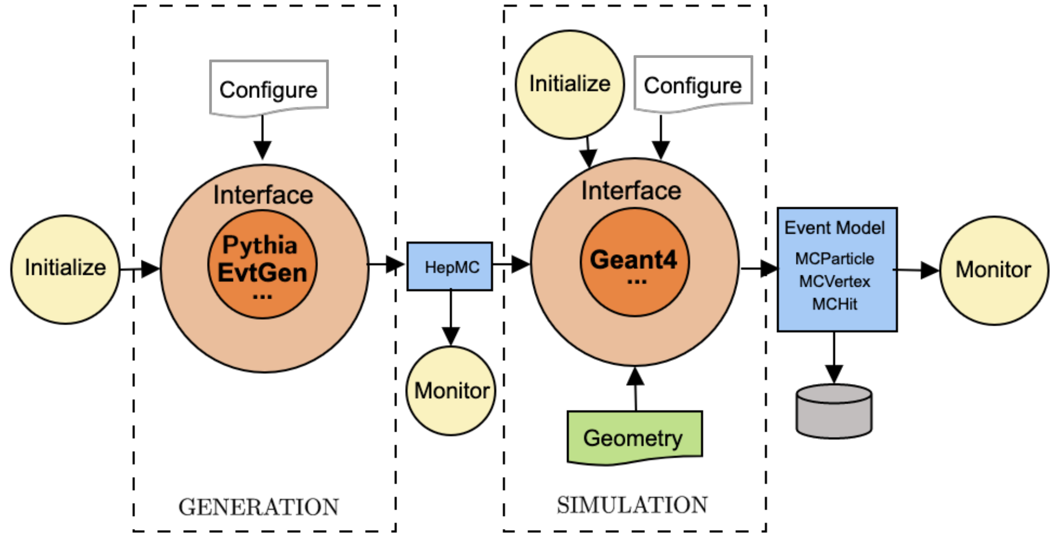
- Question: **Develop a new one** or **adopt an existing one**?
  - Gaussino is a potential solution in Key4hep. [arXiv: 2312.08152]
- Gauss-on-Gaussino: Evolution of the simulation framework from LHCb
  - The underlying framework is moving to Gaudi Functional and Gaudi Hive
  - Better support for multi-threading, machine learning, fast simulation methods
  - Gauss-on-Gaussino is a new version of LHCb simulation framework



[1] Michał Mazurek, CHEP 2023, The LHCb simulation software Gauss and its Gaussino core framework
[2] Michał Mazurek, CHEP 2023, From prototypes to large scale detectors: how to exploit the Gaussino simulation framework for detectors studies, with a detour into machine learning

Overview of Gaussino (1)

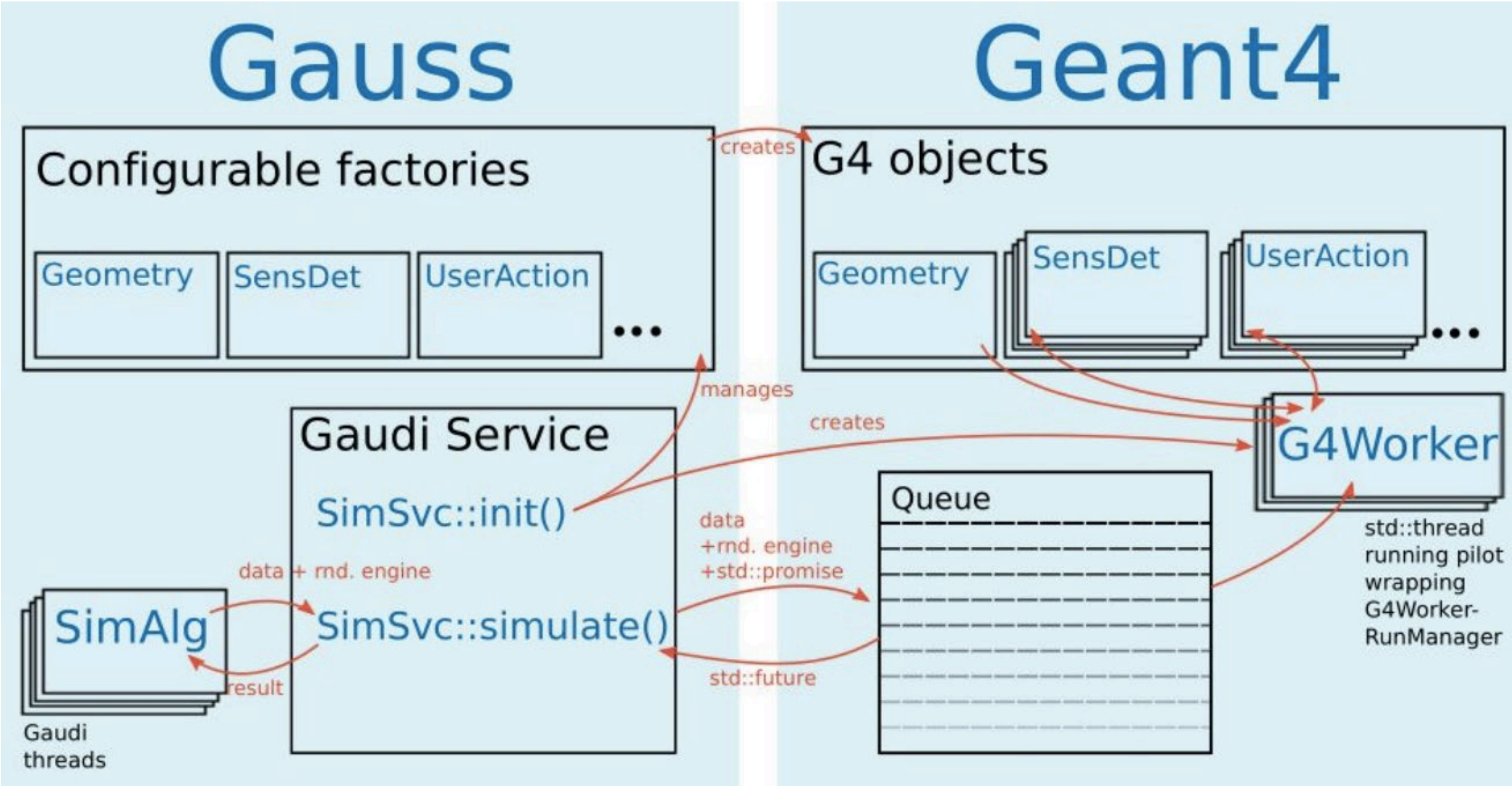
- Gaussino is a thread-safe simulation framework based on Gaudi Functional and provides interfaces to Pythia and Geant4.
- Modular design
  - Gaudi Functional Algorithms
  - Gaudi Tools
- Four components
  - Generation of events
  - The detector simulation
  - Geometry service
  - Monitoring & output
- Easy to configure by customizing the algorithms, services and tools



- Generation: `Generation` and `ParticleGun`
  - The input is **LHCb GenHeader**
  - The output is **HepMC GenEvent**
- Detector simulation: `GiGaAlg`
  - The input is **HepMC GenEvent**
  - The output is **G4Event** and **MC truths**

Overview of Gaussino (2)

- Multithreading with Geant4
  - A queue is used to communicate between Gaussino and Geant4



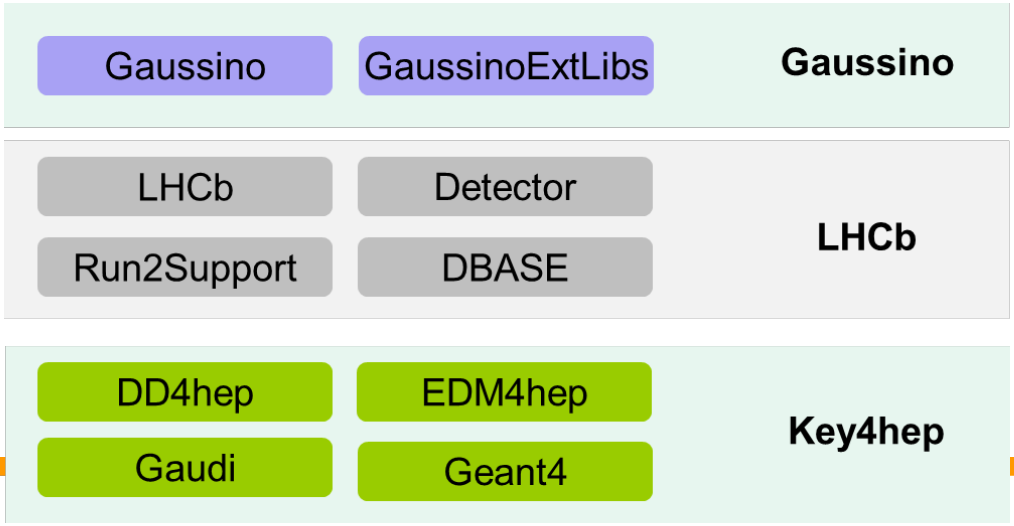
Sim/GiGaMTCore/include/GiGaMTCoreRun/GiGaWorkerPilot.h

Outline

- Introduction
- Moving to a next simulation framework
- CEPC-on-Gaussino prototype
  - Design
  - Implementation
- Summary

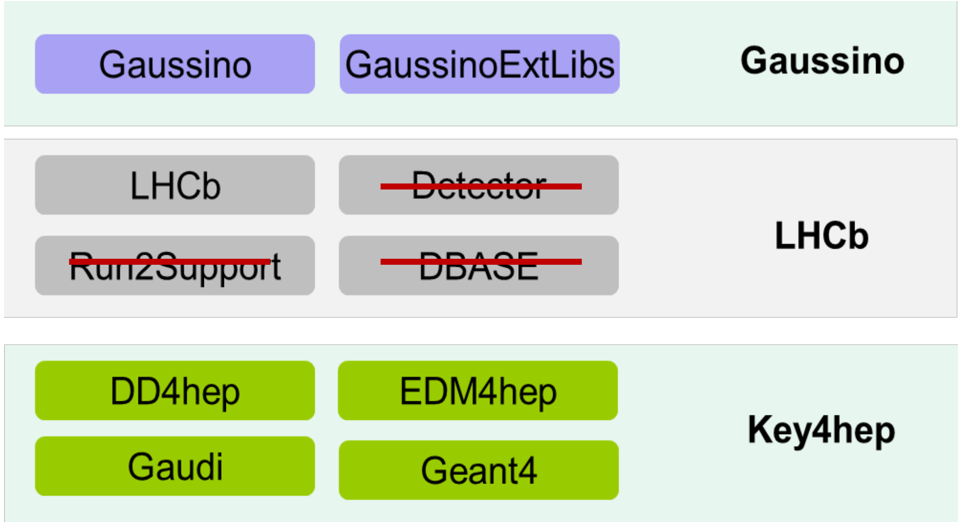
CEPC-on-Gaussino prototype (1)

- Gaussino still depends on LHCb software and removing the dependencies is on going
  - Some existing works: MR !1493 in Gaudi by Graeme.
- Development of CEPC-on-Gaussino was planned with the following three steps
  - ✓ **Using the original version having the dependency on the LHCb software**
  - ✓ **Creating a modified version with less LHCb dependency**
  - ☐ Directly using the Key4hep version without LHCb dependency (not available at the moment)



CEPC-on-Gaussino prototype (2)

- It will be too heavy to build the prototype with the whole LHCb software.
- CMakeLists.txt in LHCb and gaussinoextlibs are modified to build necessary libraries.
- After optimization, about 20 libraries are built in LHCb and GaussinoExtLibs.

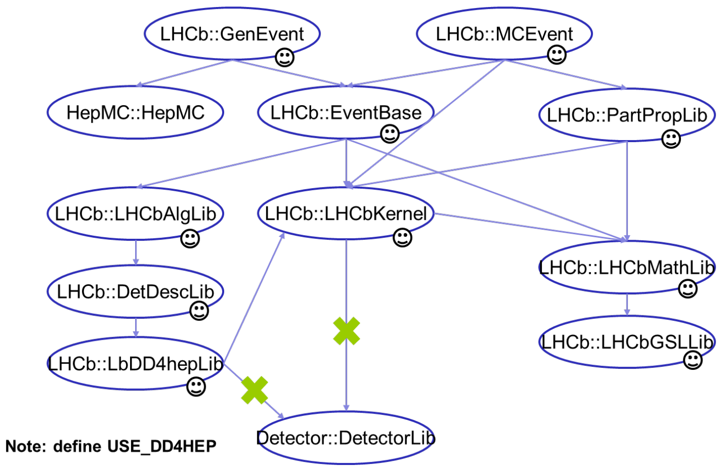



```
[lint@lxslc705]$ ls -l LHCb/InstallArea/lib/
total 24712
drwxr-xr-x 3 lint physics      18 Dec 12 03:54 cmake
-rw-r--r-- 1 lint physics   37218 Dec 12 03:54 DAQEventDict_rdict.pcm
-rw-r--r-- 1 lint physics    3053 Dec 12 03:54 EventBaseDict_rdict.pcm
-rw-r--r-- 1 lint physics    7512 Dec 12 03:54 GaudiGSLMathDict_rdict.pc
-rw-r--r-- 1 lint physics    8364 Dec 12 03:54 GenEventDict_rdict.pcm
-rw-r--r-- 1 lint physics     370 Dec 12 03:54 LHCb.components
-rw-r--r-- 1 lint physics     751 Dec 12 03:54 LHCb.confdb
-rw-r--r-- 1 lint physics   32768 Dec 12 03:54 LHCb.confdb2
-rw-r--r-- 1 lint physics   22277 Dec 12 03:54 LHCbDict.rootmap
-rw-r--r-- 1 lint physics   28815 Dec 12 03:54 LHCbMathDict_rdict.pcm
-rwxr-xr-x 1 lint physics  424208 Dec 12 03:54 libDAQEventDict.so
-rwxr-xr-x 1 lint physics 3229632 Dec 12 03:54 libDAQEventLib.so
-rwxr-xr-x 1 lint physics  236840 Dec 12 03:54 libEventBaseDict.so
-rwxr-xr-x 1 lint physics 1171648 Dec 12 03:54 libGaudiGSLLib.so
-rwxr-xr-x 1 lint physics  269464 Dec 12 03:54 libGaudiGSLMathDict.so
-rwxr-xr-x 1 lint physics 1836888 Dec 12 03:54 libGaudiGSL.so
-rwxr-xr-x 1 lint physics 1221432 Dec 12 03:54 libGenEventDict.so
-rwxr-xr-x 1 lint physics  442888 Dec 12 03:54 libGenEvent.so
-rwxr-xr-x 1 lint physics 3061496 Dec 12 03:54 libLHCbKernel.so
-rwxr-xr-x 1 lint physics 2534464 Dec 12 03:54 libLHCbMathDict.so
-rwxr-xr-x 1 lint physics 6642152 Dec 12 03:54 libLHCbMathLib.so
-rwxr-xr-x 1 lint physics  363000 Dec 12 03:54 libMCEvent.so
-rwxr-xr-x 1 lint physics  509928 Dec 12 03:54 libPartPropDict.so
-rwxr-xr-x 1 lint physics 1107720 Dec 12 03:54 libPartPropLib.so
-rwxr-xr-x 1 lint physics 2050024 Dec 12 03:54 libPartProp.so
-rw-r--r-- 1 lint physics    7934 Dec 12 03:54 PartPropDict_rdict.pcm
```

```
[lint@lxslc705]$ ls -l gaussinoextlibs/InstallArea/lib/
total 46028
drwxr-xr-x 3 lint physics       29 Dec 12 03:54 cmake
-rw-r--r-- 1 lint physics     2940 Dec 12 03:54 G__DDG4Python_rdict.pcm
-rw-r--r-- 1 lint physics    23520 Dec 12 03:54 G__DDG4_rdict.pcm
-rw-r--r-- 1 lint physics     1763 Dec 12 03:54 G__DDPython_rdict.pcm
-rw-r--r-- 1 lint physics      208 Dec 12 03:54 libDDG4LCIO.components
lrwxrwxrwx 1 lint physics       19 Oct 24 11:51 libDDG4LCIO.so -> libDDG4LCIO.so.1.25
-rwxr-xr-x 1 lint physics  3452704 Dec 12 03:54 libDDG4LCIO.so.1.25
-rw-r--r-- 1 lint physics    16549 Dec 12 03:54 libDDG4Plugins.components
lrwxrwxrwx 1 lint physics       22 Oct 24 11:51 libDDG4Plugins.so -> libDDG4Plugins.so.1.25
-rwxr-xr-x 1 lint physics 15688056 Dec 12 03:54 libDDG4Plugins.so.1.25
-rw-r--r-- 1 lint physics      220 Dec 12 03:54 libDDG4Python.components
lrwxrwxrwx 1 lint physics       21 Oct 24 11:51 libDDG4Python.so -> libDDG4Python.so.1.25
-rwxr-xr-x 1 lint physics   818528 Dec 12 03:54 libDDG4Python.so.1.25
lrwxrwxrwx 1 lint physics       15 Oct 24 11:51 libDDG4.so -> libDDG4.so.1.25
-rwxr-xr-x 1 lint physics 23030168 Dec 12 03:54 libDDG4.so.1.25
lrwxrwxrwx 1 lint physics       19 Oct 24 11:51 libDDPython.so -> libDDPython.so.1.25
-rwxr-xr-x 1 lint physics   422488 Dec 12 03:54 libDDPython.so.1.25
lrwxrwxrwx 1 lint physics       20 Oct 24 11:49 libHepMC3search.so -> libHepMC3search.so.4
-rwxr-xr-x 1 lint physics   246384 Dec 12 03:54 libHepMC3search.so.4
-rw-r--r-- 1 lint physics   363832 Dec 12 03:54 libHepMC3search-static.a
lrwxrwxrwx 1 lint physics       14 Oct 24 11:49 libHepMC3.so -> libHepMC3.so.3
-rwxr-xr-x 1 lint physics   923296 Dec 12 03:54 libHepMC3.so.3
-rw-r--r-- 1 lint physics  2108684 Dec 12 03:54 libHepMC3-static.a
drwxr-xr-x 3 lint physics       27 Dec 12 03:54 python3.9
```

CEPC-on-Gaussino prototype (3)

- **Event Data Model**
  - reuse GenEvent and MCEvent from the LHCb project
  - A minimum number of packages are selected
  - Non-required dependencies were removed
- **Detector description**
  - use the DD4hep, so that no dependent on LHCb detector description library
  - Use DD4hepCnvSvc as Geometry Service to load CEPC tracker detector.
- **Detector response:**
  - Implement G4 Sensitive Detector and Hit object for tracker detector.
  - Implement a monitor tool to save user output.




```
CEPCTest/
├── CMakeLists.txt
├── include
│   └── CEPCTest
│       ├── DDG4SensitiveDetector.h
│       ├── Geant4Hits.h
│       ├── GenericTrackerSensDetTool.h
│       └── GenericTrackerSensitiveDetector.h
├── options
│   └── cepc_gaussino.py
└── src
    ├── Components
    │   ├── GenericTrackerMonTool.cpp
    │   ├── GenericTrackerMonTool.hh
    │   └── GenericTrackerSensDetToolComponent.cpp
    └── Lib
        ├── DDG4SensitiveDetector.cpp
        ├── Geant4Hits.cpp
        ├── GenericTrackerSensDetTool.cpp
        └── GenericTrackerSensitiveDetector.cpp
```

CEPC-on-Gaussino prototype (4)

Job option

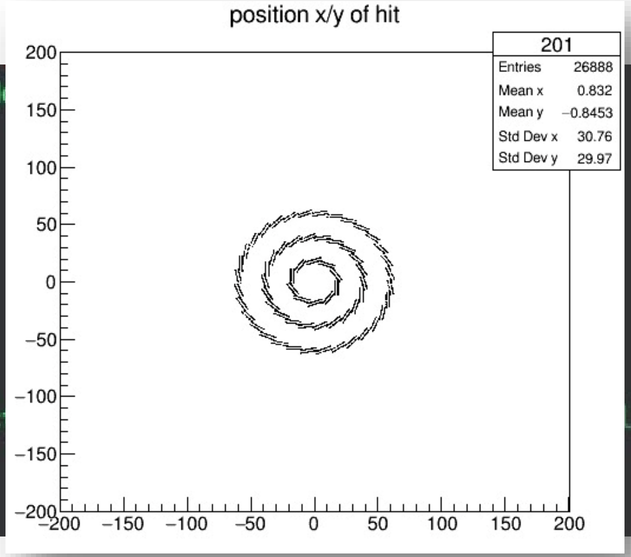
- CEPC-on-Gaussino is configured in the Python script.
- Register the sensitive detector factory and associate it with DD4hep detector name.
- Register the monitor tool for testing only.
- The output in EDM4hep is not implemented yet.

```
montool = giga.addTool(GenericTrackerMonTool, name="GenericTrackerM
montool.OutputLevel = DEBUG
montool.CollectionName = "VXDCollection"

giga.MonitorTools = ["GenericTrackerMonTool"]

GaussinoGeometry().GeometryService = "DD4hepCnvSvc"

DD4hepCnvSvc().DescriptionLocation = "CEPCSW/Detector/DetCRD/compac
DD4hepCnvSvc().SensDetMappings = {"VXD": "VXD"}
# DD4hepCnvSvc().OutputLevel = DEBUG
```



https://gitlab.cern.ch/talin/build-cepc-on-gaussino/-/blob/main/my_gaussino_dd4hep.py?ref_type=heads

CEPC-on-Gaussino prototype (5)

- The installation script is used to build LHCb, gaussinoextlibs, Gaussino, and CEPCSW with branch name cepc-on-gaussino.

```
$ git clone
ssh://git@gitlab.cern.ch:7999/talin/
build-cepc-on-gaussino.git
$ cd build-cepc-on-gaussino

$ bash build.sh build-all

$ source setup.sh

$ gaudirun.py my_gaussino_dd4hep.py
```

```
function env-setup() {
    # == LCG  ==
    lcg_version_lcg=LCG_103 # The version used by LCG self.
    lcg_platform=x86_64-centos7-gcc11-opt

    export PATH=/cvmfs/sft.cern.ch/lcg/contrib/git/bin:$PATH
    export LD_LIBRARY_PATH=/cvmfs/sft.cern.ch/lcg/contrib/git/lib64:$LD_LIBRARY_PATH

    source /cvmfs/sft.cern.ch/lcg/views/${lcg_version_lcg}/${lcg_platform}/setup.sh

    # == Gaussino and dependencies  ==

    export WORKTOP=$(pwd)

    for project in LHCb gaussinoextlibs Gaussino CEPCSW; do

        project_path=$WORKTOP/$project/InstallArea
        export CMAKE_PREFIX_PATH=$project_path:$CMAKE_PREFIX_PATH
        export PATH=$project_path/bin:$PATH
        export LD_LIBRARY_PATH=$project_path/lib:$LD_LIBRARY_PATH
        export PYTHONPATH=$project_path/lib:$PYTHONPATH
        export PYTHONPATH=$project_path/python:$PYTHONPATH

    done

    export GAUSSINOROOT=$WORKTOP/Gaussino/Sim/Gaussino
}
```

Project	Git repo
LHCb	https://gitlab.cern.ch/talin/LHCb/-/tree/cepc-on-gaussino?ref_type=heads
gaussinoextlibs	https://gitlab.cern.ch/talin/gaussinoextlibs/-/tree/cepc-on-gaussino?ref_type=heads
Gaussino	https://gitlab.cern.ch/talin/Gaussino/-/tree/cepc-on-gaussino?ref_type=heads
CEPCSW	https://gitlab.cern.ch/talin/CEPCSW/-/tree/cepc-on-gaussino?ref_type=heads
Installation script	https://gitlab.cern.ch/talin/build-cepc-on-gaussino

Implementation (1): Geometry

- Gaussino provides several ways to setup geometry.
- Specify the factory (**GiGaFactoryBase<G4VUserDetectorConstruction>**) in GiGaMT.DetectorConstruction and setup the corresponding properties.
  - GiGaMTDetectorConstructionFAC (default): property GiGaMTGeoSvc
  - GDMLConstructionFactory: property GDML
  - DD4hepDetectorConstructionFAC: property DescriptionLocation
- If use the default factory, the geometry could be customized using the GeoSvc (**IGiGaMTGeoSvc**).
  - DD4hepCnvSvc
    - Property DescriptionLocation

```
class DD4hepCnvSvc : public extends<Service, IGiGaMTGeoSvc> {
public:
  using extends::extends;
  virtual ~DD4hepCnvSvc() = default;

  // Service pure member functions
  StatusCode initialize() override;
  StatusCode finalize() override;

  // Pointer to the root of G4 geometry tree
  G4VPhysicalVolume* constructWorld() override;
  void               constructSDandField() override;

protected:
  // Function to load the geometry into DD4hep. Virtual
  // to allow being replaced in derived classes if needed.
  virtual const dd4hep::Detector& getDetector() const;
```

```
giga = GiGaMT()
dettool = giga.addTool(
    GiGaMTDetectorConstructionFAC(),
    name="DetConst",
)
giga.DetectorConstruction = getattr(giga, "DetConst")

dettool.GiGaMTGeoSvc = self.getProp("GeometryService")
dettool.SensDetVolumeMap = self.getProp("SensDetMap")
extra_tools = self.getProp("ExtraGeoTools")
dettool.AfterGeoConstructionTools = extra_tools
add_constructors_with_names(dettool, extra_tools)

algs = []
algs += self._set_external_detector(dettool)
algs += self._set_parallel_geometry(dettool)
self._set_custom_simulation_regions(dettool)
self._set_gdml_import(dettool)
self._set_gdml_export(dettool)
```

python/Gaussino/Geometry.py

Implementation (2): Sensitive Detector

- DDG4 classes are reused with minor changes:
  - A hit type: **Geant4Hits**. This is from DDG4.
  - A DDG4 based SD: **DDG4SensitiveDetector**. Also from DDG4, but adding the GiGaMessage in the base class.
  - A concrete SD: **GenericTrackerSensitiveDetector**.
- Gaussino integration part:
  - A GiGa factory **GenericTrackerSensDetTool** is used to create the SD.
  - Declare the factory component: GenericTrackerSensDetToolComponent.

GenericTrackerSensDetToolComponent.cpp 188 B

```cpp

#include "CEPCTest/GenericTrackerSensDetTool.h"

DECLARE_COMPONENT_WITH_ID(GenericTrackerSensDetTool, "GenericTrackerSensDet")
DECLARE_COMPONENT_WITH_ID(GenericTrackerSensDetTool, "VXD")
```

In order to let SD get the DD4hep instances according to the name, the component is declared with the detector name.

```cpp
DDG4SensitiveDetector::DDG4SensitiveDetector(const std::string& name)
    : G4VSensitiveDetector(name) {
    dd4hep::Detector& description = dd4hep::Detector::getInstance();

    m_detector = description.detector(name);
    m_sensitive = description.sensitiveDetector(name);
    m_readout = m_sensitive.readout();

}
```

Implementation (3): Monitoring tool

- For testing only, the histograms of positions and deposit energies are created.

```
StatusCode GenericTrackerMonTool::monitor( const G4Event& aEvent )
{
  std::lock_guard<std::mutex> guard(m_hist_lock);
  G4HCofThisEvent* collections = aEvent.GetHCofThisEvent();
  G4VHitsCollection* collect;
  dd4hep::sim::Geant4TrackerHit* hit;
  double energyTotal;
  int hitNo;
  debug() << "There are " << collections->GetNumberOfCollections() << " hit collections." << endmsg;
  for ( int iter_coll = 0; iter_coll < collections->GetNumberOfCollections(); iter_coll++ ) {
    collect = collections->GetHC( iter_coll );
    if ( collect->GetName().find( m_coll_name) != std::string::npos ) {
      size_t n_hit = collect->GetSize();
      energyTotal  = 0;
      hitNo        = 0;
      debug() << "\t" << n_hit << " hits are stored in a calorimeter collection #" << iter_coll << ": "
                << collect->GetName() << endmsg;
      for ( size_t iter_hit = 0; iter_hit < n_hit; iter_hit++ ) {
          hit = dynamic_cast<dd4hep::sim::Geant4TrackerHit*>( collect->GetHit( iter_hit ) );

          if ( hit->energyDeposit != 0 ) hitNo++;
          m_hitX->fill( hit->position.x()/CLHEP::mm );
          m_hitY->fill( hit->position.y()/CLHEP::mm );
          m_hitZ->fill( hit->position.z()/CLHEP::mm );
          m_hitEnergy->fill( hit->energyDeposit/CLHEP::GeV );
          m_hitXY->fill( hit->position.x()/CLHEP::mm, hit->position.y()/CLHEP::mm );
          energyTotal += hit->energyDeposit/CLHEP::GeV;

      }
      std::cout << "\t" << hitNo << " hits are non-zero in collection #" << iter_coll << ": " << collect->GetName()
                << std::endl;
      std::cout << "\t" << energyTotal << " MeV = total energy stored" << std::endl;
      m_totHitEnergy->fill( energyTotal );
    }
  }
  return StatusCode::SUCCESS;
}
```

Summary

- The simulation framework in CEPCSW has been developed to support the TDR.
- In order to benefit from multi-threading techniques, it is necessary to move from serial version to multi-threaded version.
- Gaussino project from LHCb has been chosen as the next simulation framework in Key4hep.
- CEPC-on-Gaussino prototype has been implemented without dependencies on the whole LHCb software stack.

Thank you for your attention!

backup

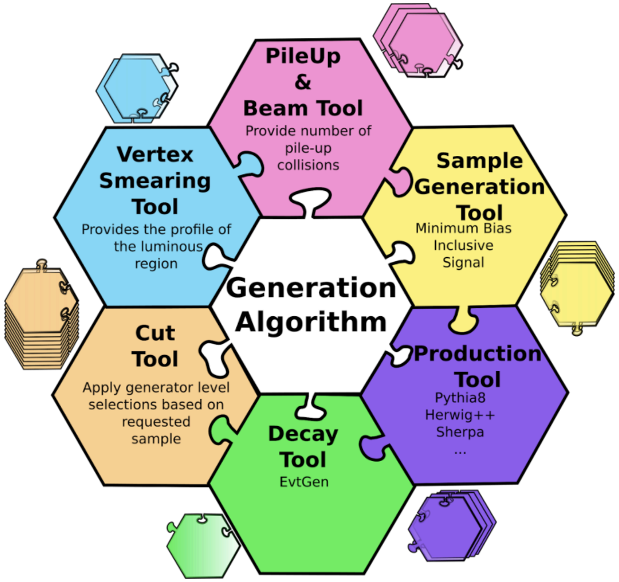
Generation phase

- Generation algorithms: “Generation” and “ParticleGun”
  - The input is LHCb GenHeader
  - The output is HepMC GenEvent

```
class Generation
    : public Gaudi::Functional::MultiTransformer<
          std::tuple<std::vector<HepMC3::GenEventPtr>, LHCb::GenCollisions, LHCb::GenHeader>( const LHCb::GenHeader& ),
          Gaudi::Functional::Traits::BaseClass_t<RndAlgSeeder>>
```

```
class ParticleGun
    : public Gaudi::Functional::MultiTransformer<
          std::tuple<HepMC3::GenEventPtrs, LHCb::GenCollisions, LHCb::GenHeader>( const LHCb::GenHeader& ),
          Gaudi::Functional::Traits::BaseClass_t<RndAlgSeeder>>
```

- The generation algorithm
  - It consists of different Gen tools
  - Thread safety of generators
- The particle gun algorithm
  - N Particles, Particle type, momentum, etc.



Detector simulation phase

- Detector simulation algorithm: GiGaAlg
  - The input is HepMC GenEvent
  - The output is G4Event and MC truths

```
class GiGaAlg : public Gaudi::Functional::MultiTransformer<std::tuple<G4EventProxies, Gaussino::MCTruthPtrs>(
                     const HepMC3::GenEventPtrs& ),
                 Gaudi::Functional::Traits::BaseClass_t<RndAlgSeeder>>
```

- GiGaMT (Service) is used by GiGaAlg to simulate events with Geant4.
  - The major interface to Geant4, including Detector Construction, Physics List and User Actions.

```
class GiGaMT : public Service, virtual public IGiGaMTSvc, virtual public IGiGaMTSetUpSvc
{
  // TODO: GiGaActionInitializer is very modular. No idea if any other option might be used here.
  // Gaudi::Property<std::string> m_MTRunMgrFactoryName{this, "MTRunManagerFactory", "GiGaMTRunManagerFAC"};
  ToolHandle<GiGaFactoryBase<GiGaMTRunManager>> m_mTRunManagerFactory{this, "MTRunManagerFactory",
                                                                      "GiGaMTRunManagerFAC"};
  ToolHandle<GiGaFactoryBase<G4VUserPhysicsList>> m_physListFactory{this, "PhysicsListFactory", "GiGaMT_FTFP_BERT"};
  ToolHandle<GiGaFactoryBase<GiGaWorkerPilot>> m_workerPilotFactory{this, "WorkerPilotFactory", "GiGaWorkerPilotFAC"};
  ToolHandle<GiGaFactoryBase<G4VUserDetectorConstruction>> m_detConstFactory{this, "DetectorConstruction",
                                                                              "GiGaMTDetectorConstructionFAC"};
  ToolHandle<GiGaFactoryBase<G4VUserActionInitialization>> m_ActionInitializerFactory{this, "ActionInitializer",
                                                                                        "GiGaActionInitializer"};
  ToolHandle<IHepMC3ToMCTruthConverter> m_converterTool{this, "HepMCConverter", "HepMC3ToMCTruthConverter"};
  ToolHandleArray<IG4MonitoringTool> m_MoniTools{this};
  Gaudi::Property<std::vector<std::string>> m_MoniToolNames{
      this, "MonitorTools", {}, tool_array_setter( m_MoniTools, m_MoniToolNames )};
```