FCC Core Software: New Functionalities

Juan Miguel Carceller

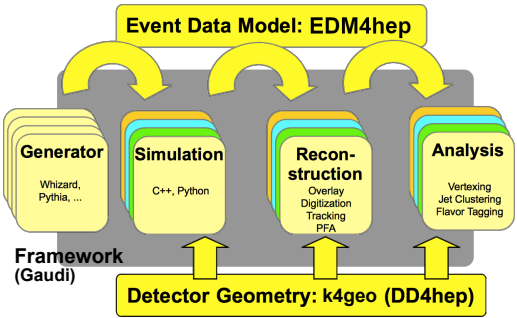
CERN

June 12, 2024

FCC Core Software: Key4hep

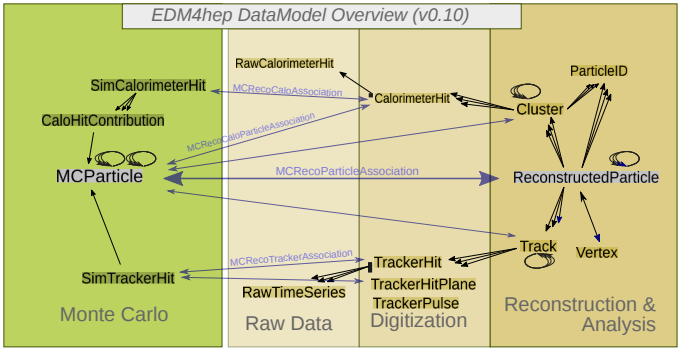
Key4hep

- Turnkey software for future colliders
- **Share components** across different experiments and communities and avoid duplication of efforts
- Complete data processing framework from generation to data analysis
- Community with people from many different experiments: **FCC**, CEPC, CLIC, EIC, ILC and Muon Collider
- Open biweekly talks with all the stakeholders



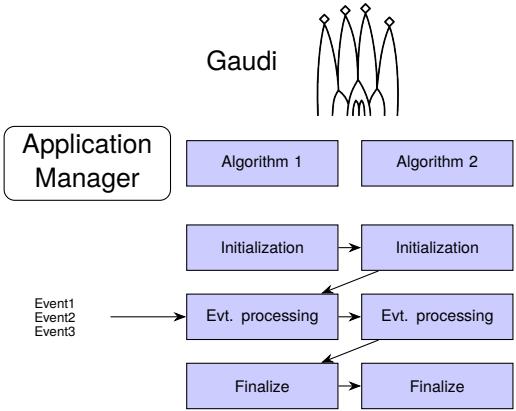
The Key4hep Event Data Model: EDM4hep

- Data Model used in Key4hep, **common language** that all components must speak
- Goals: be **generic** and **address the needs of the experiments**
- Evolves through **consensus** among all stakeholders
- Generated with Podio from a text file



The Key4hep Framework

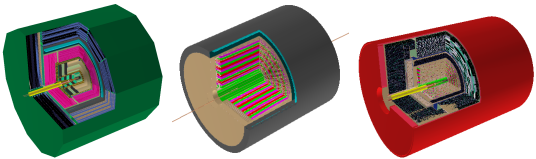
- Gaudi based core framework:
  - k4Gen for integration with generators
  - k4SimDelphes for integration with Delphes
  - k4FWCore provides the interface between EDM4hep and Gaudi
  - k4MarlinWrapper to call any Marlin processor
  - ...



- Used by LHCb, ATLAS, Key4hep and others

DD4hep

- DD4hep provides a way to build a detector description by using XML compact files
- Interfaces Geant4 and can run physics simulations with `ddsim`
- Plugin-based system
- DD4hep is the recommended toolkit for Detector Description in FCC
- Well integrated in Key4hep
- Detector models and geometries are stored in k4geo, many migrated recently from FCCDetectors



FCC-ee Detector Benchmarks:
CLD, IDEA and ALLEGRO

The Key4hep Stack

- Software provided in *stacks* deployed on cvmfs
- More than 500 packages built with Spack
- Support for Alma 9, Ubuntu 22.04 and CentOS 7
- Releases in `/cvmfs/sw.hsf.org` with tagged versions of the packages
- Nightly builds in `/cvmfs/sw-nightlies.hsf.org` with the latest version of the Key4hep and FCC packages and other packages
- Easy setup with cvmfs:

```
source /cvmfs/sw.hsf.org/key4hep/releases/setup.sh              # Latest release
source /cvmfs/sw-nightlies.hsf.org/key4hep/releases/setup.sh    # Latest nightly
```

- Questions, problems, complaints and anything else related to packages happen mostly in https://github.com/key4hep/key4hep-spack

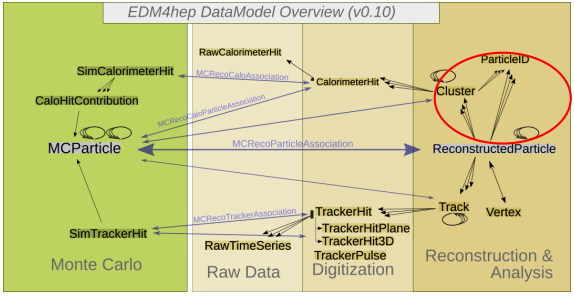
New developments

EDM4hep

- Several changes in EDM4hep before **EDM4hep 1.0**
- Some of them are not backwards compatible: files produced with the last release won't be readable in the future (they are not readable with the nightlies already)
  - Translation studied and feasible (from files produced with pre 1.0 to 1.0), ready once EDM4hep 1.0 is there
- After 1.0 aim to be backwards compatible, thanks to schema evolution in Podio
- Warning: Files created with the nightlies may not be readable in future nightlies and/or releases!

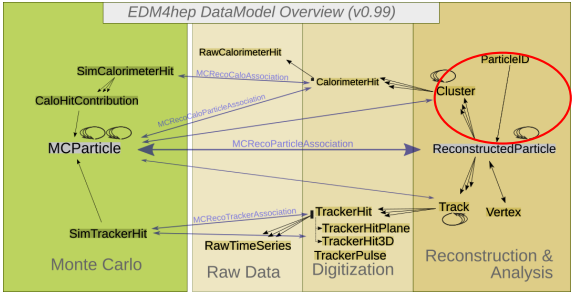
EDM4hep changes: Reversal of PID relations

- Before a Reconstructed Particle will point to N ParticleIDs
- **Motivation**: When having multiple PID algorithms, the reconstructed particles are ”fixed” and can’t be changed, so to add more ParticleIDs they would have to be copied
- After this change, ParticleID points to the Reconstructed Particle it identifies



EDM4hep changes: Reversal of PID relations

- Before a Reconstructed Particle will point to N ParticleIDs
- **Motivation**: When having multiple PID algorithms, the reconstructed particles are ”fixed” and can’t be changed, so to add more ParticleIDs they would have to be copied
- After this change, ParticleID points to the Reconstructed Particle it identifies



EDM4hep: TrackerHit interface

- We have different types of hits, for example `TrackerHit3D` and `TrackerHitPlane`
- **Motivation**: allow using an interface to any type of hit. Tracks now have a relation to the interface

```yaml
interfaces:
  edm4hep::TrackerHit:
    Description: "Tracker hit interface class"
    Author: "Thomas Madlener, DESY"
    Members:
      - uint64_t cellID
      - int32_t type
      - int32_t quality
      - float time [ns]
      - float eDep [GeV]
      - float eDepError [GeV]
      - edm4hep::Vector3d position [mm]
    Types:
      - edm4hep::TrackerHit3D
      - edm4hep::TrackerHitPlane
```

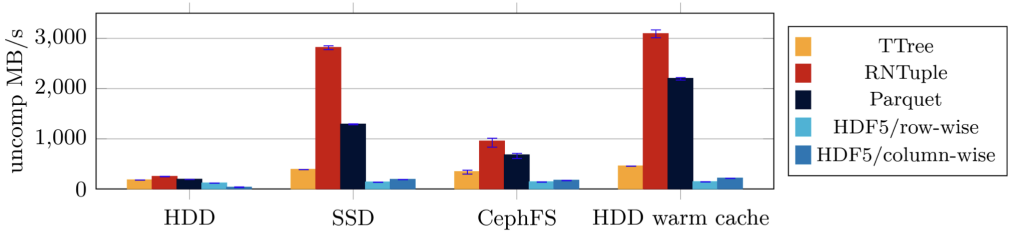
- Doesn't change saved hits, they are either `TrackerHit3D` or `TrackerHitPlane`

Other changes in EDM4hep

- Dedicated covariance matrices
  - `edm4hep::CovNf` with $N = 2, 3, 4, 6$ instead of having a `std::array<N, float>` member
  - Utilities to make working with covariance matrices easier
- Change the momentum in MCParticles from float to double
- Remove `edm4hep::ObjectID`
  - Unnecessary and unused in practice
- Ongoing changes:
  - Add MC information: new types `GeneratorEventParameters` and `GeneratorPdfInfo`

IO: RNTuple

- New format to be used instead of TTree
- Significantly less space usage than TTree and better IO throughput depending on the task
- File-based and object storage
- 20% savings for ATLAS production



IO: Reading and Writing

- Recently, a general Reader and Writer that work for any supported input format (including RNTuples) have been added
- Aware of the different backends and will be able to determine which one to choose for reading and can be easily selected for writing

Reading

```
auto reader = podio::makeReader("example.root");
auto frame = reader.readNextFrame(podio::Category::Events);
auto coll = frame.get("MCParticles");
```

Writing

```
// Assume we have a frame called frame
auto writer = podio::makeWriter("example.root");
frameWriter.writeFrame(frame, podio::Category::Event);
```

- Ongoing work to add this functionality to Gaudi algorithms to make changing between TTrees and RNTuples easy

Gaudi Algorithms

- Support for `Gaudi::Functional`
  - Functional algorithms do not have an internal state and are suitable to run multithreaded
- New service added, `IOSvc` that allows one to easily switch to multithreading
- To make use of these features, existing algorithms have to move to `Gaudi::Functional` and use `IOSvc`

Gaudi Algorithms

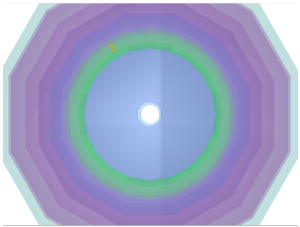
- Example: complete algorithm that takes as input `MCParticles`

```
struct ExampleFunctionalConsumer final : k4FWCore::Consumer<void(const edm4hep::MCParticleCollection& input)> {
  ExampleFunctionalConsumer(const std::string& name, ISvcLocator* svcLoc)
      : Consumer(name, svcLoc, KeyValues("InputCollection", {"MCParticles"})) {}

  void operator()(const edm4hep::MCParticleCollection& input) const override {
    if (input.size() != 2) {
      fatal() << "Wrong size of MCParticle collection, expected 2 got " << input.size() << endmsg;
      throw std::runtime_error("Wrong size of MCParticle collection");
    }
  }
};
```

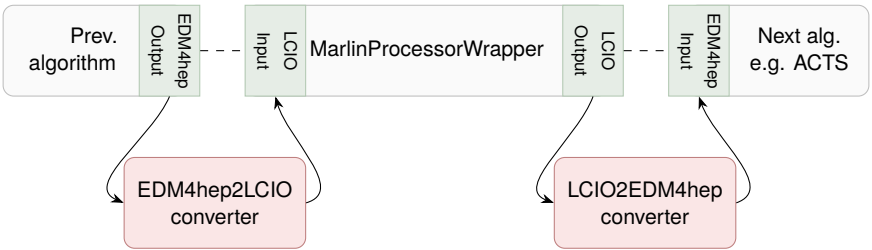
Integrations and Native Algorithms

- There is ongoing work on either native tools or integration with other tools
- Native means that the algorithms use EDM4hep and play well with Key4hep
- Examples
  - ACTS in Key4hep (L. Reichenbach)
  - Pandora Particle Flow Algorithm in Key4hep (S. Sasikumar)
  - Example of digitisation algorithm ported from iLCSoft
  - A background overlay algorithm is being tested



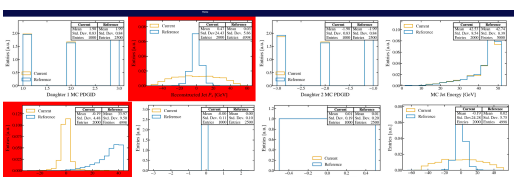
LCIO Converters

- There has been an overhaul of the EDM4hep - LCIO converters
- Marlin processors can be used in Gaudi using the `MarlinProcessorWrapper`
- EDM4hep input can be used seamlessly in processors taking LCIO input and giving LCIO output
- Standalone converter `lcio2edm4hep` to convert files



Key4hep Validation: Simulation and Reconstruction

- Check the simulation and reconstruction chain
- Run daily, use the latest Key4hep nightlies
- Run a simulation with DD4hep, then reconstruction, then analysis scripts and then make plots
- Results are compared to a reference sample
- Plots are deployed to WebEOS
- https://key4hep-validation.web.cern.ch/
- Work in progress, no documentation yet



- It would be good to add FCC detectors so that they are validated during their development

The Key4hep Stack and CI

- No more support for CentOS 7 in the future: nightlies are not being built anymore and there won't be a next release on CentOS 7
- Compiled with C++20
- Many improvements mostly due to user request: visualizations, different packages missing environment variables or not working, etc.
- Help with development: Running `k4_local_repo` while inside a package will clear the environment of that package and add environment variables pointing to the local version
- CI: Fast builds on top of the releases and nightlies when making a PR in the Key4hep repos

Summary

- The Key4hep team supports FCC studies and people
- Moving towards **EDM4hep 1.0**
  - Consolidate EDM4hep
  - After 1.0 backwards compatibility
- Lots of progress in Key4hep in different areas
  - Algorithms and native algorithms for full sim
- The centralized effort done in Key4hep benefits both the FCC community and other experiments

Acknowledgements

- CERN Strategic R&D Programme on Technologies for Future Experiments (CERN-OPEN-2018-006)
- European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 101004761.

Backup

RNTuple in Podio (and EDM4hep)

- Support for RNTuple has been added in podio with `RNTupleWriter` and `RNTupleReader`
- Similar interface to the TTree Reader and Writer

TTree

```
ROOTReader reader{};
reader.openFile(filename)
auto event = podio::Frame(reader.readEntry("events", 0));
```

```
ROOTWriter writer(filename);
writer.writeFrame(frame);
writer.finish();
```

RNTuple

```
RNTupleReader reader{};
reader.openFile(filename)
auto event = podio::Frame(reader.readEntry("events", 0));
```

```
RNTupleWriter writer(filename);
writer.writeFrame(frame);
writer.finish();
```

EDM4hep: Dedicated covariance matrices

- Previously, the covariance matrices were `std::array<N, float>`
- Now, they have their own classes:

```
edm4hep::CovMatrix3f:
  Description: "A generic 3 dimensional covariance matrix with values stored in lower triangular form"
  Members:
    - std::array<float, 6> values // the covariance matrix values
  ExtraCode:
    declaration: "
      constexpr CovMatrix3f() = default;\n
      constexpr CovMatrix3f(const std::array<float, 6>& v) : values(v) {}\n
      template<typename... Vs>\n
      constexpr CovMatrix3f(Vs... v) : values{v...} {}\n
      constexpr CovMatrix3f& operator=(std::array<float, 6>& v) { values = v; return *this; }\n
      bool operator==(const CovMatrix3f& v) const { return v.values == values; }\n
      bool operator!=(const CovMatrix3f& v) const { return v.values != values; }\n
    "
```

- Additional utility functions

Key4hep Builds: Tests

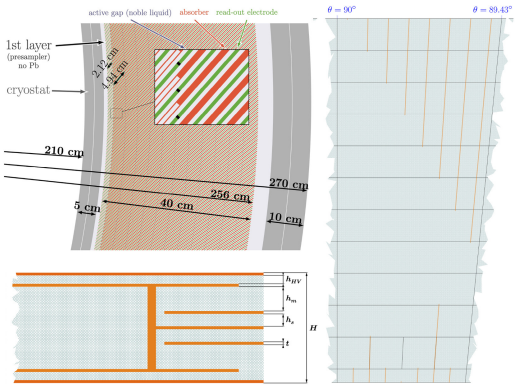
- New usability tests are being added: compilation, ROOT, python, python packages, whizard, key4hep tools
- These tests come from experience, mainly from what people use that one day doesn't work, to make sure it doesn't repeat

```
cat > ee.sin <<EOF
    process ee = e1, E1 => e2, E2
    sqrts = 360 GeV
    n_events = 10
    sample_format = lhef
    simulate (ee)
EOF
run_test "whizard test" "whizard -r ee.sin"
```

Pandora

- Liquid Argon (LAr) detectors are being studied for future experiments (e.g. FCC)
- Swathi will study jet energy resolution for IDEA-LAr when a full simulation for IDEA is implemented

- Currently studying LAr with CLD (with full simulation)
- While adding the LAr calorimeter inside CLD overlaps were found so changes in the geometry had to be done
- Implementing a Gaudi algorithm to use Pandora PFA with Gaudi



Key4hep Validation: Simulation and Reconstruction

- Example of a plot using a release with a bug as the reference one