

# Progress in CEPCSW Simulation framework and Validation system

**Tao Lin (林韬)**<sup>1</sup>, **Teng Li (李腾)**<sup>2</sup>  
(on behalf of the core software group)

lintao@ihep.ac.cn

<sup>1</sup> IHEP, <sup>2</sup> SDU

CEPC Workshop 2021

Nov 8-11, 2021

# Outline

1. Introduction
2. Progress in the Simulation Framework
3. Progress in the Validation System
4. Summary and Plan

# Introduction

- CEPC core software provides
  - **Key4hep based software:** framework (Gaudi), event management (k4FWCore), event data model (EDM4hep), geometry management (DD4hep)
  - **CEPC specific software:** simulation framework, analysis framework, machine learning, parallelization
  - Build system, CI and validation system
- Since the last CEPC workshop, the core software group made following major updates:
  - Integration with the latest Key4hep. For example, the latest version of EDM4hep is adopted.
  - Support of realistic simulation in the simulation framework.
  - Development of the validation system.
- This talk will focus on the progress of the simulation framework and validation system.

# Progress in the Simulation Framework

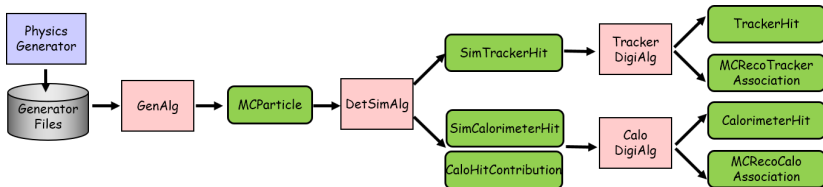
# Overview of the Simulation Framework

- A simulation framework is being developed.
  - Adopt EDM4hep as event data model and DD4hep as detector description (including the magnetic fields).
  - Develop Gaudi Algorithms to take charge of the complete simulation workflow.
  - Develop a customized G4 Run Manager to integrate with the Gaudi algorithm.
  - Support the customized detector response and the MC truth information.
  - Support the full and fast simulation transparently.
  - Support the background mixing.
- The core packages in CEPCSW:



# Simulation data flow with EDM4hep

- The EDM4hep is used at each stage:
  - physics generator → MCParticles
  - detector simulation → MCParticles, SimTrackerHits, SimCalorimeterHits
  - digitization → TrackerHits, CalorimeterHits *etc*
- Two collections of MCParticles are created: one for generator stage, one for detector simulation stage.
  - In the latest version of PODIO/EDM4hep, the loaded collections can't be modified any more.
  - The original collection is cloned and the secondary particles created in the detector simulation are put into the cloned collection.



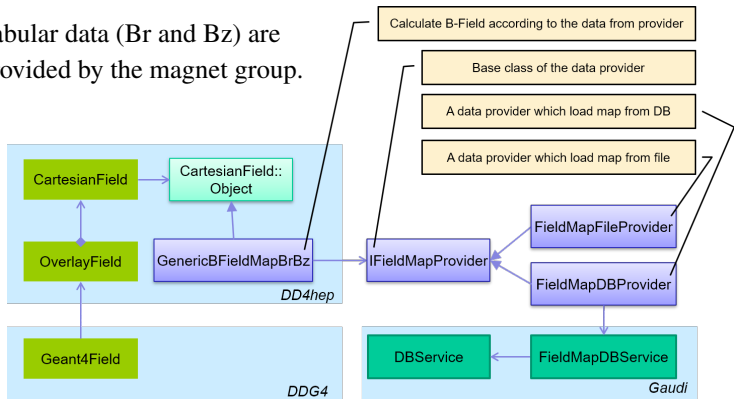
## Detector description and magnetic field (1)

- DD4hep is adopted to describe both the geometry and magnetic field.
  - A detector is described by compact files (XML), detector constructors (C++), readout (sensitive detector), segmentation (logical partition in SD) and ID Description.
- The geometry is converted from the DD4hep detector model to Geant4 detector model using DDG4 automatically.
- The readout in DD4hep is used to setup the sensitive detector in Geant4.
- The magnetic field is implemented by adding a thin wrapper layer, so the field maps are not copied from DD4hep to Geant4.
  - An implementation of uniform magnetic field is provided by DD4hep.
  - For the non-uniform magnetic field, an extension to DD4hep has been developed.

## Detector description and magnetic field (2)

- The non-uniform magnetic fields has been implemented with a generic field map to calculate the magnetic field at any point and a field map provider to provide the value on a grid.

Tabular data ( $B_r$  and  $B_z$ ) are provided by the magnet group.

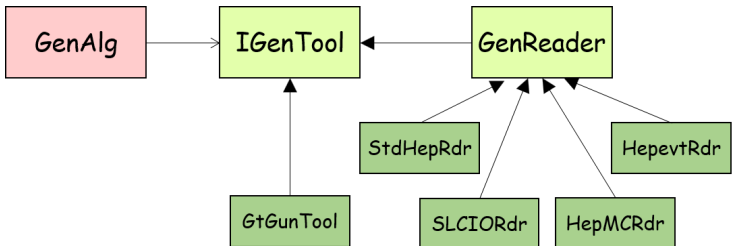


For the details, see the package `Detector/MagneticFieldMap`



# Physics generator interface

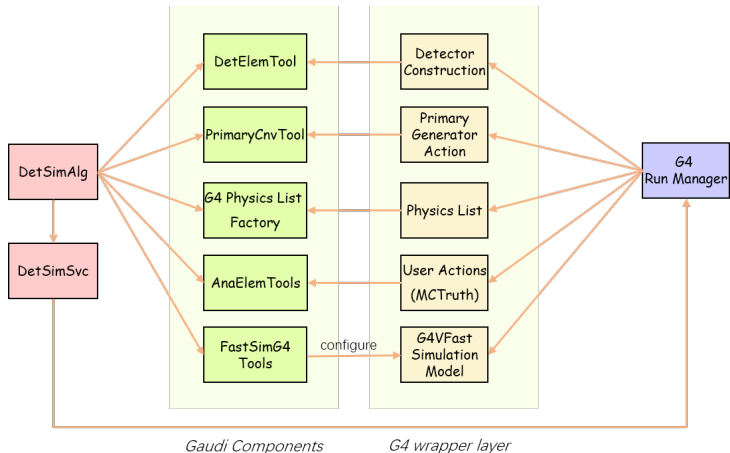
- The generator algorithm is configured with different tools
  - Particle gun
  - Support different file formats: HepEVT, HepMC, StdHep, LCIO *etc*



- It is easy to integrate more generators by adding new tools.

# Detector simulation software (1): Integration with Geant4

- Full integration is done by developing a thin G4 wrapper layer:



## Detector simulation software (2): Physics lists

- The simulation framework provides a flexible way to configure the physics list.
- Using the helper class `G4PhysListFactory` from Geant4 to create the physics list.
  - The default is `QGSP_BERT`.
  - More lists could be found in [https://geant4.kek.jp/lxr/source/physics\\_lists/lists/src/G4PhysListFactory.cc](https://geant4.kek.jp/lxr/source/physics_lists/lists/src/G4PhysListFactory.cc)
- Additional physics processes are also enabled
  - PAI or PAI photon model in EM processes
  - Step limiters
  - Fast simulation
- Due to time consuming, the optical processes are not enabled.

## Detector simulation software (3): Detector responses

- The detector responses are implemented in the Geant4 sensitive detectors.
- Reuse part of DDG4 as the Geant4 hit objects are already defined.
- For the dedicated detectors, following SDs are implemented:
  - Calorimeter: `CalorimeterSensDetTool`
  - Drift Chamber: `DriftChamberSensDetTool`
  - TPC: `TimeProjectionChamberSensDetTool`
- At the end of each event, the Geant4 hit objects are converted to EDM4hep objects.

## Detector simulation software (4): MC truth

All the necessary relationships to re-build the relations between reconstructed particles and MC particles are stored:

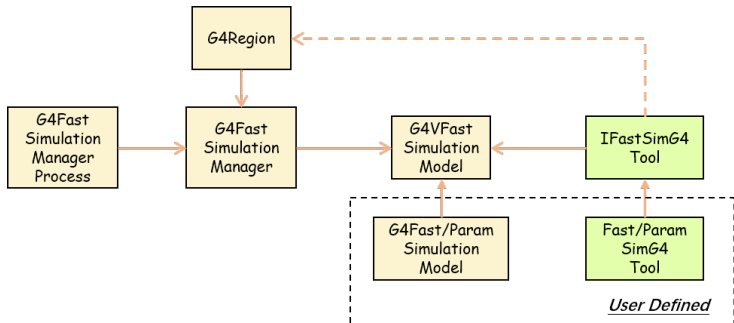
- The primary particles and decayed secondaries are stored in the MC particle collections.
- Given a hit object, the primary particle could be retrieved.
- There is a flag to indicate the hit is generated from secondary or not.

For the user defined MC truth information, users can maintain their own user trees by the ntuple service of Gaudi.

- Users are encouraged to contribute the implementation into the official repository.
- All the user output of the defined MC truth can be controlled in job options.

## Detector simulation software (5): Fast simulation

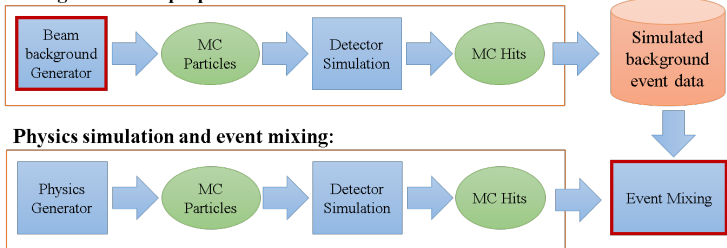
- The fast simulation interface is developed in the simulation framework in order to integrate with Geant4.
  - The FastSimG4Tools are used to create and manage the fast simulation models.
  - When a particle enters a region setup with fast simulation model, Geant4 will trigger the fast simulation.



# Background simulation (1)

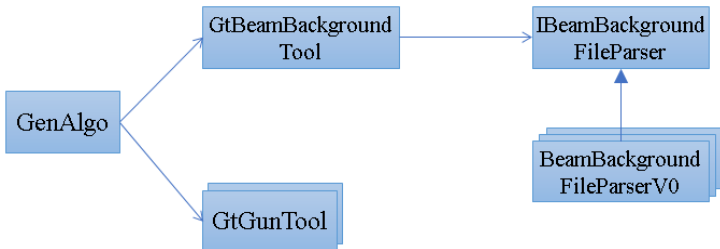
- The background simulation needs to be considered in a more realistic simulation.
- The MC hit-level background mixing is adopted in the design.
  - The simulated background data is produced first and then reused by the event mixing.
  - Two modules are under development: one for the background generator, another for the event mixing module.

## Background data preparation:



## Background simulation (2)

- The beam background generator interface is developed to read the beam background data.
  - A file in a dedicated format provides the information of a MC particle, such as position and momentum.
  - Particle type, energy and rotation (half crossing angle) could be configured.
  - Multiple inputs are allowed, so user could specify two directions of the beams.

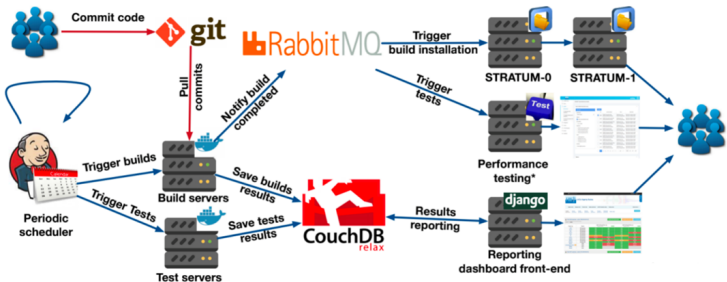




# Progress in the Validation System

# About Continuous Integration (CI) and Validation System

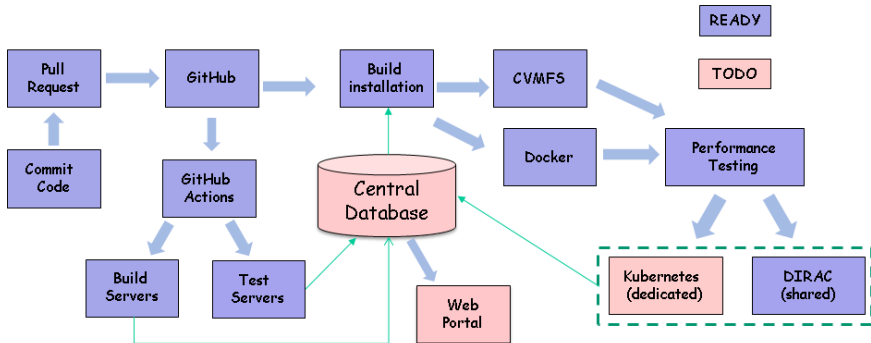
- Chris Burr from LHCb gave a nice talk in the last year's workshop.
- The CI and validation system consists of
  - Standard/Customized CI
  - Nightly build system
  - Dashboard



The system comprises several open source software, which needs the integration in order to let them work together.

# Status of the validation system for CEPC

- A CI and validation system is proposed for CEPC.
  - As a generic solution, it could be also used in the Key4hep project.
- A prototype has been setup in the last year.
  - Hardware: dedicated computing resources for the CI and validation system
  - Software: GitHub/Gitlab Runner, validation toolkit (jMonitor), dashboard



# Self-hosted GitHub Runners

- The pull requests or merge will trigger the build system automatically.
  - If any problems found, the developers will be notified automatically.
- The build matrix is used to define the different build configurations.

✓ Merge pull request #200 from mirgquest/master CI #274

🔄 Re-run all jobs



🏠 Summary

Jobs

✓ build (LCG\_98, ninja)

✓ build (KEY4HEP\_STACK, ninja)

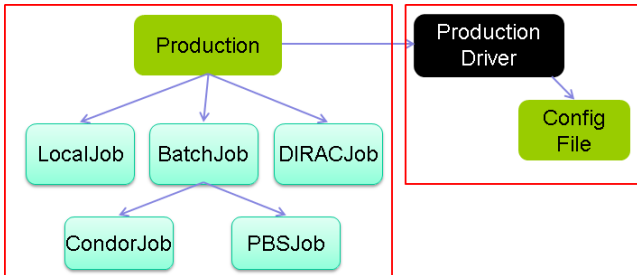
```
build (LCG_98, ninja)
succeeded 11 hours ago in 2m 53s

Search logs

> ✓ Set up job 3s
> ✓ Run actions/check.out@v2 3s
> ✓ Run a one-line script 0s
> ✓ Run a multi-line script 0s
▼ ✓ Run the build script 2m 47s
  1 ▶ Run pwd
  8 /tmp/lint/github-runner/github-runner-1/_work/CEPCSW/CEPCSW
  9 LCG_RELEASE: LCG_98
  10 CEPCSW_BLDTOOL: ninja
  11 INFO: Setup CEPCSW externals: /cvmfs/cepcsw.ihep.ac.cn/prototype/releases/externals/98.0.0/setup-
  98.0.0.sh
  12 PID TTY          TIME CMD
  13 31206 ?                00:00:00 build.sh
```

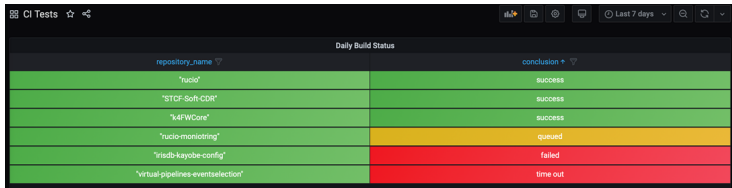
# Validation toolkit

- Docker images are created and the CVMFS is setup automatically.
  - <https://hub.docker.com/r/cepc/cepcsw-cvmfs>
- A profiler called jMonitor has been developed.
  - Support customized workflow, including log parser, CPU/memory usage, automatic result comparison
  - <https://github.com/key4hep/key4hep-validation/tree/main/Profiler>
- A new back-end is added to support the DIRAC jobs, so that long jobs could be run on the GRID system.



# Dashboard

- A dashboard is under development based on the existing technologies, such as GitHub API, Metrics Collector, InfluxDB and Grafana.
- From the dashboard, it is easy for the users to monitor all their tasks.



The screenshot shows a dashboard titled "CI Tests" with a "Daily Build Status" table. The table has two columns: "repository\_name" and "conclusion". The rows represent different repositories and their build statuses.

repository_name	conclusion
"rucio"	success
"STCF-Soft-CDR"	success
"k4FWCore"	success
"rucio-monitoring"	queued
"irisdb-kayobe-config"	failed
"virtual-pipelines-eventsselection"	time out

# Summary and Plan

## Simulation framework

- The simulation framework has been developed.
- More realistic simulation is already considered, including the non-uniform magnetic field and the background mixing.

## Validation system

- The validation system is setup from scratch and most of the functionalities are work now.
- As this system will be also used by Key4hep projects, the development of the validation system is high priority.
- The complete system will be ready in the next year.

CEPCSW GitHub: <https://github.com/cepc/CEPCSW>

Thank you