Key4hep: Current Status and Recent Developments

Juan Miguel Carceller

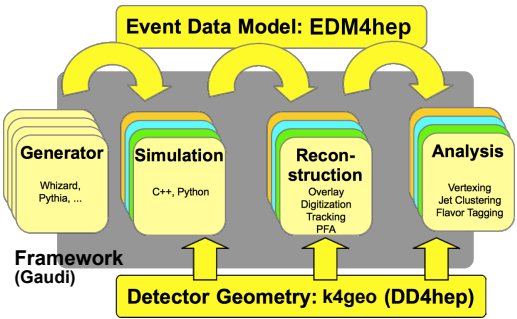
on behalf of the Key4hep team

CERN

January 30, 2024

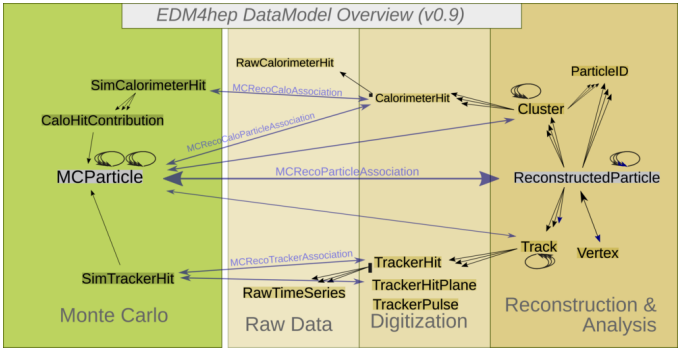
Key4hep

- Turnkey software for future accelerators
- Share components to reduce maintenance and development cost and allow everyone to benefit from its improvements
- Complete data processing framework, from generation to data analysis
- Community with people from many different experiments: **FCC**, C^3, CEPC, CLIC, EIC, ILC, Muon Collider, etc.
- Open biweekly talks with all the stakeholders



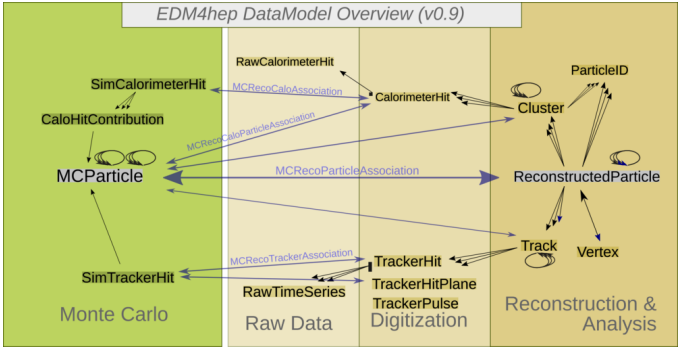
The Key4hep Event Data Model: EDM4hep

- Data Model used in Key4hep, it is the language that all components must speak



The Key4hep Event Data Model: EDM4hep

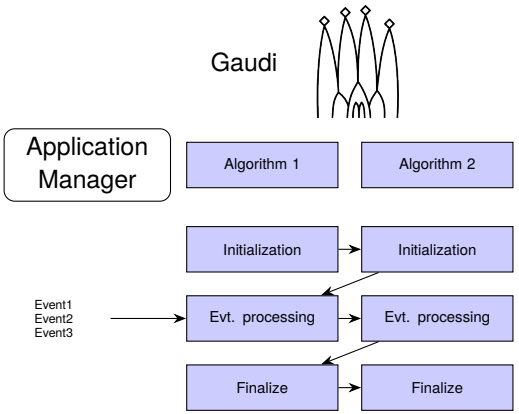
- Data Model used in Key4hep, it is the language that all components must speak



More details in Thomas' talk

The Key4hep Framework

- Gaudi based core framework:
  - k4Gen for integration with generators
  - k4SimGeant4 for integration with Geant4
  - k4SimDelphes for integration with Delphes
  - k4geo for detector models, previously lcgeo
  - k4FWCore provides the interface between EDM4hep and Gaudi
  - k4MarlinWrapper to call Marlin processors
  - …



- Used by LHCb, ATLAS, Key4hep and others

New developments

Frame reading and writing in Key4hep

- The Frame (from podio) is a data container where collections can be stored
- Key feature: support for multithreading
- Typically represents an event but can be anything else
- A backend decides how it is written to a file (ROOT files with ROOT TTrees most of the time)
- Automatic Frame reading and writing was introduced to the Key4hep Gaudi algorithms
  - Previously files were being saved to an Event Store (didn't support multithreading)
  - Now files produced are read and written as Frames
  - Changes were transparent; users didn't need to modify their code

Simple interface with `get` and `put`

```
frame.get("MCParticleCollection");
frame.put(std::move(coll), "NewCollection");
```

Also in python:

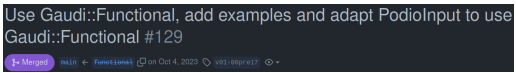
```
from podio.root_io import Reader
reader = Reader('myfile.root')
events = reader.get('events')
for frame in events:
    coll = frame.get('MCParticleCollection')
```

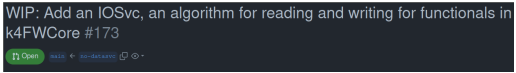
Removal of the EventStore

- The EventStore (deprecated in podio since February 2023) has been removed from podio
- Followed with the removal in k4FWCore
- Using it intentionally in algorithms has been deprecated for a while, components were called `PodioLegacyInput`, `PodioLegacyOutput`, etc.
- It's still in the latest release, but not in the nightlies, no one complained so far

Gaudi Algorithms

- Recent progress and work to use Gaudi Functional algorithms

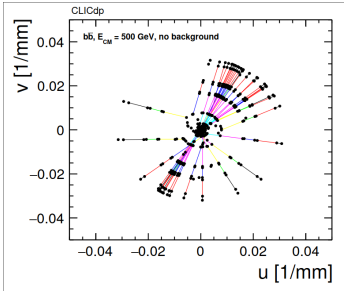


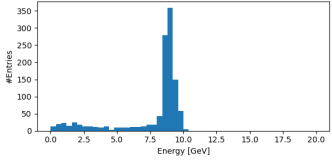


- Functional algorithms do not have an internal state and are suitable to run multithreaded
- Nothing particularly fancy, already in use in ATLAS and LHCb
- Depending on the algorithm only minor modifications may be needed, other algorithms may not easily converted
- To be studied on a case by case basis

Integrations and Native Algorithms

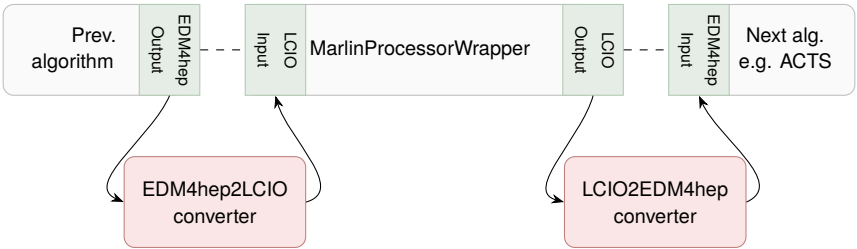
- There is ongoing work on either native tools or integration with other tools
- Native means that the algorithms use EDM4hep and play well with Key4hep
- Examples
  - ACTS in Key4hep (Leonhard on Thu)
  - Pandora Particle Flow Algorithm in Key4hep (Swathi on Thu)
  - Example of digitisation algorithm ported from iLCSoft





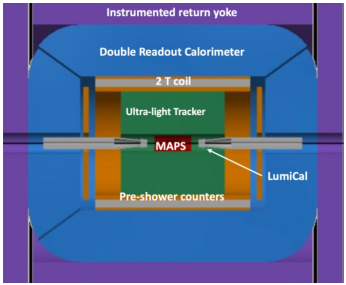
LCIO Converters

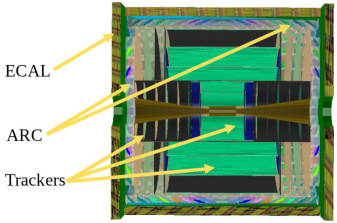
- There has been an overhaul of the EDM4hep - LCIO converters
- Current status
  - Marlin processors can be used in Gaudi using a `MarlinProcessorWrapper`
  - EDM4hep input can be used and not worry about Marlin processors taking LCIO input and giving LCIO output
  - Standalone converter `lcio2edm4hep` to convert files



Detector and Reconstruction studies

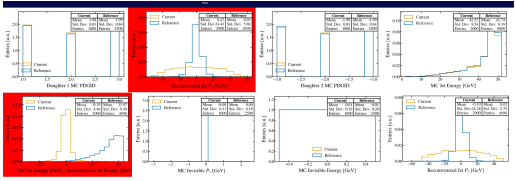
- New detectors have been added recently to k4geo, some of them migrated from FCCDetectors:
  - IDEA
  - ALLEGRO
  - CLD with the ARC subdetector
  - IDEA Vertex Detector
- Segmentations migrated from FCCDetectors





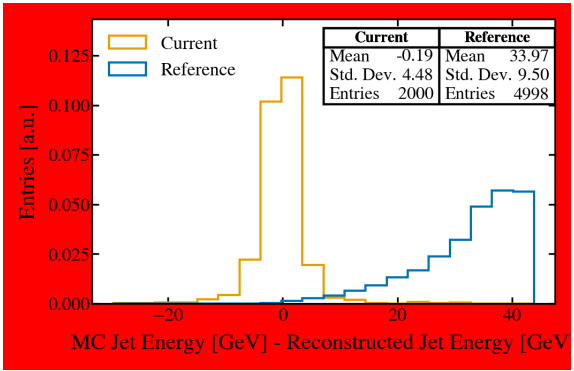
Key4hep Validation: Simulation and Reconstruction

- 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



Key4hep Validation: Simulation and Reconstruction

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



The Key4hep Stack

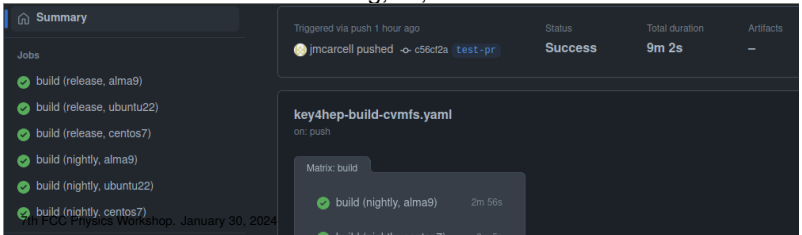
- Software provided in *stacks* deployed on cvmfs
- More than 500 packages (most are dependencies) built with Spack
- Releases in `/cvmfs/sw.hsf.org` with tagged versions of the packages
- Nightly builds in `/cvmfs/sw-nightlies.hsf.org` with the latest version of the Key4hep packages and other packages
- 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 happens mostly in https://github.com/key4hep/key4hep-spack

The Key4hep Stack and CI

- Both the latest release (and future ones) and the nightly builds now support CentOS 7, AlmaLinux 9 and Ubuntu 22.04 (previously it was only CentOS 7)
- Warning: CentOS 7 is close to its end of life in June so we will stop building for it at some point
- Nightly builds are now compiled using the C++20 standard, meaning we could start using C++20 features
- Many improvements mostly due to user request: visualizations, different packages missing environment variables or not working
- Standardization at different levels: formatting, CI, etc.



Summary

- Lots of progress in Key4hep in different areas
- The Key4hep team supports FCC studies and people
- Many more improvements, impossible to list them all!
- More to come, expect more integrations and native algorithms in Key4hep, bug fixes and quality of life improvements
- Very wide community with people from different experiments, small team to support all of this

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

Podio and EDM4hep

- Schema evolution: it is possible to modify our definitions in EDM4hep and still be able to read old data
- New RNTuple backend to write RNTuples (experimental new format for ROOT files)
- Python bindings for EDM4hep

```
import edm4hep
particle = edm4hep.MCParticle() # default initialized particle
particle.getCharge() # 0.0
```

- Together with the podio bindings it is possible to read or generate data and save it to a file, all from python

The Key4hep Framework

- Key4hep provides the interface and the glue to make the different pieces talk to each other
- Components are picked up from other places (for example Gaudi for the event processing framework)
- iLCSoft can be used thanks to the k4MarlinWrapper, we benefit from lots of years of development and tested software
- Results on detector studies, analyses, etc

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"
```

podio: RNTuple backend

TTree based

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

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

RNTuple based

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

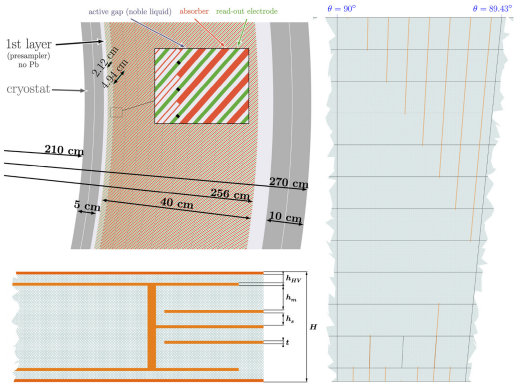
```
ROOTNTupleReader reader{};
reader.openFile(filename)
auto event = podio::Frame(reader.readEntry("events", 0));
```

- For the future:
  - Comparisons between the RNTuple and TTree-based backends: reading and writing speed, file size
  - Python bindings for the RNTuple writer and reader

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



New Workflow for Simulation at FCC

- `ddsim` was chosen for simulation
- Now the workflow is to first do generation (with whatever generator, using Gaudi or not), then run with `ddsim` and then run reconstruction
- More information in