HANDLING FIELDS IN KEY4HEP

Juraj Smieško

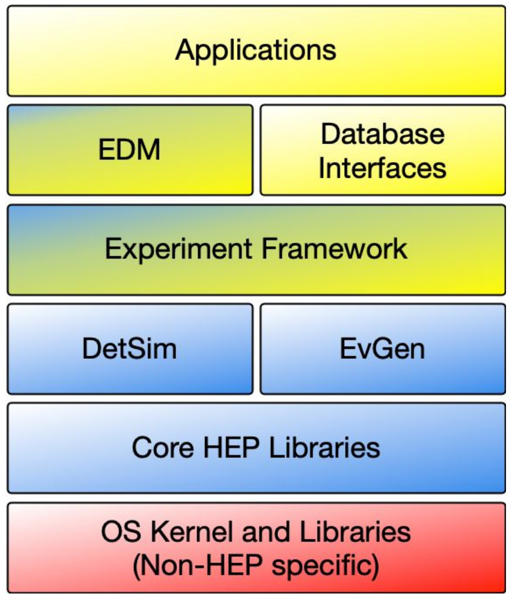
CERN

FCC Detector Full Sim Working Meeting

CERN, 26 Jun 2023

KEY4HEP

- Set of common software packages, tools, and standards for different Detector concepts
- Common for FCC, CLIC/ILC, CEPC, EIC, …
- Individual participants can mix and match their stack
- Main ingredients:
    - Data processing framework: Gaudi
    - Event data model: EDM4hep
    - Detector description: DD4hep
    - Software distribution: Spack



Applications

EDM

Database Interfaces

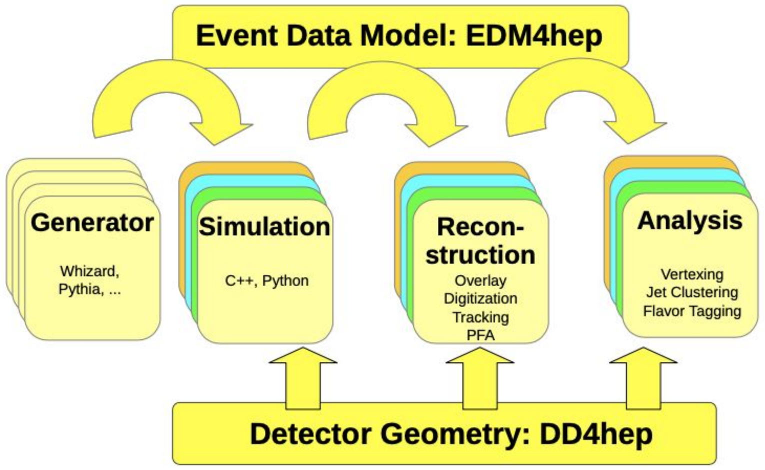
Experiment Framework

DetSim

EvGen

Core HEP Libraries

OS Kernel and Libraries (Non-HEP specific)



Event Data Model: EDM4hep

Generator

Whizard, Pythia, ...

Simulation

C++, Python

Reconstruction

Overlay
Digitization
Tracking
PFA

Analysis

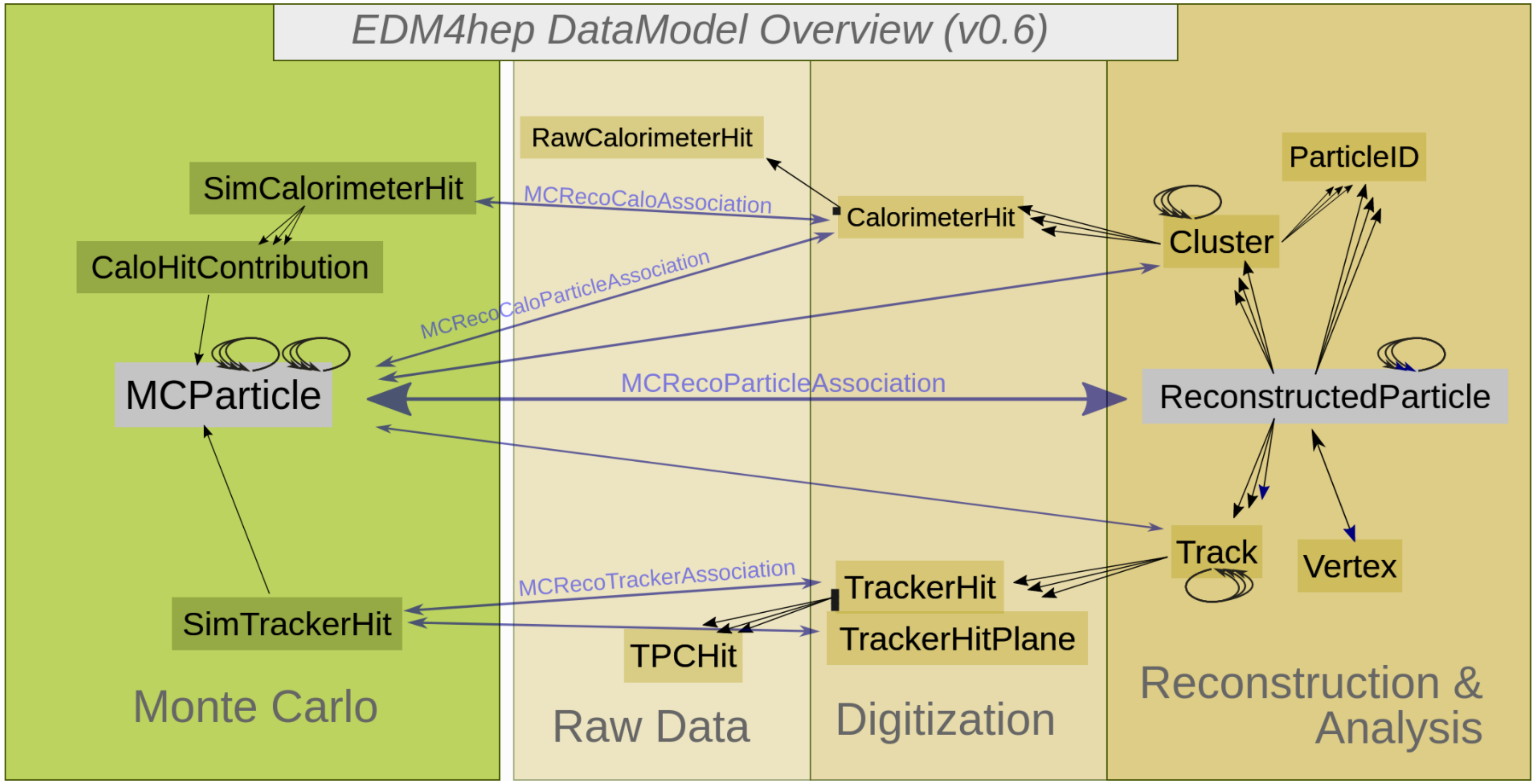
Vertexing
Jet Clustering
Flavor Tagging

Detector Geometry: DD4hep

EDM4HEP I.

Describes event data with the set of standard objects.

- Specification in a single YAML file
- Generated with the help of Podio



EDM4HEP II.

Example object:

```
#-------------  CalorimeterHit
edm4hep::CalorimeterHit:
  Description: "Calorimeter hit"
  Author : "F.Gaede, DESY"
  Members:
    - uint64_t cellID            //detector specific (geometrical) cell id.
    - float energy               //energy of the hit in [GeV].
    - float energyError          //error of the hit energy in [GeV].
    - float time                 //time of the hit in [ns].
    - edm4hep::Vector3f position //position of the hit in world coordinates in [mm].
    - int32_t type               //type of hit. Mapping of integer types to names via coll
```

- Current version: `v0.8.0`
- Objects can be extended / new created
- Bi-weekly discussion: Indico

FULLSIM IN KEY4HEP

DDsim

- Part of the DD4hep
- Used to simulate CLD
- Steering with Python script

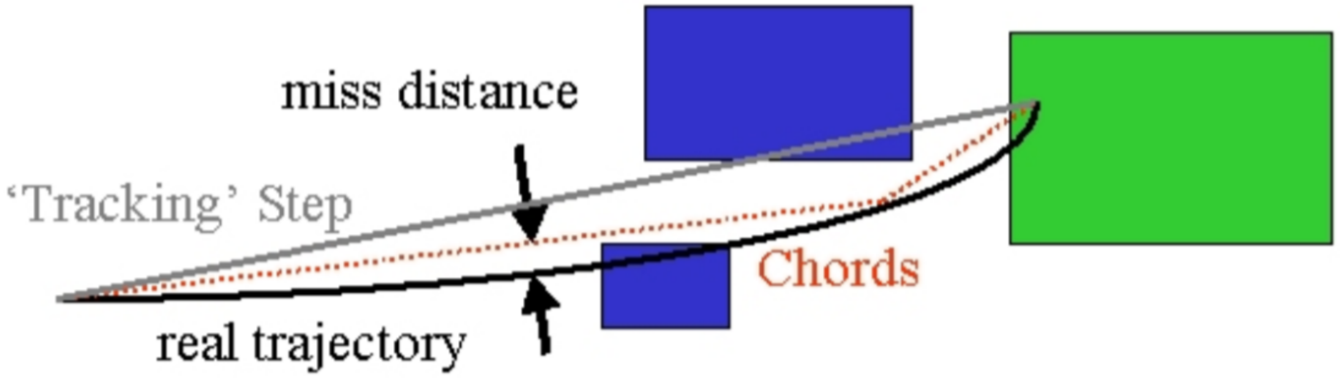
k4SimGeant4

- Set of Gaudi algorithms/tools
- Used to simulate FCC LAr
- Can be part of the larger steering

Both simulations output EDM4hep format, but there are some minor differences

FIELDS IN GEANT4

- Geant4 can propagate particle through magnetic, electric, electromagnetic and gravitational fields
- The tracking can be done to arbitrary accuracy
- Equation of motion of the particle in the field is integrated usually by Runge-Kutta method
- There are several method implementations, suitable for different conditions
- Inside one step, the path is broken up into small segments: chords
- The magnetic field is managed by `G4FieldManager`
- User needs to implement `G4Field` method: `GetFieldValue(const double Point[4], double *fieldArr)`



FIELD IN DD4HEP COMPACT FILE

- Electric or magnetic field(s) described in `field(s)` tag
- DD4hep creates combined field: `OverlayedField`
- Constant Electric or Magnetic Fields are defined as follows:

```
<field name="MyMagnet" type="ConstantField" field="electric">
  <strength x="x-val" y="y-val" z="z-val">
</strength></field>
```

- Magnetic Dipoles are defined as follows:

```
<field name="MyMagnet" type="DipoleMagnet" rmax="50*cm" zmin="0*cm" zmax=
  <dipole_coeff>1.0*tesla</dipole_coeff>
  <dipole_coeff>0.1*tesla/pow(cm,1)</dipole_coeff>
  <dipole_coeff>0.01*tesla/pow(cm,2)</dipole_coeff>
</field>
```

- Other notable field types: `solenoid`, `FieldXYZ`

DDSIM

- Simulation runs field(s) provided in the `field` tag
  - Constant, solenoid, 3D Fieldmap, ...
- Integration parameters can be provided in the Python steering:

```
########################################################################
## Configuration for the magnetic field (stepper)
########################################################################
SIM.field.delta_chord = 0.25*mm
SIM.field.delta_intersection = 0.001*mm
SIM.field.delta_one_step = 0.01*mm
SIM.field.eps_max = 0.001*mm
SIM.field.eps_min = 5e-05*mm
SIM.field.equation = "Mag_UsualEqRhs"
SIM.field.largest_step = 10.0*m
SIM.field.min_chord_step = 0.01*mm
SIM.field.stepper = "ClassicalRK4"
```

Example taken from CLICPerformance

K4SIMGEANT4

- Simulation service `SimG4Svc` needs mag. field tool
  - Interface: `ISimG4MagneticFieldTool`
- Three tools implemented:
  - `SimG4ConstantMagneticFieldTool`: Constant field in barrel
  - `SimG4MagneticFieldFromMapTool`: 2D Comsol map, 3D map from ROOT file
  - `SimG4MagneticFieldTool` (PR #37): Propagates field defined in compact file
- Example of constant mag field:

```
from Configurables import SimG4ConstantMagneticFieldTool
field = SimG4ConstantMagneticFieldTool("SimG4ConstantMagneticFieldTool")
field.FieldComponentZ = -2 * units.tesla
field.FieldOn = True
field.IntegratorStepper="ClassicalRK4"
```

Example taken from k4RecCalorimeter

CONCLUSIONS

- With PR #37 both simulation methods can run with various fields
- DD4hep offers ability to specify mag. field alongside the other detector "parts" in the compact file
- Tool approach of k4SimGeant4 allows to change the field in the steering
- Remark: Field maps interface is not well defined

BACKUP

MAGFIELDSCANNER

- With PR #37 there is now possibility to probe your magnetic field with `MagFieldScanner`
- There are three probe types:
  - XYPlane: shows mag. field on the XY plane at any z
  - ZPlane: shows mag. field on the plane lying on z-axis at any phi (angle from x-axis)
  - Tube: shows mag. field on a tube with radius r
- Example result: Mag. field: constant B_z = -2T, rMax = 150cm, abs(zMax) = 20m