Overview of CEPCSW Framework

Tao Lin

(on behalf of CEPC software group)

2020-07-13

CEPCSW Meeting

Contents

- Motivation
- Key4hep based CEPCSW Framework
- DD4hep based Geometry Management
- Example: ECAL design studies with CEPCSW
- Summary

Science at CEPC and SppC

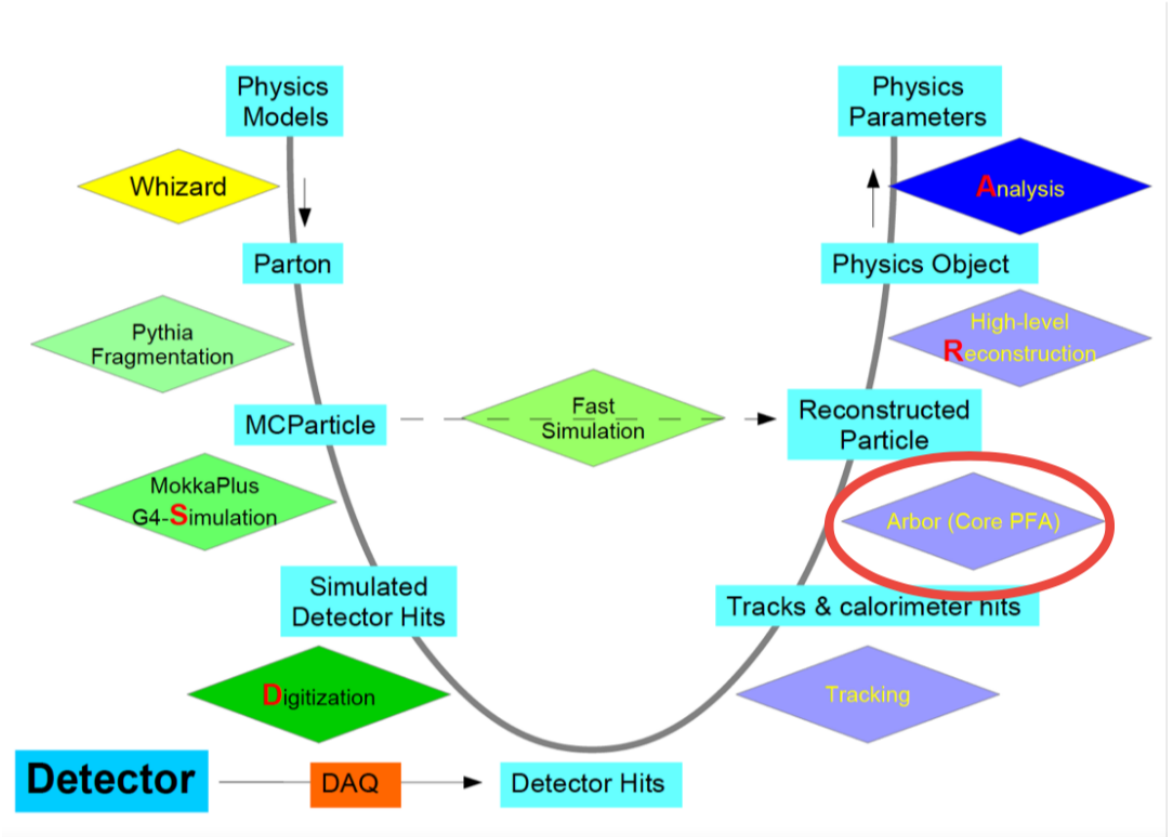
- CEPC (90-250 GeV)
  - Higgs factory: 1M Higgs boson
    - Absolute measurements of Higgs boson width and couplings
    - Searching for exotic Higgs decay modes (New Physics)
  - Z and W factory: 100B-1T Z boson
    - Precision test of the SM
    - Rare decay
  - Flavor factory: b, c, tau and QCD sudies
- SppC (~ 100 TeV)
  - Direct search for new physics
  - Precision test of SM
  - Complementary Higgs measurements to CEPC g(HHH), g(Htt)

Huge Data Volume

- Read out in DAQ from CDR
  - Maximum event rate: ~100 KHz at Z peak
  - Data volume to trigger : ~2 TB/s
- Event size from simulation
  - Size of signal event: ~500 KB/event for Z,
    ~ 1 MB/event for Higgs
  - signal+background: 5~10 MB/event for Z,
    10~20 MB/event for Higgs
- Data storage on disk
  - Higgs/W factory (8 years) with 10^8 events: 1.5~ 3 PB/year
  - Z factory (2 years) with 10^{11} ~10^{12} events: 0.5~ 5 EB/year

ILCSoft-based CEPC Software

From Manqi



CEPC-SIMU-2017-001,
CEPC-SIMU-2017-002,
(DocDB id-167, 168, 173)

- Generators (Whizard & Pythia)
- Data format & management (LCIO & Marlin)
- Simulation (MokkaC)
- Digitizations
- Tracking
- PFA (Arbor)
- Single Particle Physics Objects Finder (LICH)
- Composed object finder (Coral)
- Tau finder
- Jet Clustering (FastJet)
- Jet Flavor Tagging (LCFIPLus)
- Event Display (Druid)
- General Analysis Framework (FSClasser)
- Fast Simulation (Delphes + FSClasser)

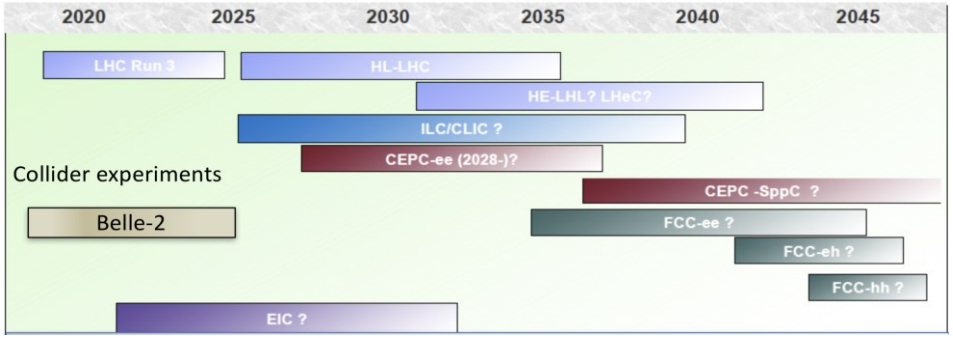
General Software | ILCSoft | ILCSoft + Development | Developments

2

Played an important role for the entire CDR Study

Timeline

- Now entering TDR stage
  - Accelerator
  - Detector
  - Software



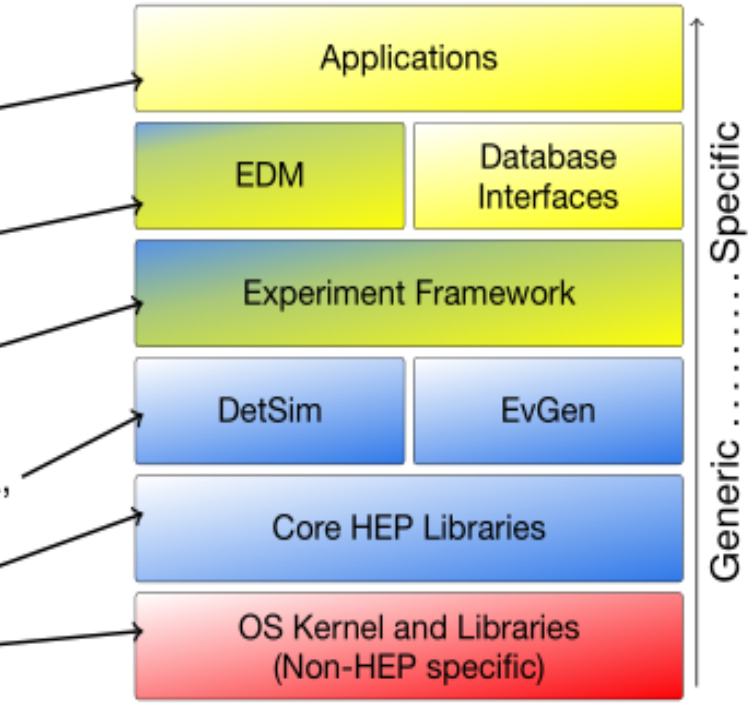
- A new framework is considered to meet the future requirements (at the Oxford workshop, April 2019)
  - Detector performance study: comparison of multiple detector designs and physics performance studies.
  - New software technologies: modern software engineering, parallel computing, machine learning etc.
- Future Collider Software Workshop (June 2019)
  - CEPC, FCC, CLIC, ILC, SCTF
  - => Common software stack (Key4hep)

Key4hep: a common HEP software stack

- Andre Sailer etc, CHEP 2019
  https://indico.cern.ch/event/773049/contributions/3474763/attachments/1938664/3213633/191105_sailer_key4hep.pdf

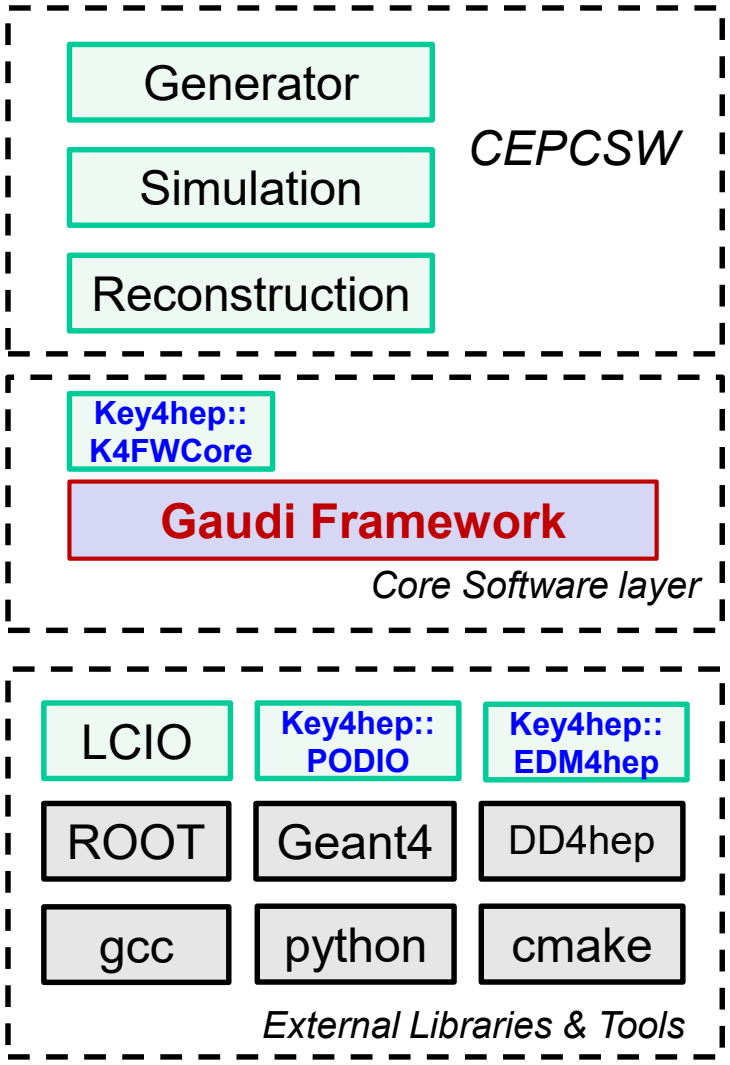
Applications usually rely on large number of libraries, where some depend on others

- 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 ...



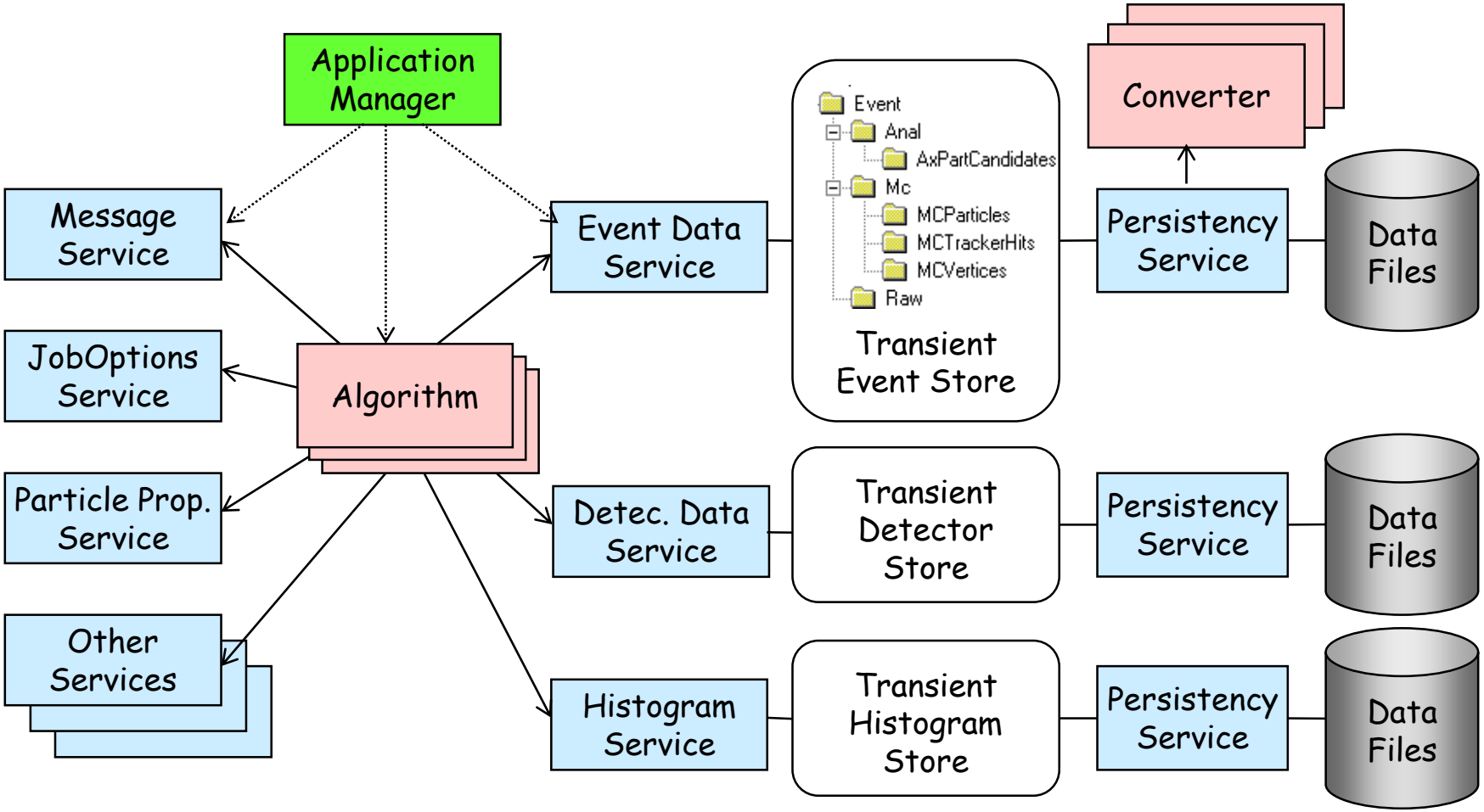
CEPCSW is based on Key4hep

- Common tools
  - CMake: building & deployment
    - Gaudi cmake macros
  - Spack: Package manager
    - K4spack: https://github.com/key4hep/k4-spack
  - Git: version control
    - https://github.com/cepc/CEPCSW
  - CVMFS: software distribution
    - CEPC specific: /cvmfs/cepcsw.ihep.ac.cn/prototype
- Layered External Libraries
  - CEPC specific libraries
  - Key4hep libraries
  - LCG libraries (from CERN CVMFS)



Gaudi Framework

- Developed by LHCb, became CERN standalone project.
- BESIII and Daya Bay experiments used Gaudi.

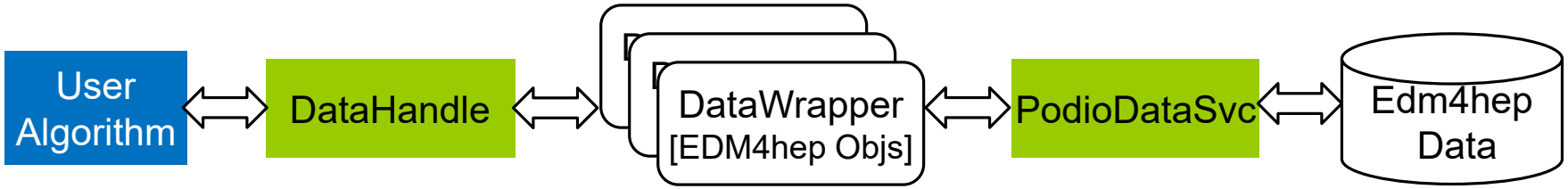


Features of Gaudi Framework

- Application manager: the job controller
  - Creation, configuration and management of services and algorithms
  - Algorithm scheduling during the event loop
  - Terminating the job properly
- Rich 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

K4FWCore

- K4FWCore contains core Gaudi components for Key4hep.
  - Originally developed by FCCSW, then adopted by Key4hep project.
- Data Wrapper and Data Handler
  - User interface to put and get data in Gaudi TES (Transient Event Store)
- Basic ROOT I/O and data management
  - PODIO based Data service
  - PodioInput: algorithm to read data from input file(s) on disk.
  - PodioOutput: Algorithm to write data to output file on disk.



Github: https://github.com/key4hep/K4FWCore

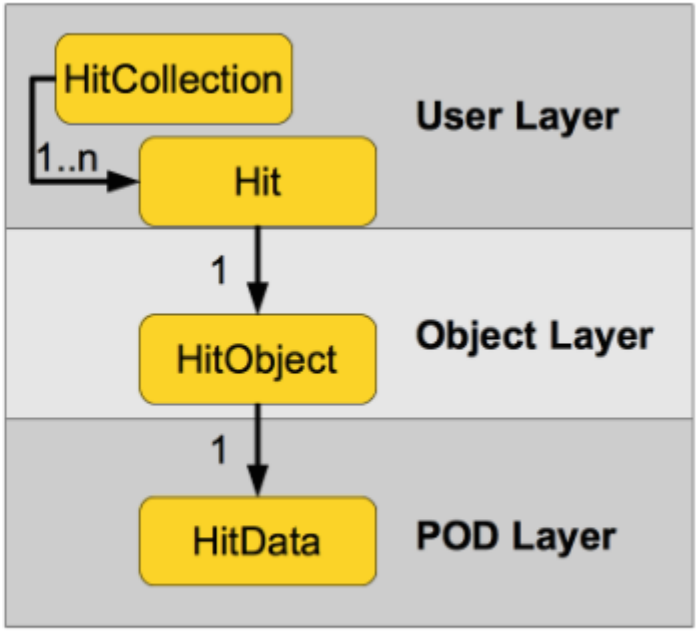
PODIO: an Event-Data Model toolkit

- Generate C++ code automatically from YAML files.
  - Support analysis in ROOT and Python.

- 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**)

POD: Plain Old Data.

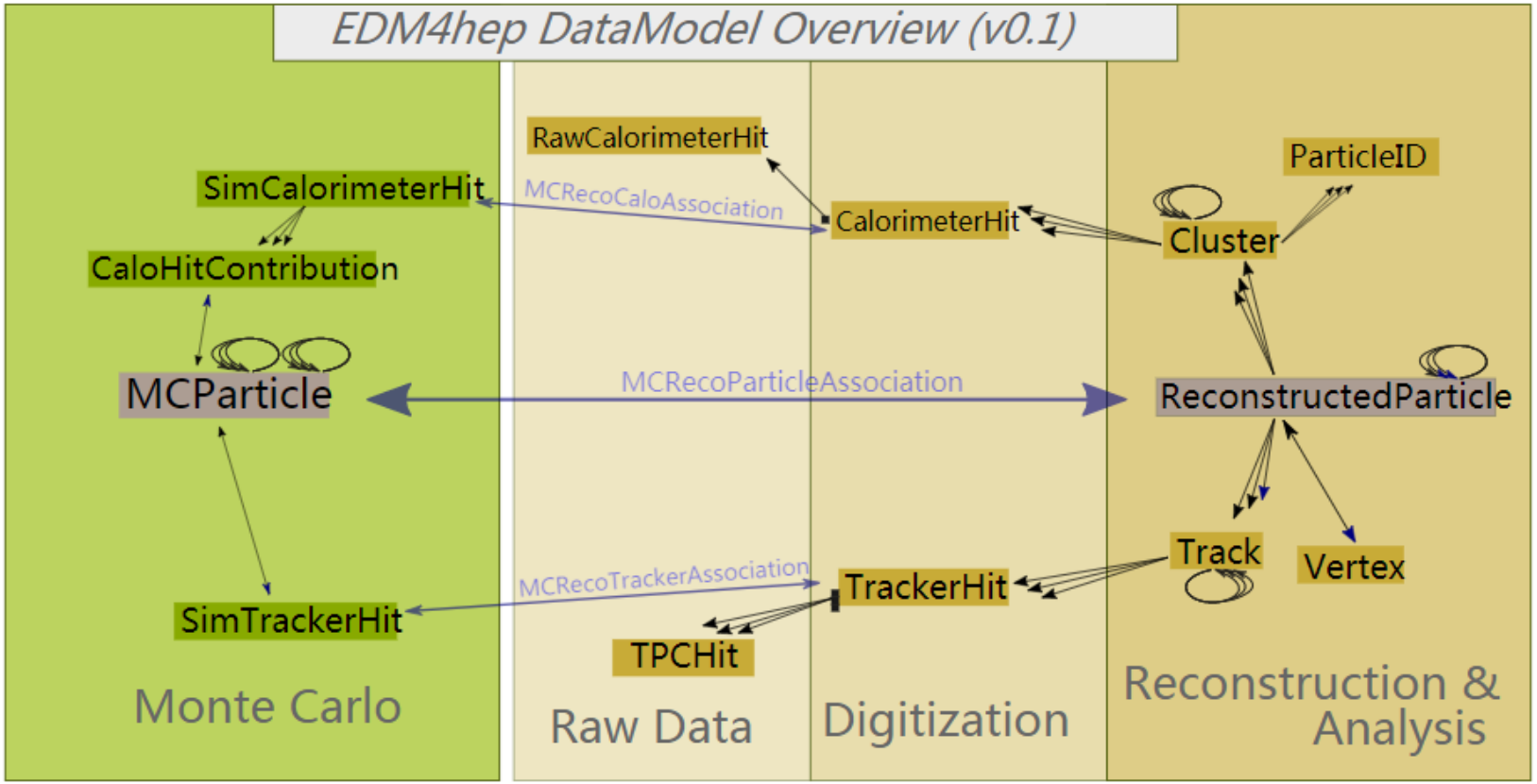
[Ref]: F. Gaede, etc. , CHEP2019



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

EDM4hep

- EDM4hep: official and common EDM in Key4HEP
  - The code is generated by PODIO.
  - The first version (v0.1) has been released recently

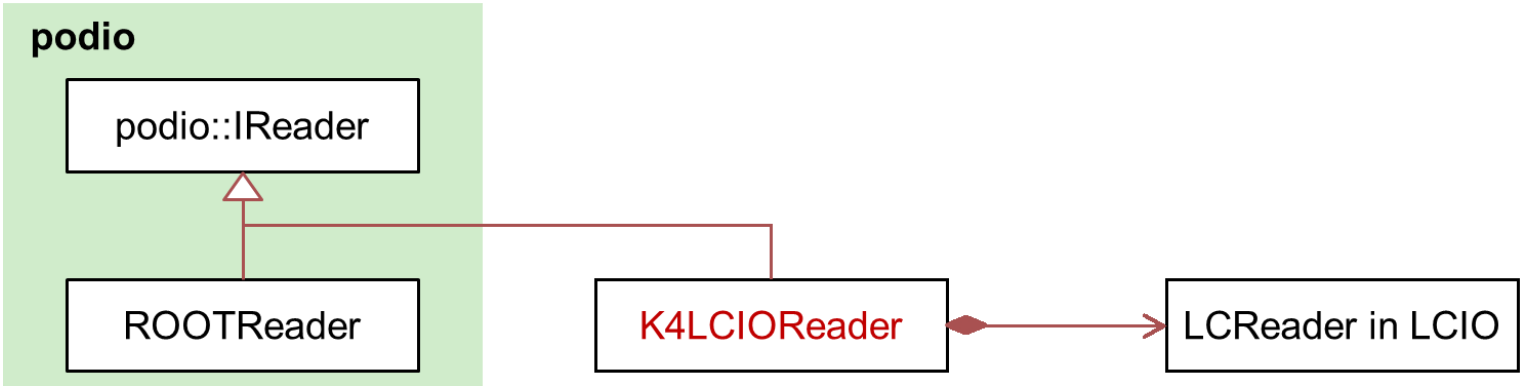


Github repository: https://github.com/key4hep/EDM4hep

EDM Converters: Reading LCIO Data

Jiaheng Zou

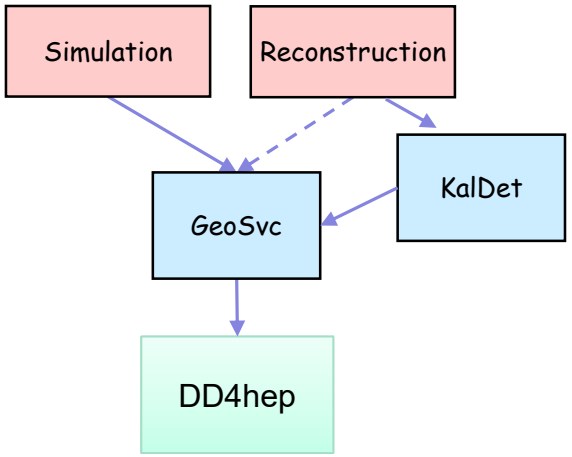
- K4LCIOReader: https://github.com/ihep-sft-group/K4LCIOReader
  - Generate EDM4hep data collections from LCIO format data
  - A standalone package that can be used without Gaudi

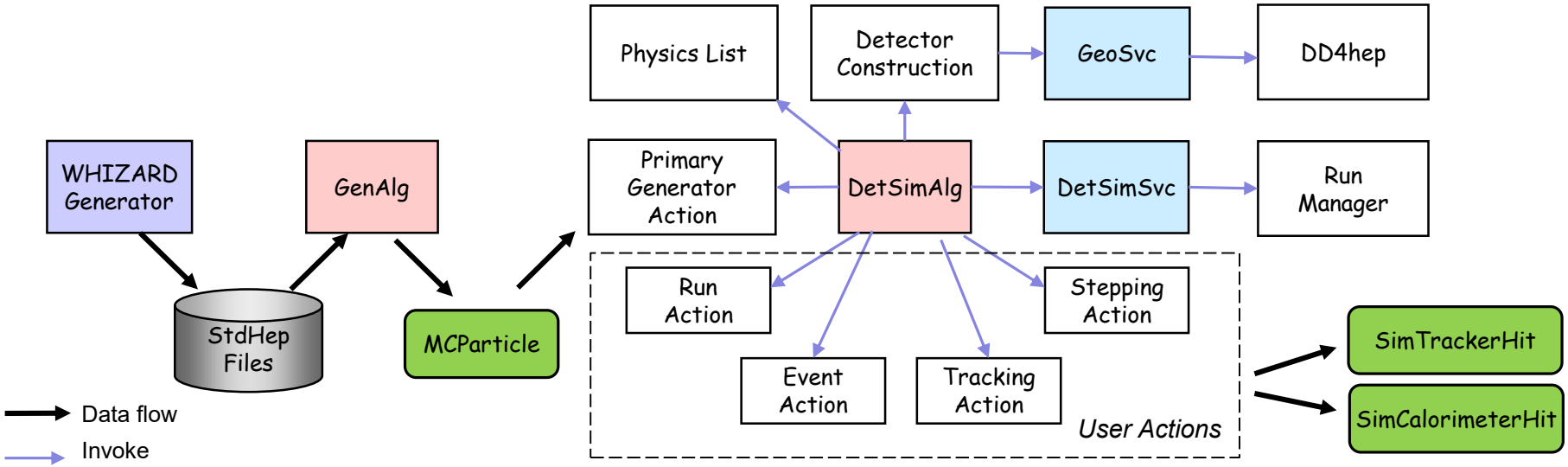


- LCIOInput: https://github.com/ihep-sft-group/LCIOInput
  - A Gaudi algorithm wrapper of K4LCIOReader
  - Read LCIO data in Key4HEP modules

Geometry management (1)

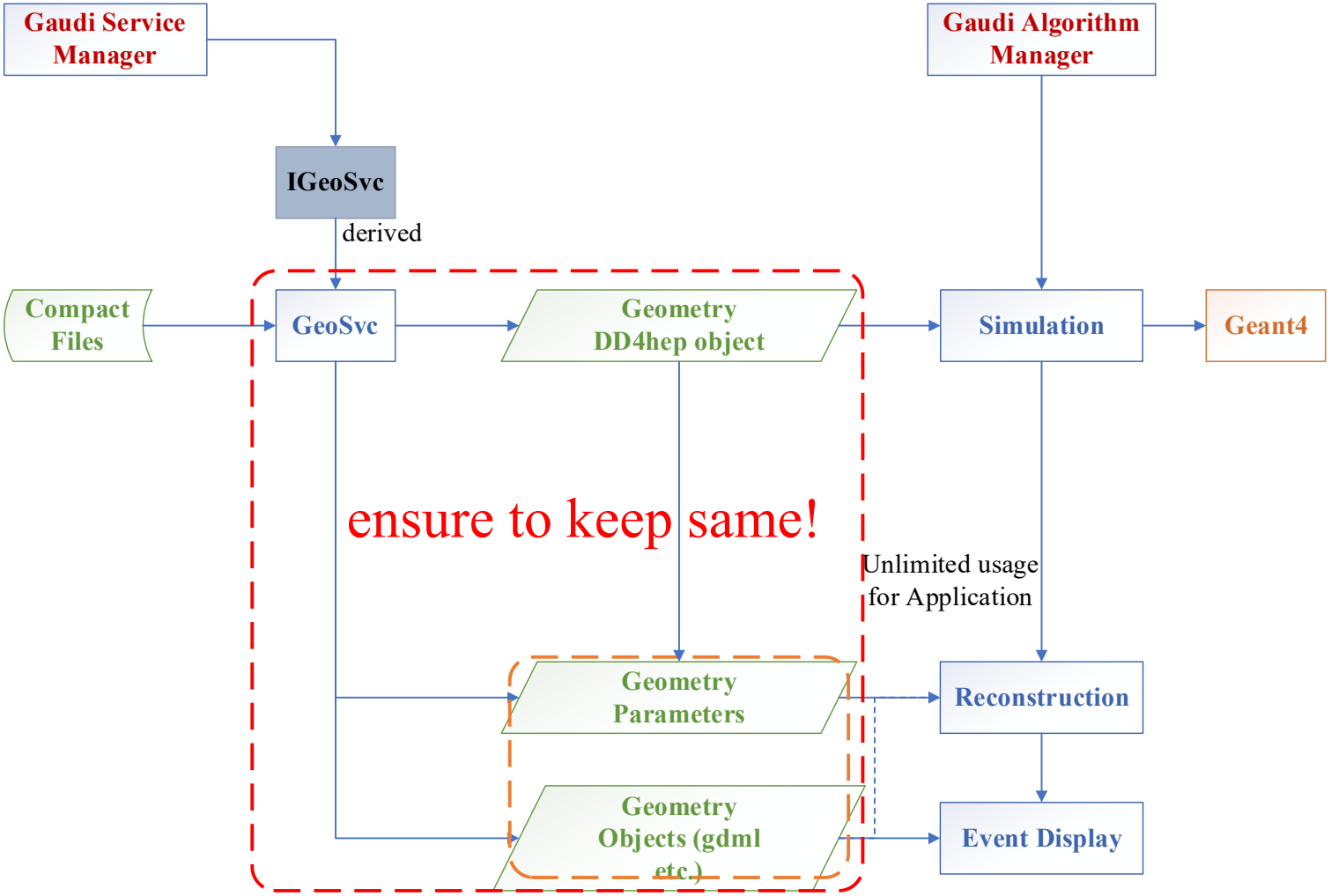
- DD4hep based detector description.
- Unified Geometry Service
  - Interfaced to DD4HEP
  - Used by simulation and reconstruction
- Simulation tool
  - Integrated with physics generator & Geant4





Geometry data management (2)

Chengdong Fu



ECAL design studies with CEPCSW

- By using DD4hep, it is easy to simulate different detector options in the same software framework.
- An example: BGO Crystal ECAL Matrix (based on Chunxiu Liu's)
  - 3D BGO array with 60x60x60 cells. Each cell is $1cm^3$
- A package Detector/DetEcalMatrix was developed.
  - Detector construction by DD4hep driver EcalMatrix.
  - All the parameters are defined in XML files.
  - The readout size is controlled using the segmentation.
  - The cell ID is associated with the segmentation by DD4hep.
- Output format: EDM4hep

```
<detectors>
  <detector id="1" name="CaloDetector" type="EcalMatrix"
            readout="CaloHitsCollection" vis="VisibleGreen"
            sensitive="true">
    <position x="0"  y="0"  z="1835*mm+30*cm"/>
    <dimensions dx="30*cm"  dy="30*cm"  dz="30*cm"/>
  </detector>
</detectors>

<readouts>
  <readout name="CaloHitsCollection">
    <segmentation type="CartesianGridXYZ"
                  grid_size_x="1*cm"
                  grid_size_y="1*cm"
                  grid_size_z="1*cm"/>
    <id>system:8,x:32:-6,y:-6,z:-6</id>
  </readout>
</readouts>
```

A customized driver is developed, then the parameters are controlled in the XML file.

Summary

- To meet the future requirements, a new software is developed.
- Key4hep is the common software stack for future colliders.
- The prototype of CEPCSW is developed based on Key4hep
  - Framework: Gaudi, K4FWCore
  - Event data model: EDM4hep, PODIO
  - Geometry management: DD4hep
- CEPCSW is open source
  - https://github.com/cepc/CEPCSW
  - Welcome your contributions!