CEPCSW Prototype and Future Plan

Zou Jiaheng

On behalf of the CEPCSW working group

2019.11.18

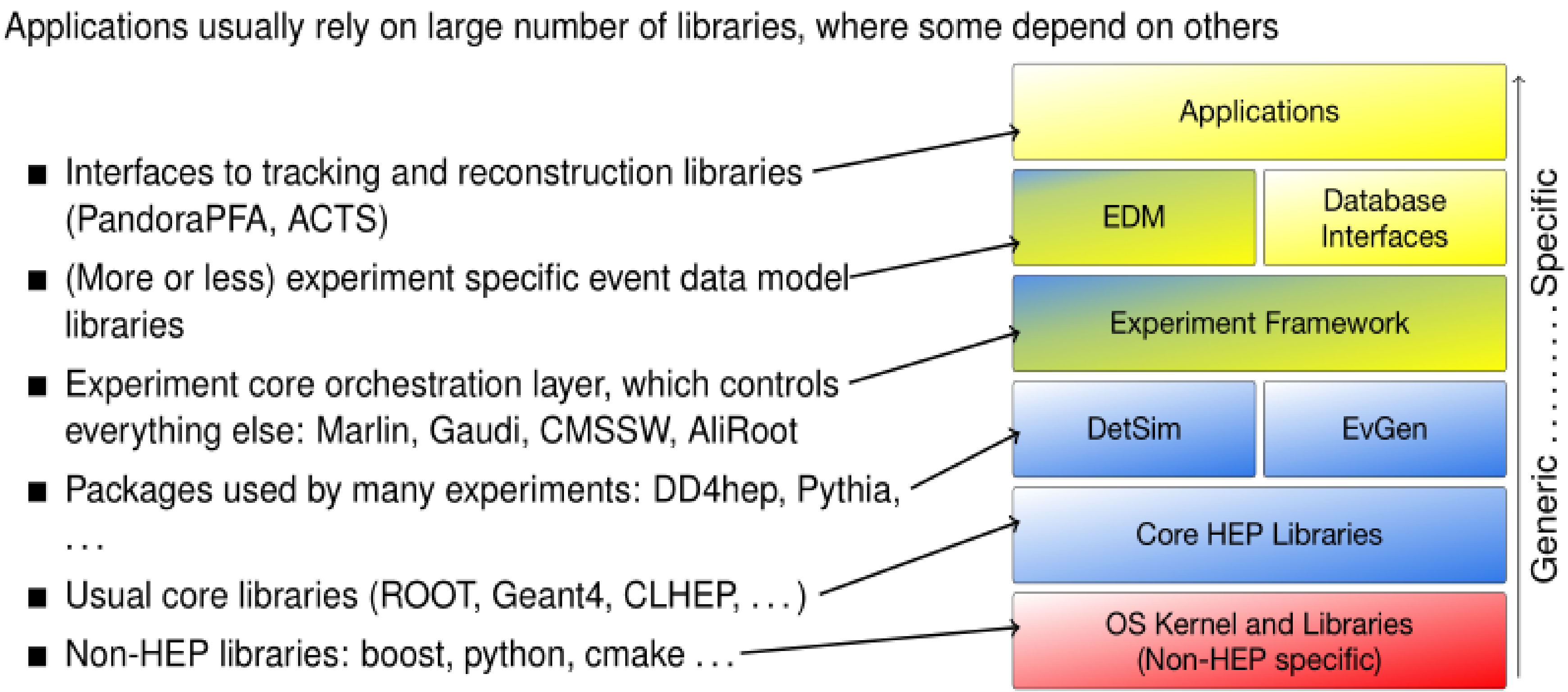
Introduction of CEPC Software

- CEPC software originally started from the iLCSoft (many thanks)
  - LCIO, Marlin, tracking and flavor-tagging
  - New components for CEPC: simulation, reconstruction…
  - Used for the CDR study, which is released in Nov, 2018
- A new framework for TDR is considered at the Oxford workshop, April 2019
  - to demonstrate the capabilities to meet future requirements
  - to support continuous integrations of new software components
- The common view at the Bolognia workshop, June 2019
  - A Common Software Stack (KEY4hep) for future collider experiments
    - CEPC, CLIC, FCC, ILC, SCTF
  - Maximize the sharing of software components between experiments

A typical HEP Software Stack

[Ref]: André Sailer, etc. , CHEP2019

https://indico.cern.ch/event/773049/contributions/3474763/attachments/1938664/3213633/191105_sailer_key4hep.pdf



Applications usually rely on large number of libraries, where some depend on others

- Interfaces to tracking and reconstruction libraries (PandoraPFA, ACTS)
- (More or less) experiment specific event data model libraries
- Experiment core orchestration layer, which controls everything else: Marlin, Gaudi, CMSSW, AliRoot
- Packages used by many experiments: DD4hep, Pythia, . . .
- Usual core libraries (ROOT, Geant4, CLHEP, . . .)
- Non-HEP libraries: boost, python, cmake . . .

Applications

EDM

Database Interfaces

Experiment Framework

DetSim

EvGen

Core HEP Libraries

OS Kernel and Libraries (Non-HEP specific)

Generic . . . . Specific

The Goal of CEPCSW Prototype

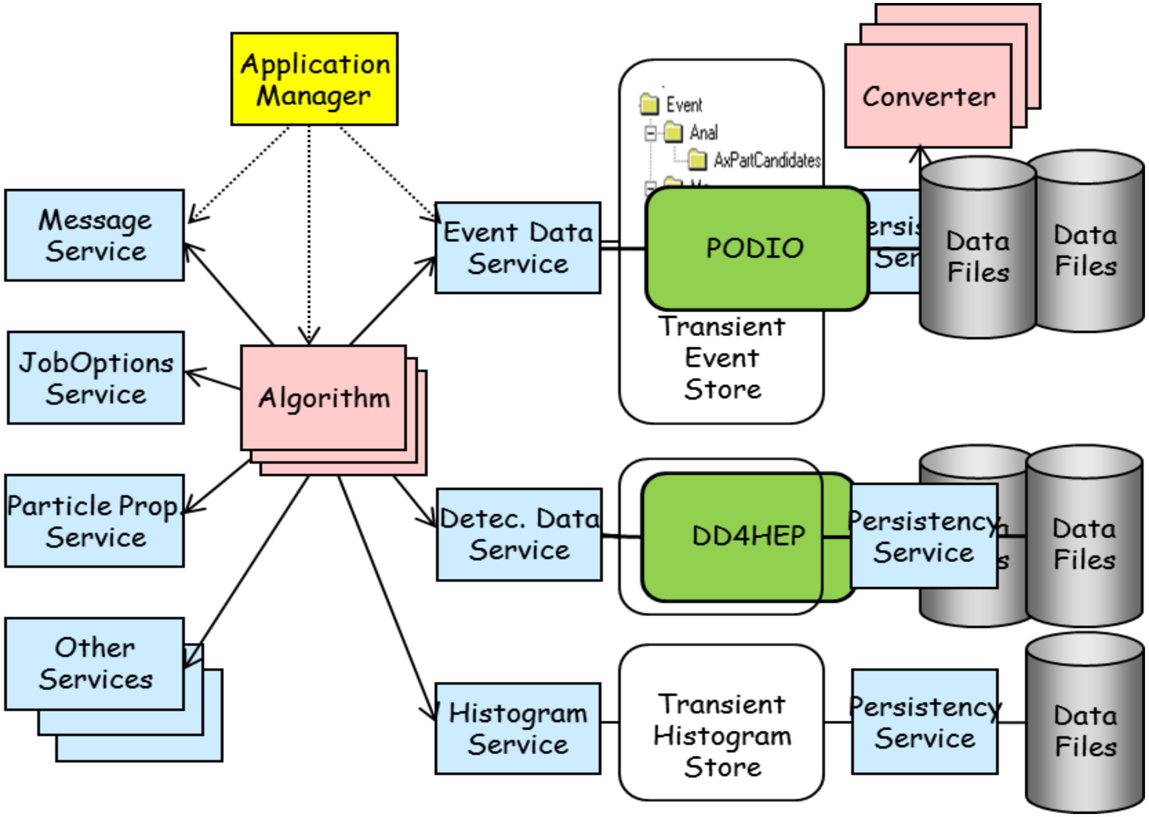
- Based on KEY4hep (Common Software Stack for HEP)
- Reuse existing components
  - EDM4hep/PODIO, DD4hep, Gaudi, ROOT …
- Implement the specific components for CEPC
- Provide a ready-to-work environment to algorithm developers and physicists
  - Migrate tracking algorithms from Marlin to CEPCSW
  - Integrate more algorithms and features
- Move from Marlin to the new software system

Tasks of CEPCSW Prototype

Components	Tasks	Status
General	Software infrastructure Core modules	√
EDM & I/O	PLCIO data model and I/O LCIO compatible reader	√
Geometry and Simulation	DD4hep integration Simulation framework	√
Reconstruction	SiliconTracking	√
	More reconstruction algorithms	In progress
Build and release	Git, CMake, CVMFS	Ready

Gaudi: the Underlying Framework

- The core part of the framework is small
- key components:
  - Application Manager
  - Services
  - Algorithms
  - Tools



- Data is separated from algorithms – physicists can concentrate on the algorithms
- Originally developed for LHCb, also used by BESIII and DYB in China

The Gaudi Framework

- Application manager: the job controller
  - Creation, configuration and management of services and algorithms
  - Algorithm scheduling during the event loop
  - Terminating the job properly
- User components
  - Algorithm: the concrete calculations to the event
  - Service: the common functions which can be invoked by users
  - Tool: subroutines belong to an algorithm
- High Performance Computing
  - Multithreading computing is supported since v29
  - Parallelized functional and reentrant algorithms
  - Transparent data management in memory

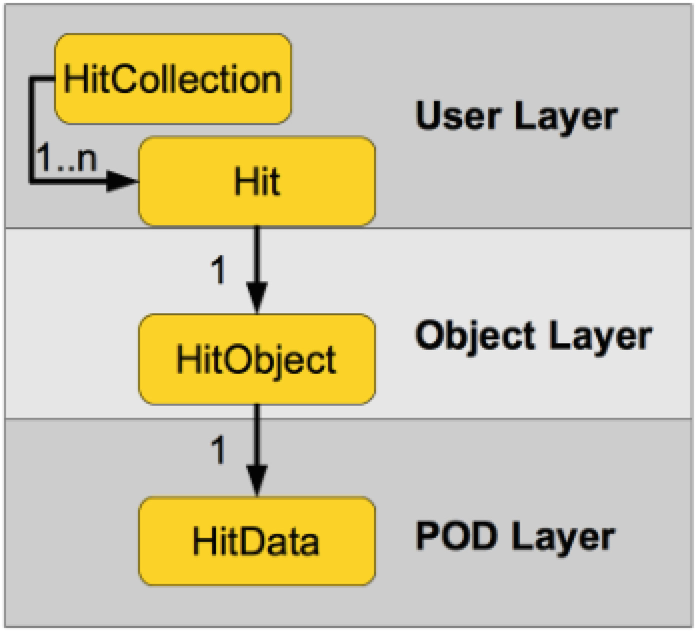
PODIO: an Event-Data Model toolkit

[Ref]: F. Gaede, etc. , CHEP2019

https://indico.cern.ch/event/773049/contributions/3473254/attachments/1939721/3215730/gaede_podio_chep19.pdf

PODIO is originally developed in context of the FCC study

- user layer (API):
  - handles to EDM objects (e.g. **Hit**)
  - collections of EDM object handles (e.g. **HitCollection**).
- object layer
  - transient objects (e.g. **HitObject**) handling *references* to other objects and *vector members*
- POD layer
  - the actual POD data structures holding the persistent information (e.g. **HitData**)



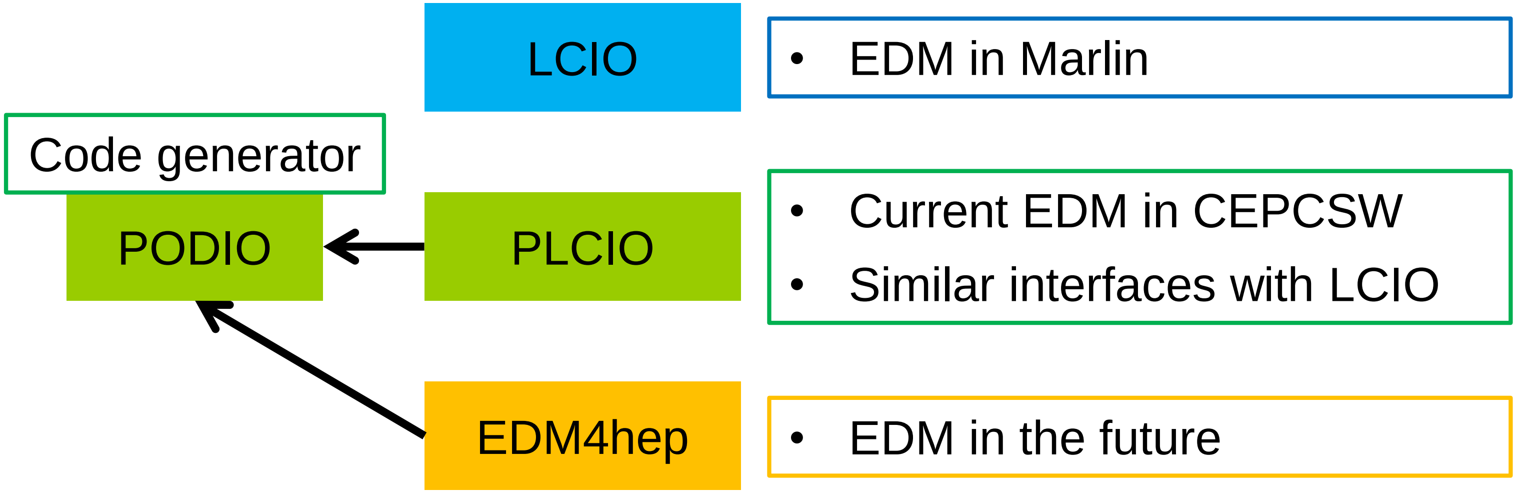
direct access to POD also possible - if needed for performance reason

EDM4hep

- The EDM4hep project is being constructed in the context of CSS
  - Based on LCIO and FCC-edm
- Provide a common event data model
  - Common core classes described in a yaml file
  - C++ Code is generated by PODIO
  - The persistency layer (ROOT, HDF5, ...) can be changed easily
  - Each experiment can implement their own extensions
- A project followed by HEP Software Foundation
  - Regular meeting in every 2 weeks (CERN, DESY, IHEP ...)
  - https://github.com/HSF/EDM4hep
- But, it is not ready yet : (

Current EDM in CEPCSW Prototype

- CEPCSW will use PLCIO before EDM4hep is ready
- PLCIO is an implementation of the LCIO event data model in PODIO



LCIO

- EDM in Marlin

Code generator

PODIO

PLCIO

- Current EDM in CEPCSW
- Similar interfaces with LCIO

EDM4hep

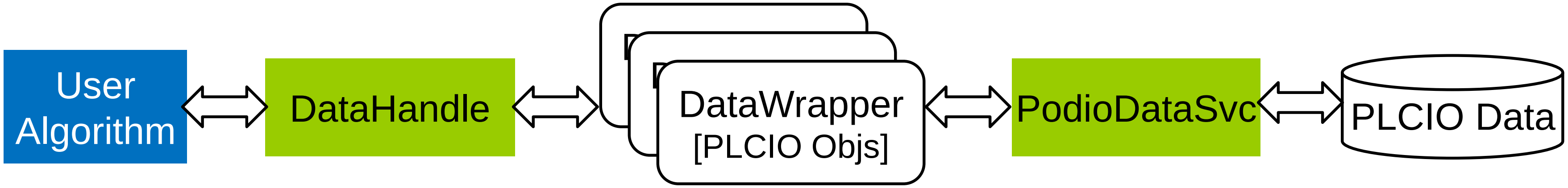
- EDM in the future

- EDM4hep
  - The migration from PLCIO to EDM4hep should be easy
- CEPC is the first user of PLCIO
  - Missing classes, potential problems …

FWCore

FCCSW FWCore

- DataWrapper: PLCIO data collection -> DataObject in Gaudi
- DataHandle: user interface to register/retrieve data to/from Gaudi TES (Transient Event Store)
- PODIO data service: read/write PODIO data objects

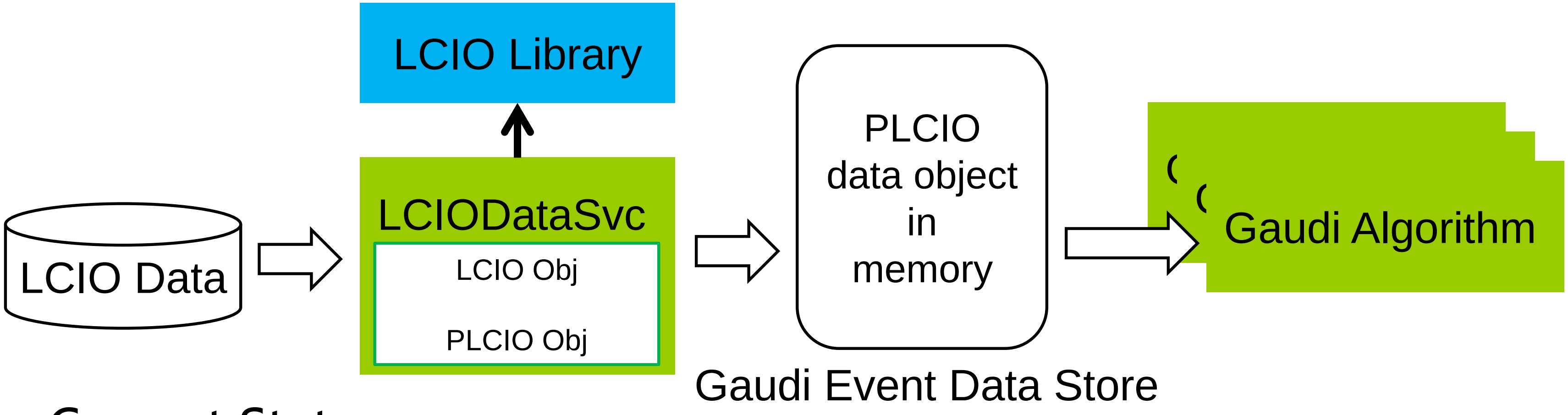


CEPCSW FWCore

- Mainly taken from FCCSW FWCore (many thanks)
- Extension to read LCIO data generated by Marlin

Read the Existing LCIO Data

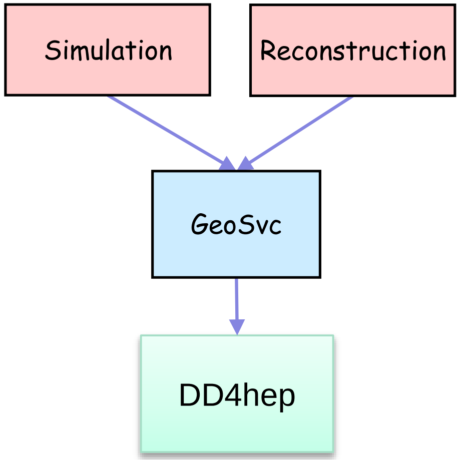
- LCIODataSvc
  - Read LCIO files via the LCIO library
  - Convert LCIO data objects to PLCIO data objects
  - Register PLCIO data objects to Gaudi Event Data Store

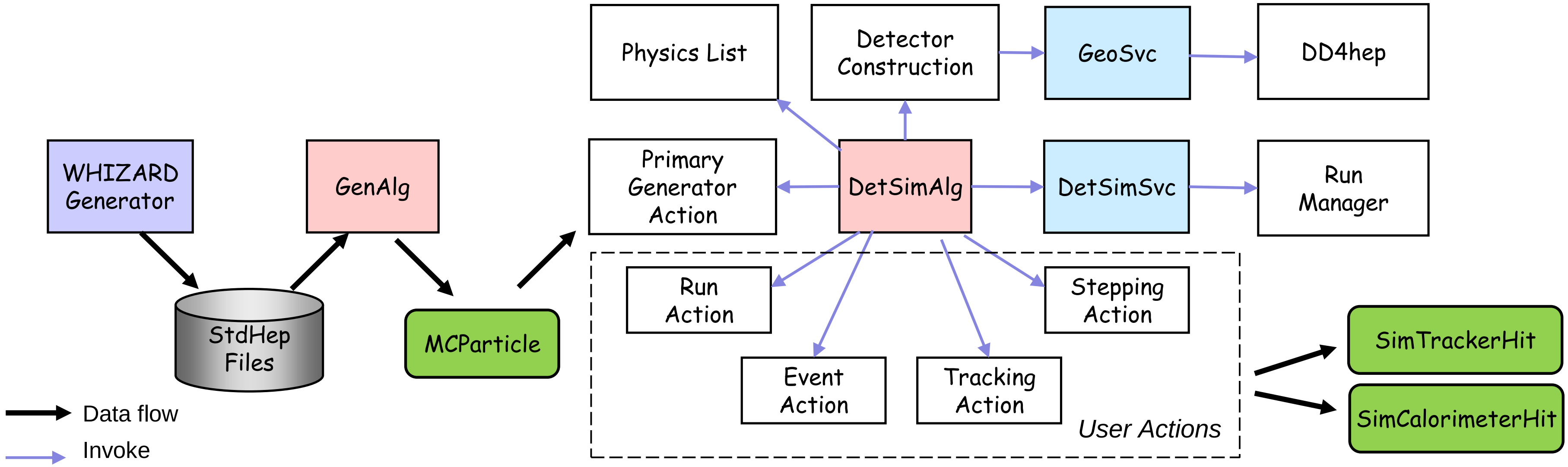


- Current Status
  - Data converters for reconstructed data types
  - Some of the data relations are not fully recovered (there are some limits to data analysis now)

Detector Description and Simulation

- Unified Geometry Service
  - Interfaced to DD4HEP
  - Used by simulation and reconstruction
- Simulation tool
  - Integrated with physics generator & Geant4
- See Tao’s report later

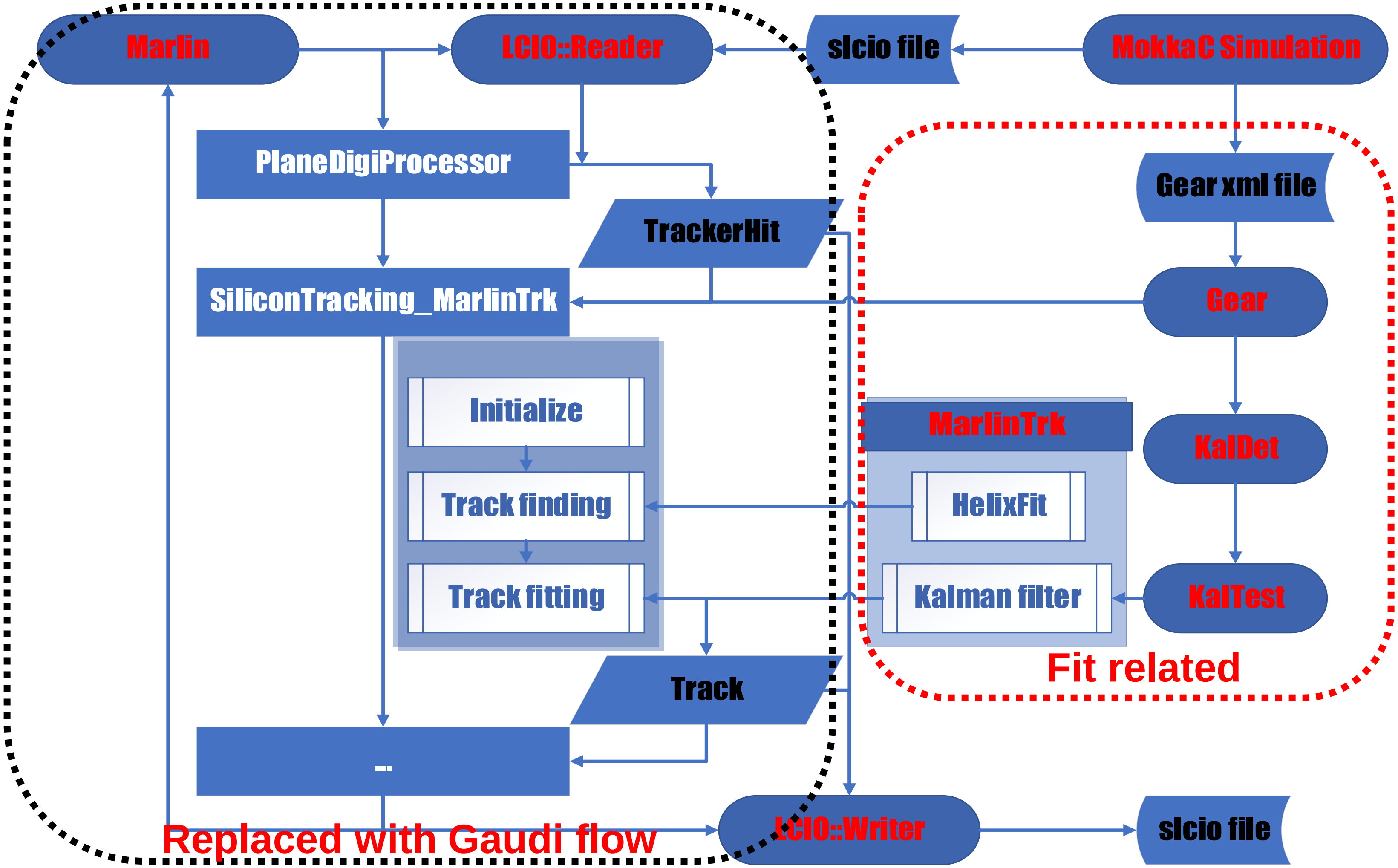




Reconstruction: SiliconTracking

As a first step, the SiliconTracking algorithm is migrated from Marlin to CEPCSW

The results are intelligible same as Marlin's. See Chengdong's report later



Software Infrastructure and Building

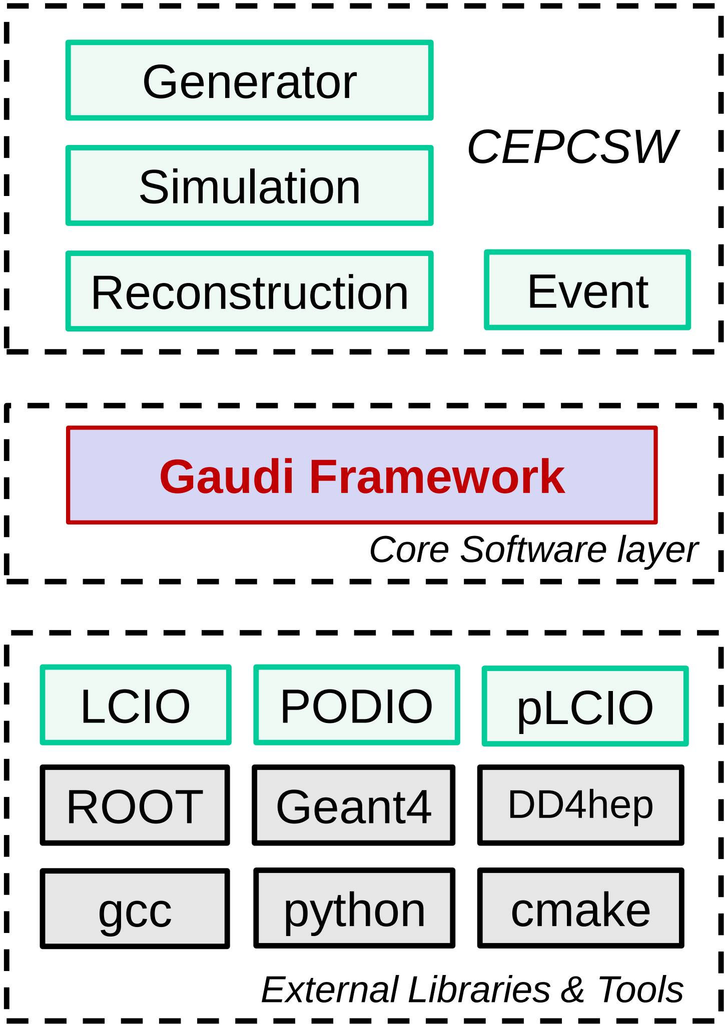
Common tools

- CMake: Build & deployment
  - Gaudi cmake macros
- Git: version control
  - http://cepcgit.ihep.ac.cn/cepc-prototype
- CVMFS: software distribution
  - CEPC specific:

    /cvmfs/cepcsw.ihep.ac.cn/prototype

Software building

- Based on FCCSW & LCG software stack now (many thanks)
- Move to KEY4hep in the in the future

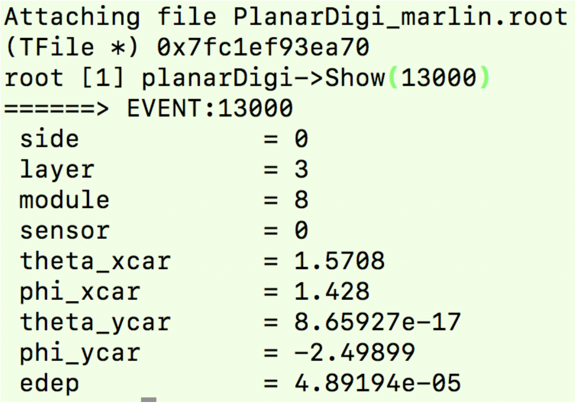


A Preliminary Testing

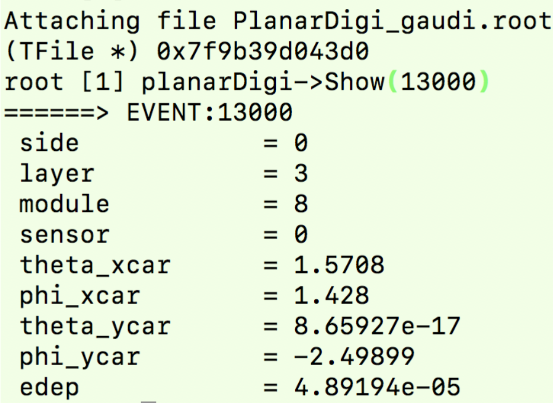
- A digitization algorithm migrated from Marlin
- Geometry: GearSvc migrated from Marlin
- Data and I/O
  - Read .slcio (LCIO) format files with LCIODataSvc
  - Write .podio (PLCIO) format files with PodioDataSvc
- Compare the results with Marlin

Physics Results

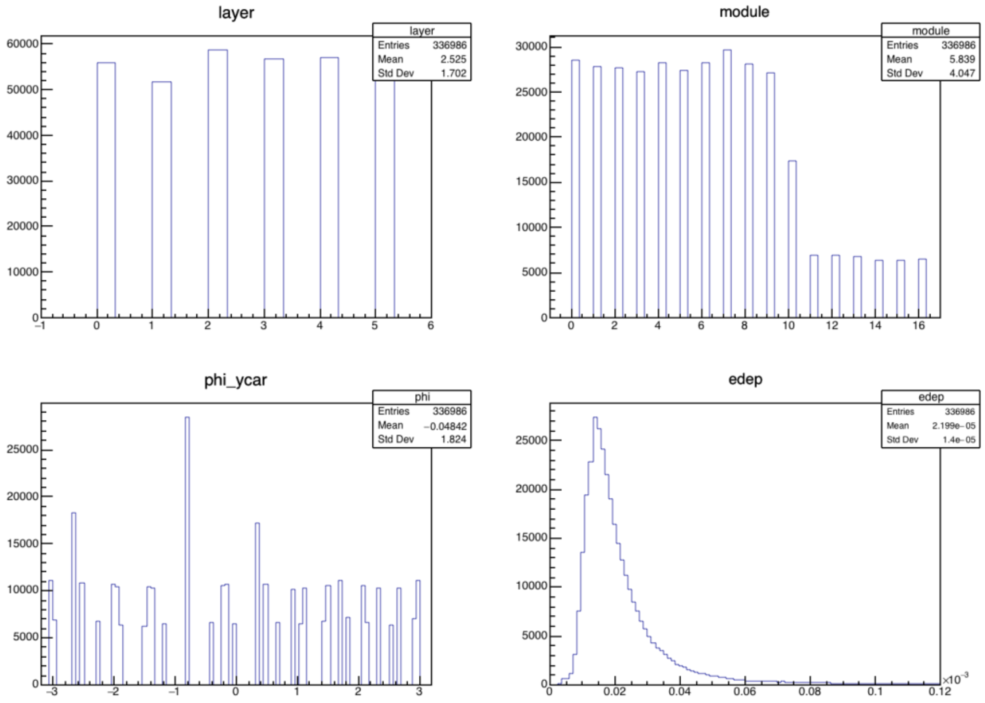
- The results of CEPCSW and Marlin are exactly the same



```
Attaching file PlanarDigi_marlin.root
(TFile *) 0x7fc1ef93ea70
root [1] planarDigi->Show(13000)
======> EVENT:13000
 side                = 0
 layer               = 3
 module              = 8
 sensor              = 0
 theta_xcar          = 1.5708
 phi_xcar            = 1.428
 theta_ycar          = 8.65927e-17
 phi_ycar            = -2.49899
 edep                = 4.89194e-05
```



```
Attaching file PlanarDigi_gaudi.root
(TFile *) 0x7f9b39d043d0
root [1] planarDigi->Show(13000)
======> EVENT:13000
 side                = 0
 layer               = 3
 module              = 8
 sensor              = 0
 theta_xcar          = 1.5708
 phi_xcar            = 1.428
 theta_ycar          = 8.65927e-17
 phi_ycar            = -2.49899
 edep                = 4.89194e-05
```



Future Plans

- Software migration from Marlin to CEPCSW
  - Existing algorithms (reconstruction)
  - Geometry management: GEAR -> DD4hep
  - Recover the relations between PLCIO data object
  - Common services, such as database accessing
- Parallel Computing
  - Use the latest version of Gaudi
  - Writing functional and reentrant algorithms
  - EDM & I/O performance analysis and optimization
- Integration with Deep Learning algorithms
- Package management with SPACK, fewer external libs

Summary

- CEPCSW prototype has been developed using Gaudi, DD4hep, Geant4 and PLCIO, etc.
- In the prototype
  - Both detector simulation and tracking algs can be run successfully
  - By implementing data conversion, previously produced MC data can be reused
- It is ready to add more algorithms to the prototype by following given examples
- Future development will be based on KEY4hep collaborating with CERN

Thank You！

谢谢