Progress of the software framework for CEPC

Tao Lin on behalf of the CEPCSW working group

(Wenxing Fang, Chengdong Fu, Fangyi Guo, Xingtao Huang, Teng Li, Weidong Li, Chunxiu Liu, Mengyao Liu, Manqi Ruan, Xin Shi, Shengsen Sun, Linghui Wu, Shuiting Xin, Dan Yu, Ye Yuan, Yao Zhang, Guang Zhao, Mingrui Zhao, Jiaheng Zou, etc.)

lintao@ihep.ac.cn
IHEP

CPS-HEP (Virtual)
August 16-20, 2021
Qingdao, China

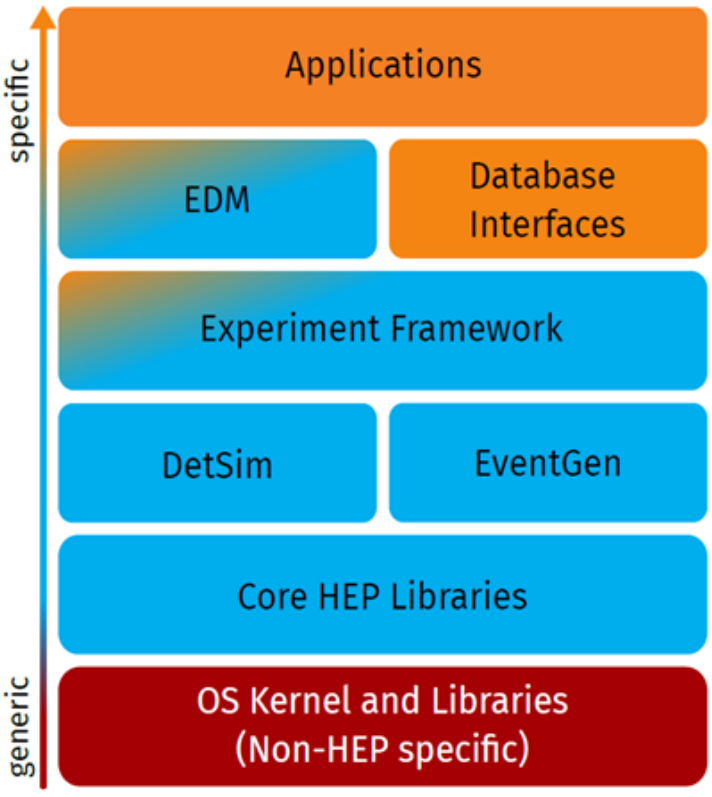
Outline

1. Introduction
2. CEPCSW and Core Software
3. Event Data Model and I/O
4. Detector description and geometry management
5. Detector simulation framework
6. Software infrastructure
7. Summary

Introduction

- CEPC software originally started from the iLCSoft and is used for the CDR study.
  - LCIO (Event Data Model and I/O), Gear (Geometry management), MokkaC (Simulation), Marlin (Reconstruction and analysis)
- April 2019: proposed a new CEPC software prototype for TDR study at Oxford workshop.
- June 2019: reached the consensus among CEPC, FCC, ILC, CLIC etc. at Bologna workshop.
  - Develop a Common Turnkey Software Stack (Key4hep) for future collider experiments.
  - Share the software components between different experiments.
- Now: CEPCSW is fully integrated with Key4hep and supports the development of simulation and reconstruction algorithms.

Common Software Stack

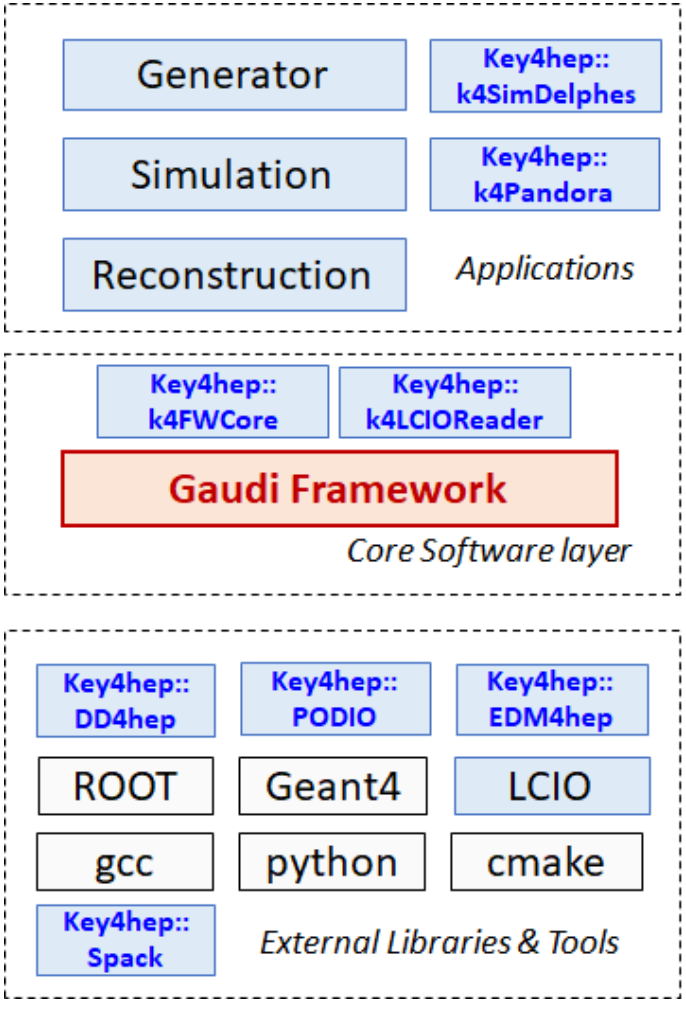


From Thomas Madlener, Epiphany Conference 2021

- Application layer of modules/algorithms/processors performing physics task
- Data access and representation layer
- Experiment core orchestration layer
  - (Marlin, Gaudi, CMSSW, ...)
- Specific components reused by many experiments
  - (DD4hep, Delphes, Pythia, ...)
- Commonly used HEP core libraries
  - (ROOT, Geant4, CLHEP, ...)
- Commonly used tools and libraries
  - (Python, CMake, boost, ...)

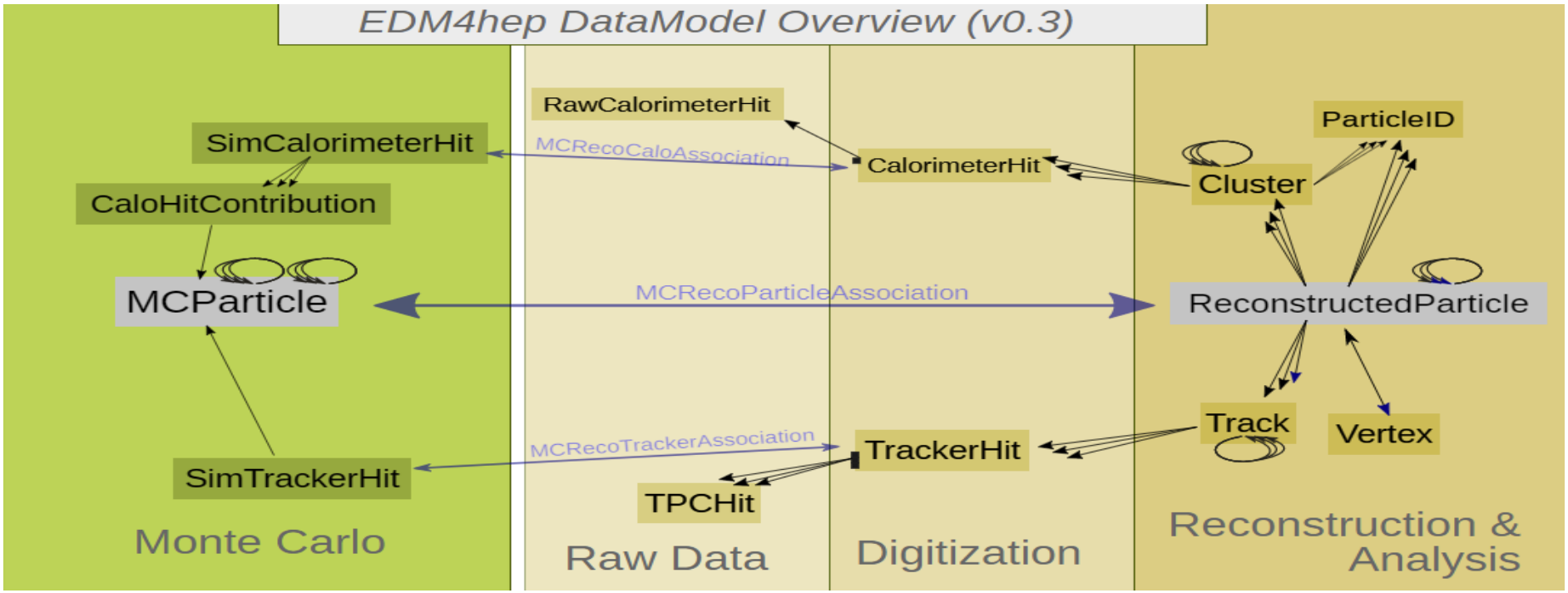
CEPCSW and Core Software

- Archicture of CEPCSW
  - External libraries
  - Core software
  - CEPC applications for simulation, reconstruction and analysis
- Core software
  - Gaudi framework: defines interfaces to all software components and controls their execution.
  - EDM4hep: generic event data model for HEP experiments
  - FWCore: manages the event data
  - DD4hep: geometry description
- The software components included in Key4hep are used in CEPCSW.



Event Data Model with EDM4hep

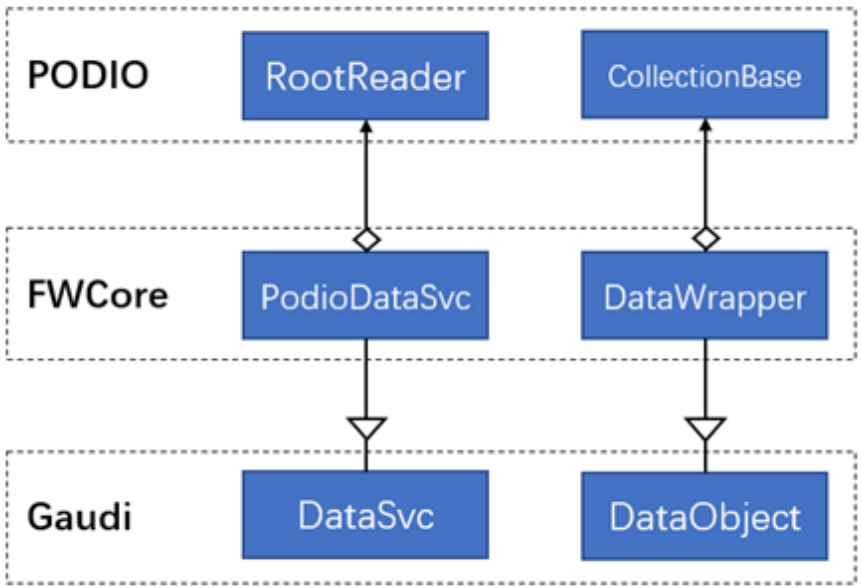
- Adopted EDM4hep as the official EDM
- Inspired by LCIO
- Generated from YAML files via PODIO toolkit



- `https://github.com/key4hep/EDM4hep`

k4FWCore: integration with Gaudi

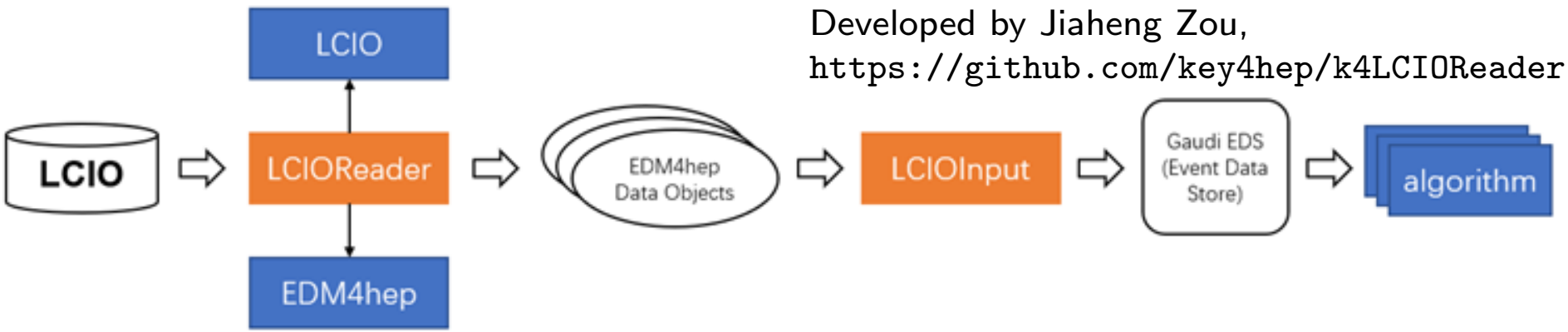
- k4FWCore provides the management of EDM4hep in Gaudi.
  - PodioDataSvc: data I/O (PODIO)
  - DataWrapper: PODIO data collection managed in Gaudi's Event Data Store.



- `https://github.com/key4hep/k4FWCore`

Reading LCIO Data

- k4LCIOReader: generate EDM4hep data collections on the fly from LCIO input files in Gaudi.

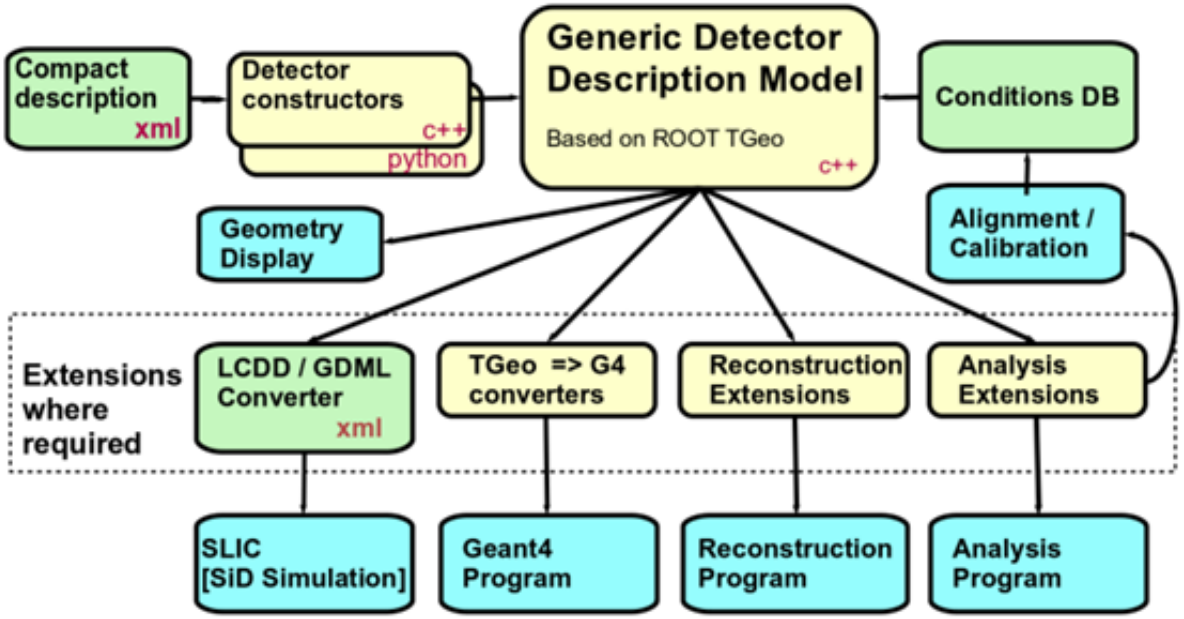


Developed by Jiaheng Zou,
`https://github.com/key4hep/k4LCIOReader`

- SIO-backend PODIO: save/read EDM4hep data objects in SIO format
  - SIO is originally a part of LCIO.
  - Now a standalone project.
  - `https://github.com/iLCSoft/SIO`

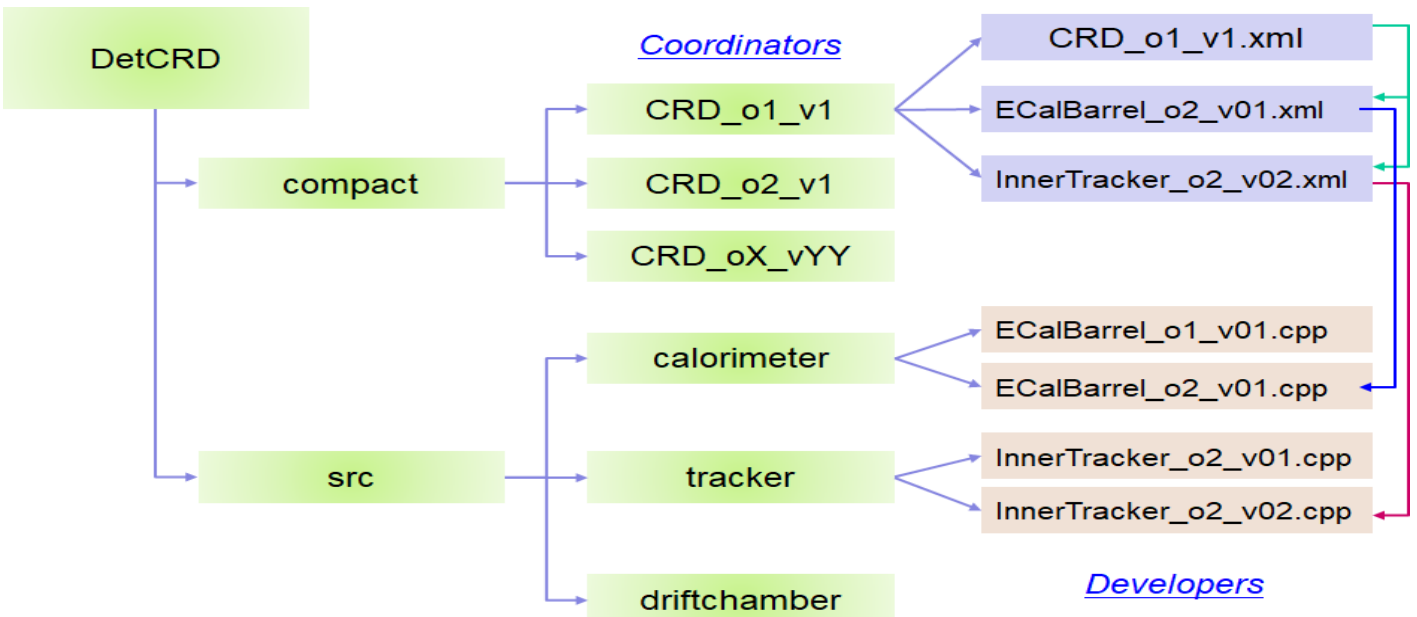
DD4hep: Detector Description Toolkit

- Originally developed for ILC and CLIC but with all of HEP in mind.
- A complete detector description with a single source of information
  - Geometry, materials, visualization, readout, alignment, calibration, reconstruction etc.
- Covering the full life cycle of an experiment
  - Detector concepts, optimization, construction and operation



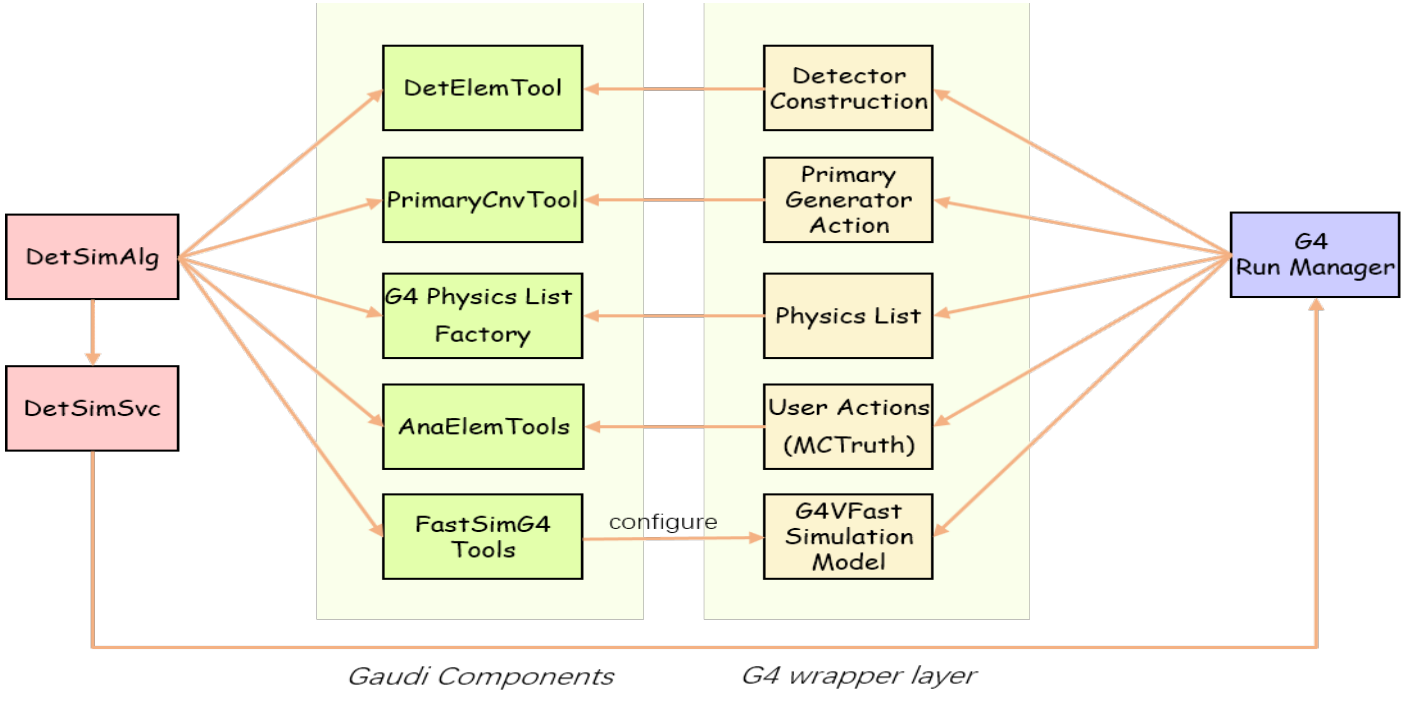
Detector Geometry Management

- The detector geometry convention
  - Sub-detectors: described by the XML compact files and the C++ constructors.
  - Full-detector options: only configured by the compact files.
- All the detector options could be managed by database an git repository.



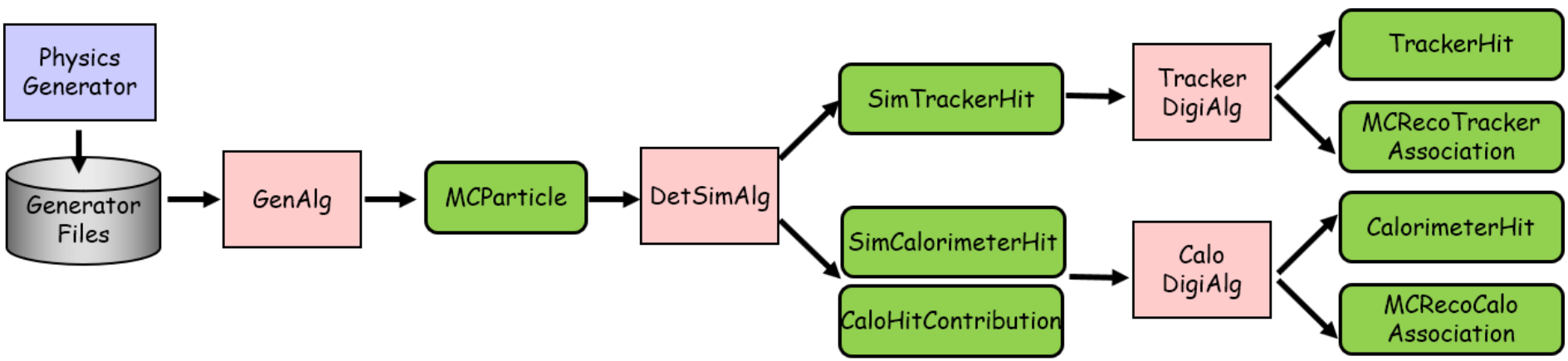
Detector simulation framework

- A detector simulation framework has been developed in CEPCSW.
  - A thin layer is developed to integrate Geant4 and Gaudi.
  - The event loop is controlled by Gaudi with a customized G4RunManager.
  - The geometry conversion from DD4hep to Geant4 is done by DDG4.



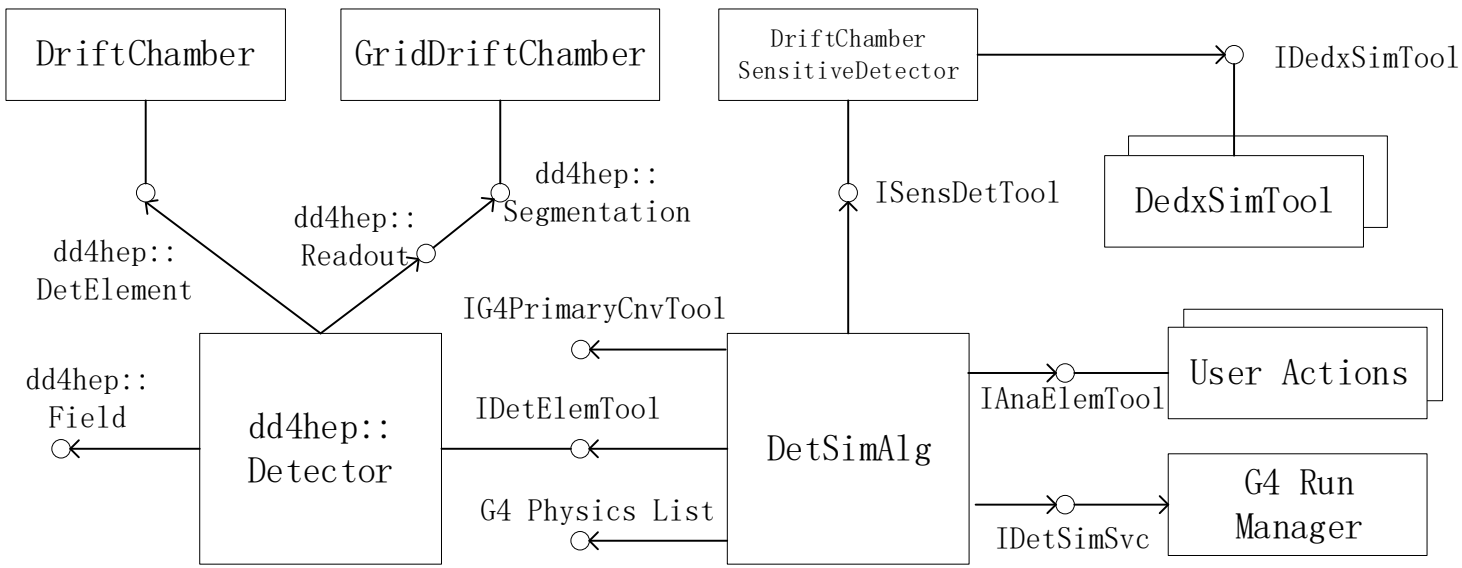
Data Associations in MC Data Production

- Physics generator interfaces provides the primary particles information.
- Detector simulation provides the MCTruth information, which links the hits and primary MCParticles.
- Digitization provides the association between the Digi objects and Hit objects via Associations.



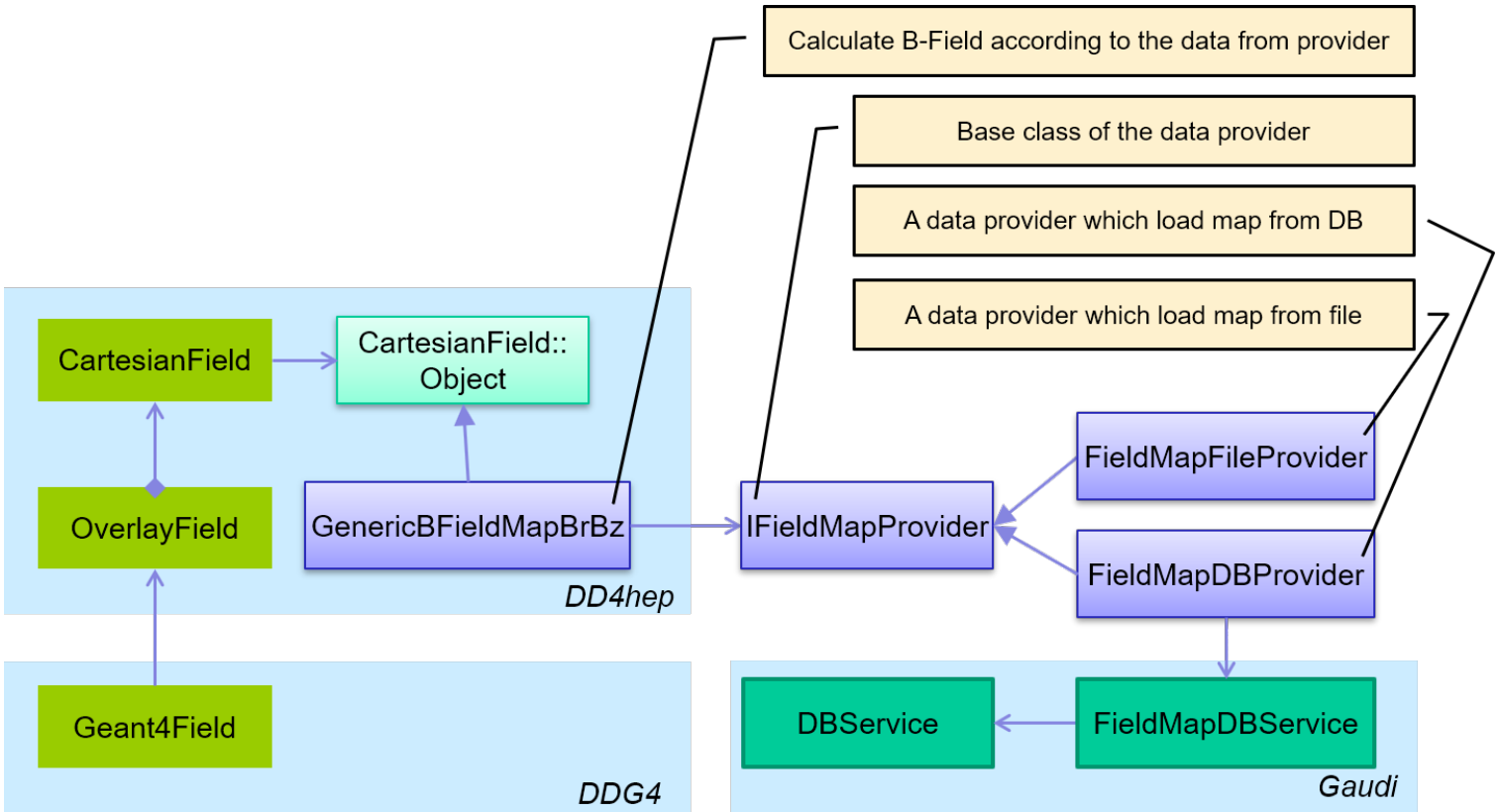
An example: Drift Chamber in Reference Detector

- The simulation of drift chamber is fully implemented in this framework.
  - The simulation of dE/dx or dN/dx is implemented in Gaudi tools.



An example: Non-uniform B-fields

- The simulation of non-uniform B-fields is also implemented.

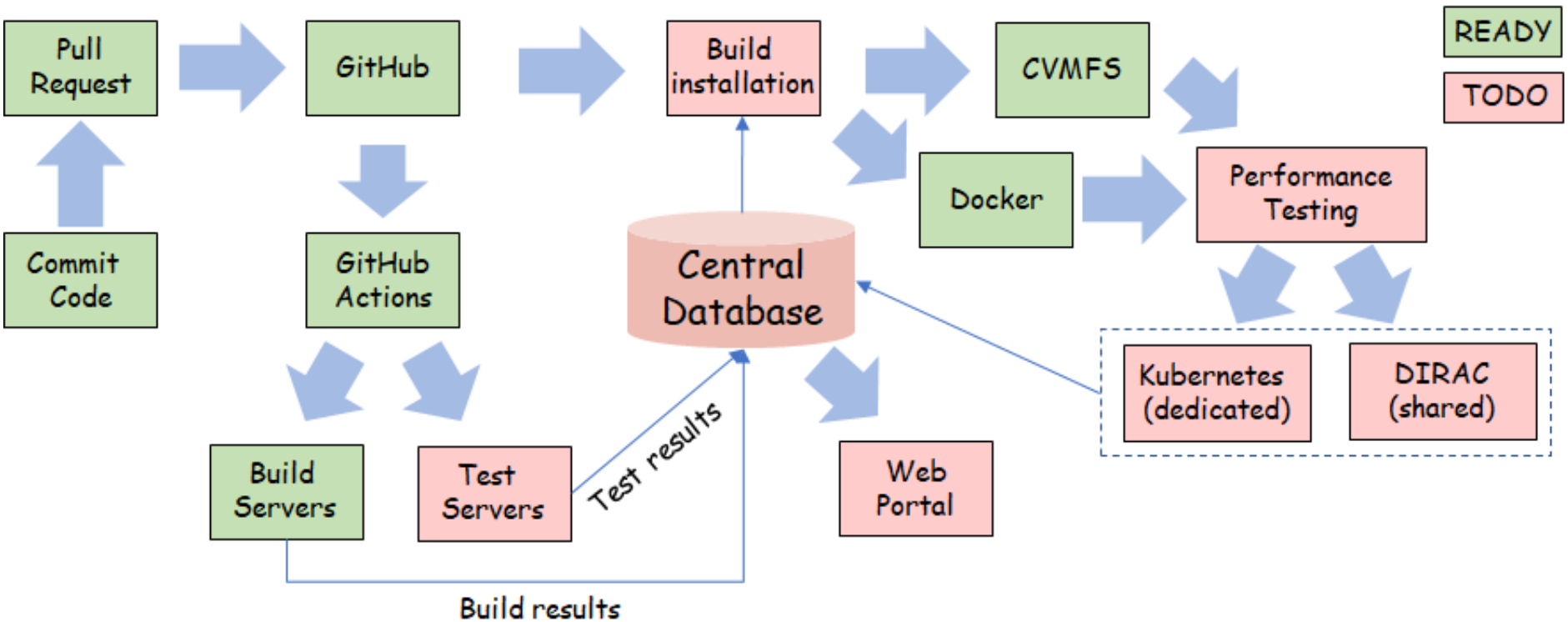


Software Infrastructure

- Source code is open and available in GitHub.
- Pull Request (PR) workflows.
- Modern OS/C++/Python
  - CentOS 7 is the default OS.
  - Modern C++ 17 standard as default.
  - Python 3 is used for the job configuration.
- Modern CMake configuration.
  - Facilitates the usages as dependencies for other components.
  - Supports Make and Ninja backends.
- The pre-compiled external libraries are distributed via CVMFS and accessible in all sites.
  - Also supports both LCG stack (CERN EP-SFT) and Key4hep stack (via Spack).
- The Docker container based solution is also provided, so users could setup CEPCSW in their own computers.

Automatic Validation

- Testing and validation are run automatically at different levels.
- GitHub Action is now used for the automatic validation.



Summary

- Key4hep project is setting up the common HEP software stack to support the future experiments.
  - Reuse the existing software components (Gaudi, PODIO, DD4hep).
  - Develop the new components (EDM4hep).
- CEPCSW is the first application of Key4hep.
  - A lot of migration work and new developments had been made by the software group.
  - New software components are also contributed to the Key4hep.
- The CEPCSW is complete open source, welcome the new users and developers.

CEPCSW GitHub: `https://github.com/cepc/CEPCSW`

Thank you