Status of Key4hep and EDM4hep

Thomas Madlener

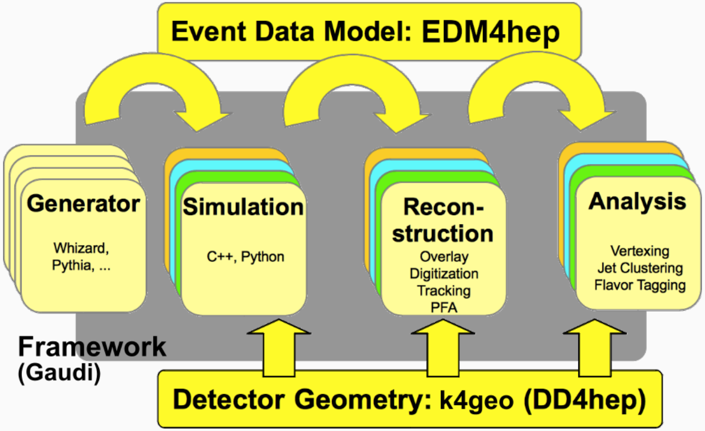
for the Key4hep developers

2023 International Workshop on Circular Electron Positron Collider

July 5, 2023

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 101004761.

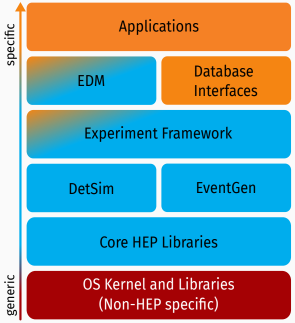
From generation to analysis - the general workflow



- Many steps involved from generating events to analyzing them
- Hundreds of SW packages
    - Building & deploying
    - Consistency
    - Reproducibility
- Try to give an overview of the **Key4hep** SW ecosystem

Key4hep - A (very) brief introduction

- Future detector studies rely on well maintained software for studying their potential
- Maintenance of a consistent HEP SW stack is non-trivial
    - Ecosystem of interacting components
- Sharing the burden allows everybody to reap the benefits
    - Make best use of scarce (human) resources
- **Regular contributions from ILC, CLIC, FCC, CEPC, EIC, (MuonCollider), ...**
- Support from major R&D initatives
    - CERN R&D for Future Experiments, AIDAinnova WP12, ECFA



Key4hep goals

- Provide and maintain a consistent SW stack that allows to do physics studies for **all projects**
- Ensure interoperability of the necessary building blocks
- Reuse existing solutions where possible
    - A lot of experience from LHC experiments and LC communities
- Focus new developments on EW/Higgs factory specifics
- Share knowledge, processes, workflows and resources
    - Best practices, tutorials, documentation, ...

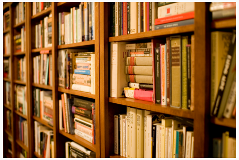
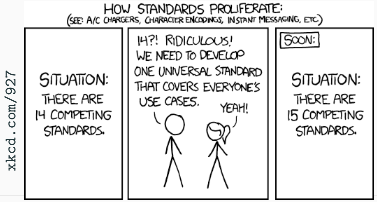


Photo by Stewart B. / CC-BY

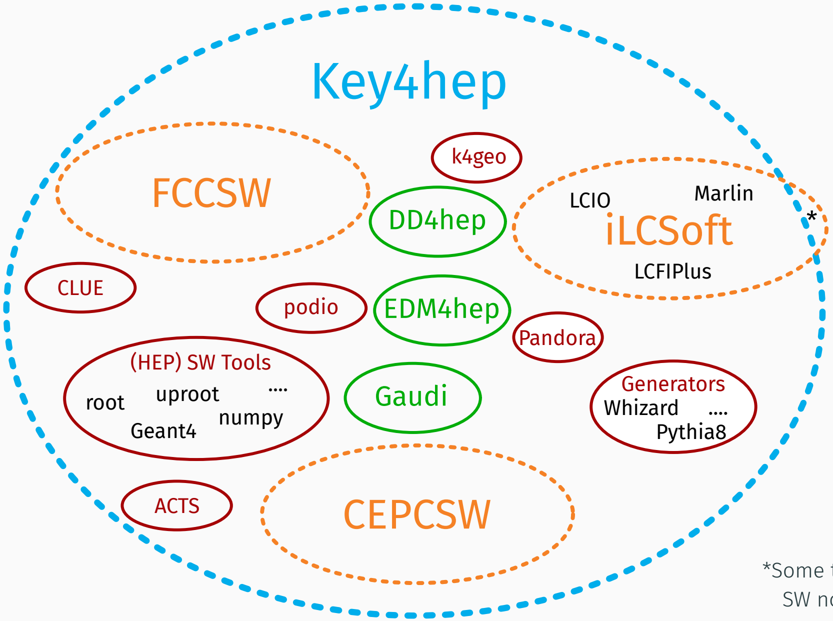
Non-goal

- Develop and maintain project specific software and workflows



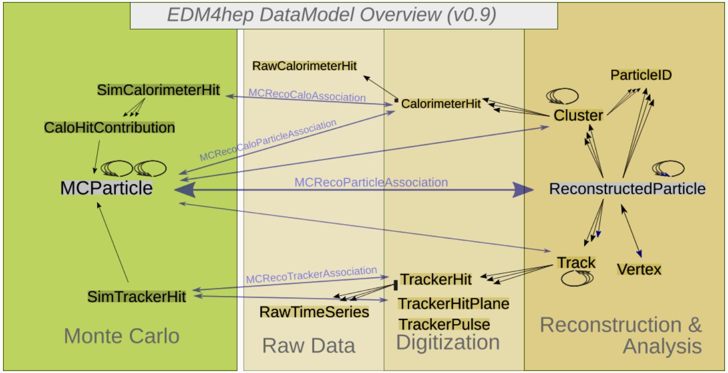
xkcd.com/927
HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

Key4hep (simplified) overview



*Some testbeam related SW not yet included

EDM4hep - The common EDM for Key4hep



key4hep/EDM4hep

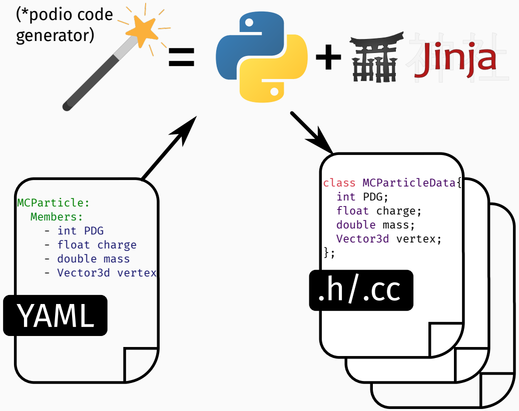
edm4hep.web.cern.ch

AIDASoft/podio

- Based on `LCIO` and `FCC-edm`
  - Focus on usability in analysis
- Quite stable over the last two years
- Addition of datatypes for **CEPC drift chamber study**
- Can easily be extended
  - Used by EDM4eic
  - Main purpose: prototyping
- Generated via `podio`

The podio EDM toolkit

- Implementing a performant event data model (EDM) is non-trivial
- Use `podio` to generate code starting from a high level description
- Provide an easy to use interface to the users
- Main customers
  - [key4hep/EDM4hep](#)
  - [eic/EDM4eic](#)
- **Finishing schema evolution for v1.0**



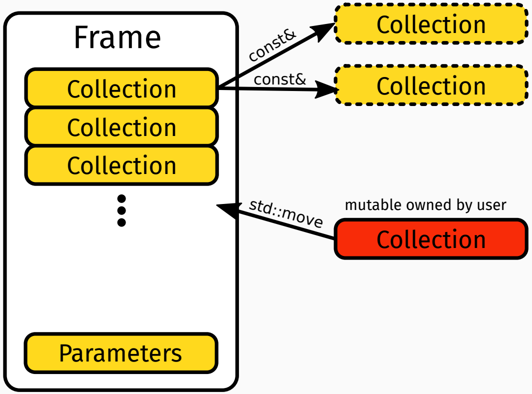
```
MCParticle:
  Members:
    - int PDG
    - float charge
    - double mass
    - Vector3d vertex
```

```
class MCParticleData{
  int PDG;
  float charge;
  double mass;
  Vector3d vertex;
};
```

[AIDASoft/podio](#)

The `Frame` - A generalized (event) data container

- Replaces deprecated `EventStore`
- *Type erased* container aggregating all relevant data
- Defines an *interval of validity* / category for contained data
  - Event, Run, readout frame, ...
- Easy to use and thread safe interface for data access
  - Immuatable read access only
  - Ownership model reflected in API
- Decouples I/O from operating on the data

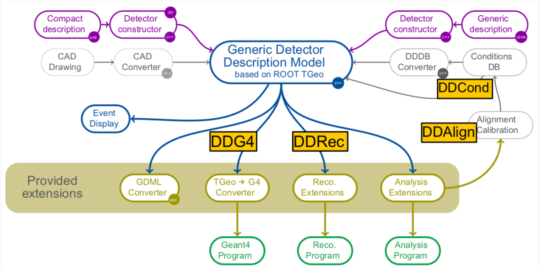


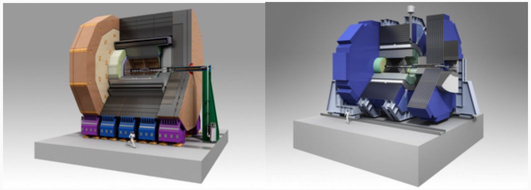
```cpp
template<typename CollT>
const CollT& get(const std::string& name) const;

template<typename CollT, /*enable_if*/>
const CollT& put(CollT&& collection,
                 const std::string& name);
```

DD4hep - Detector description

dd4hep.web.cern.ch

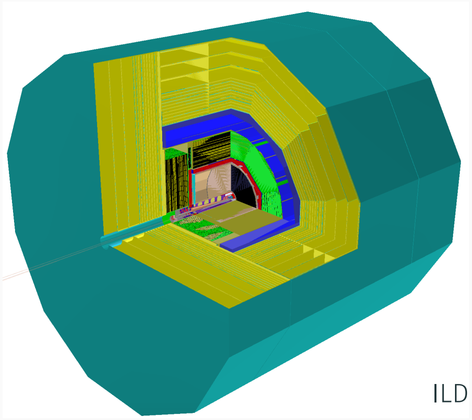
- Complete detector description
    - Geometry, materials, visualization, readout, alignment, calibration, ...
- From a **single source of information**
    - Simulation, reconstruction, analysis
- Comes with a powerful plug-in mechanism that allows customization
- More or less "industry standard" now
    - ILC, CLIC, FCC, CEPC, EIC, LHCb, CMS, ...
- `ddsim` - standalone simulation executable

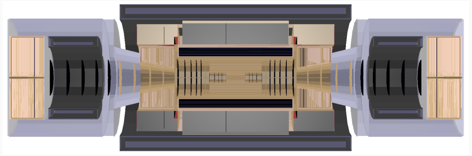




k4geo - The detector geometry repository

- `iLCSoft/lcgeo` $\rightarrow$ `key4hep/k4geo`
- Many existing detector models from LC studies
- Ongoing migration of detector concepts from HEP-FCC/FCCDetectors
  - Noble liquid ECAL
- New ARC detector concept in CLD
- IDEA detector (work in progress)
- Goal: central repository for detector descriptions


ILD


FCC-hh

Experiment Framework

- `Gaudi`, originally developed by LHCb, now also used by ATLAS, FCCSW and smaller experiments
  - Supports concurrency
  - “Battle-proven” from data taking during LHC operations
- Key4hep has decided to adapt `Gaudi` as its experiment framework
  - Contribute to its development where necessary
- Integration and migration of iLCSoft algorithms into Key4hep with the help of a `Marlin`→`Gaudi` wrapper
  - Allows to use `Marlin` processors within the `Gaudi` framework
  - key4hep/k4MarlinWrapper

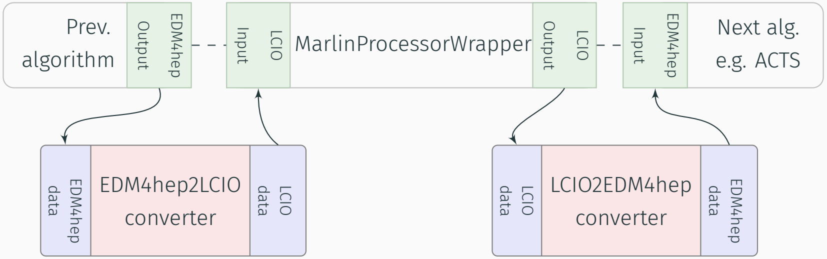
Frame based I/O in k4FWCore

- key4hep/k4FWCore offers core Key4hep services for Gaudi
  - **Data service for podio generated EDMs**
  - Historically grown separate implementation
- Replaced custom Reader / Writer with podio provided ones
  - (Almost) completely transparent
- `podio::Frame` not visible to user
- Some usability improvements in the works

```cpp
using namespace edm4hep;

// declare handle
DataHandle<MCParticleCollection> m_pHandle{
    "Particles",
    Gaudi::DataHandle::Reader,
    this};

// declare handle as property
declareProperty("ParticleColl",
                m_pHandle,
                "mc collection");

// use as
const auto particle = m_pHandle.get();
```

k4MarlinWrapper

- Wraps **Marlin processor** in a Gaudi algorithm and allows to **run them unchanged**
- Automatic, on-the-fly conversion between LCIO and EDM4hep
- **Allows to “mix and match” existing reconstruction algorithms with new developments**

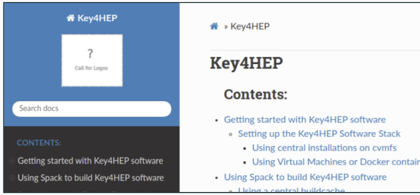


Key4hep resources

- (Rolling) latest release of the complete Key4hep software stack

```
source /cvmfs/sw-nightlies.hsf.org/key4hep/setup.sh
source /cvmfs/sw.hsf.org/key4hep/setup.sh
```

- **Release early and release often**
  - Solicit feedback as early as possible
- Documentation available at key4hep.web.cern.ch
- Active weekly meetings ($\sim 10 - 15$ attendees)
  - https://indico.cern.ch/category/11461/
- **Feedback and contributions are greatly appreciated**



Key4HEP
?
Call for Logos
Search docs
CONTENTS:
Getting started with Key4HEP software
Using Spack to build Key4HEP software
» Key4HEP

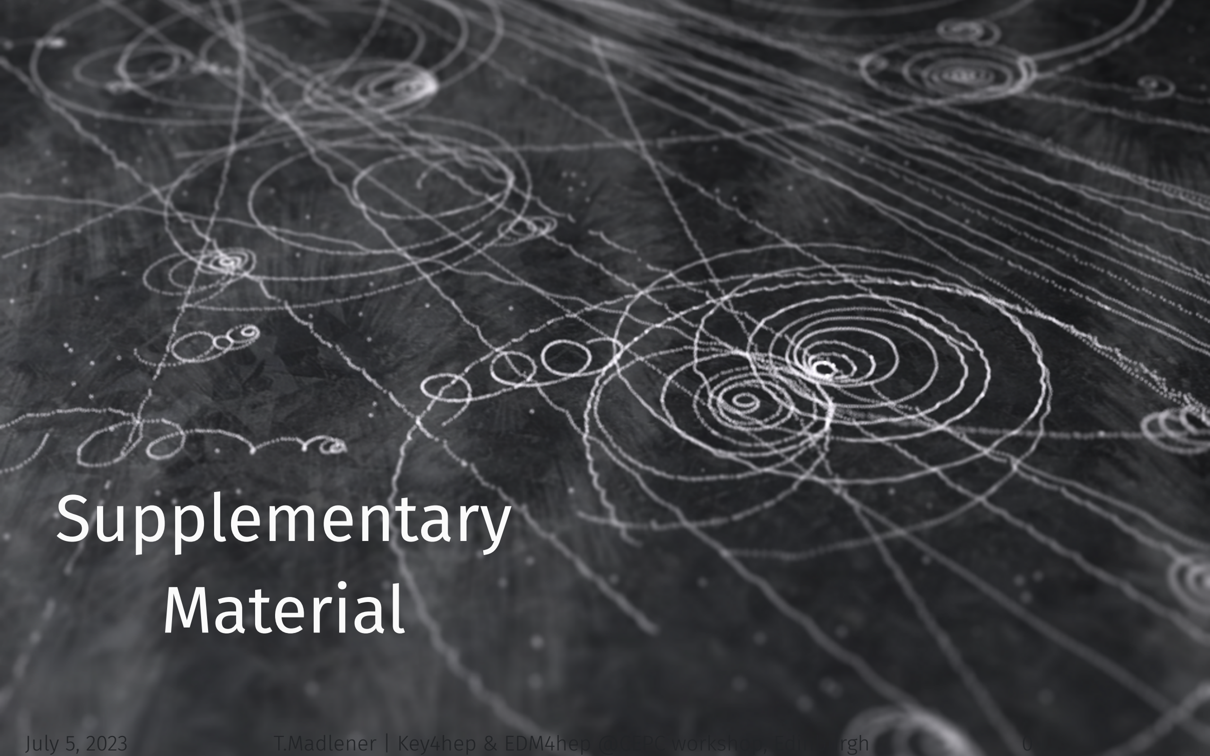
Key4HEP

Contents:

- Getting started with Key4HEP software
  - Setting up the Key4HEP Software Stack
    - Using central installations on cvmfs
    - Using Virtual Machines or Docker contair
- Using Spack to build Key4HEP software
  - Using a central buildcache

Summary

- Key4hep aims at providing a common software stack for **all future collider projects**
- Very successful in **bringing together communities** and **focusing on common approaches**
- Ongoing effort to stabilize core components
- **Key4hep can be used for future collider studies now**
    - CEPCSW is doing this already
- **Still a lot of work ahead**
    - Very happy to welcome new contributors



Supplementary Material

Pointers to software (re)sources

- Key4hep
  - key4hep.github.io/key4hep-doc
  - key4hep - github organisation
- EDM4hep
  - key4hep/EDM4hep
  - cern.ch/edm4hep
- DD4hep
  - AIDASoft/DD4hep
  - dd4hep.web.cern.ch
- iLCSoft
  - iLCSoft - github organisation
  - ilcsoft.desy.de

MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A 0x6339392C, 0x7363682E.
I HATE YOU.

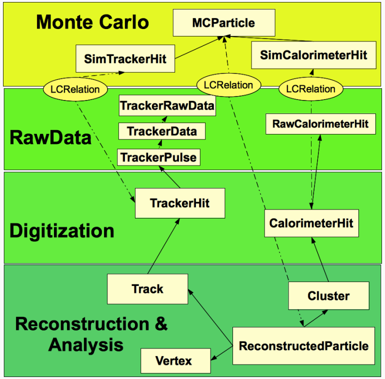
xkcd.com/138

Key4hep packages

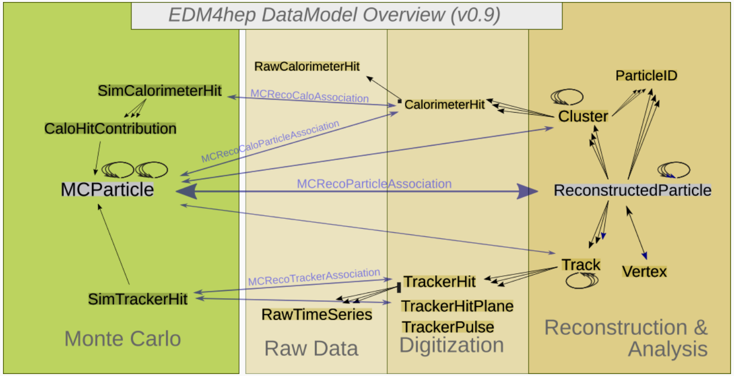
- `k4FWCore` key4hep/k4FWCore
  - Core Key4hep framework providing core functionality, e.g.
    - Data Service for EDM4hep inputs
    - Overlay for backgrounds
- `k4SimDelphes` for Delphes fast simulation key4hep/k4SimDelphes
- `k4MarlinWrapper` Marlin proc. wrapper key4hep/k4MarlinWrapper
- Many packages migrated from FCCSW to Key4hep
  - `k4SimGeant4` for Geant4 simulation integration HEP-FCC/k4SimGeant4
  - `k4Gen` for generic generator interface HEP-FCC/k4Gen
  - ...
- Ongoing work to integrate more components
  - ACTS tracking framework acts-project/acts | key4hep/k4ActsTracking
  - CLUE fast clustering algorithms .cern.ch/kalos/CLUE | key4hep/k4CLUE

LCIO → EDM4hep converter reloaded

LCIO



EDM4hep



- Large existing data sets in LCIO format
- Very similar high level structure but some differences in details

LCIO → EDM4hep converter reloaded

- Complete overhaul of pre-existing functionality
    - Shared library in key4hep/k4EDM4hep2LcioConv
    - Originally implemented in key4hep/k4LCIOReader
- Standalone executable (no Gaudi or Marlin!)

```
lcio2edm4hep input.slcio output.edm4hep.root
```

  - For all details see README
  - Available in recent nightly builds
- Using the `podio::Frame`
- Support all features that are necessary for ILD

Ongoing work (selection)

ACTS integration

- ACTS can now digest DD4hep detectors (with annotations)
- Minimal EDM4hep I/O support
  - More general solution under discussion
- Major effort with significant personpower requirements

Gaudi modernization

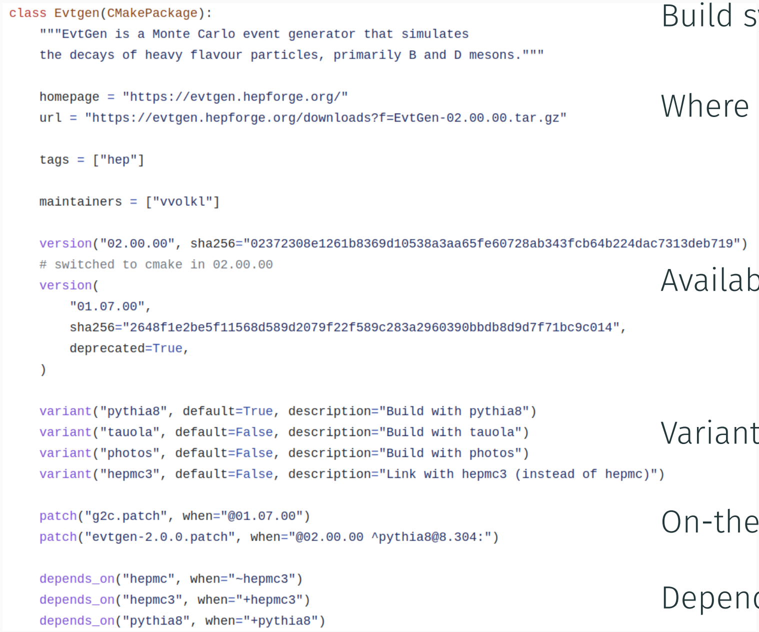
- Switch towards more modern Gaudi approach (*Gaudi Functional*)
  - “Thread safe by default”
- Missing documentation is a major hurdle

Spack for Key4hep

- Spack is a package manager
  - Independent of operating system
  - Builds all packages from source
- Originaly developed by the HPC community
  - Emphasis on dealing with **multiple configurations** of the same package
- Basic building block is a formalized build procedure $\rightarrow$ **spack recipe**
  - Build instructions, dependencies, versions and location of source code
  - $\sim$ 6650 packages currently available from spack
  - Key4hep maintains repository with additional packages
- The whole Key4hep software stack can be built from scratch using spack

```
spack install key4hep-stack
```

Spack recipe



```
class Evtgen(CMakePackage):
    """EvtGen is a Monte Carlo event generator that simulates
    the decays of heavy flavour particles, primarily B and D mesons."""

    homepage = "https://evtgen.hepforge.org/"
    url = "https://evtgen.hepforge.org/downloads?f=EvtGen-02.00.00.tar.gz"

    tags = ["hep"]

    maintainers = ["vvolkl"]

    version("02.00.00", sha256="02372308e1261b8369d10538a3aa65fe60728ab343fcb64b224dac7313deb719")
    # switched to cmake in 02.00.00
    version(
        "01.07.00",
        sha256="2648f1e2be5f11568d589d2079f22f589c283a2960390bbdb8d9d7f71bc9c014",
        deprecated=True,
    )

    variant("pythia8", default=True, description="Build with pythia8")
    variant("tauola", default=False, description="Build with tauola")
    variant("photos", default=False, description="Build with photos")
    variant("hepmc3", default=False, description="Link with hepmc3 (instead of hepmc)")

    patch("g2c.patch", when="@01.07.00")
    patch("evtgen-2.0.0.patch", when="@02.00.00 ^pythia8@8.304:")

    depends_on("hepmc", when="~hepmc3")
    depends_on("hepmc3", when="+hepmc3")
    depends_on("pythia8", when="+pythia8")
```

Build system

Where to find source code

Available versions

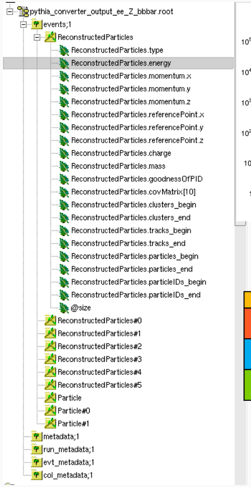
Variants / build options

On-the-fly patches

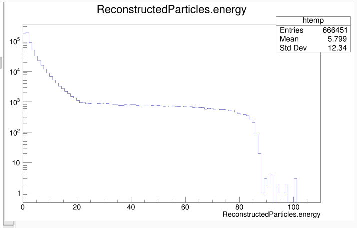
Dependencies

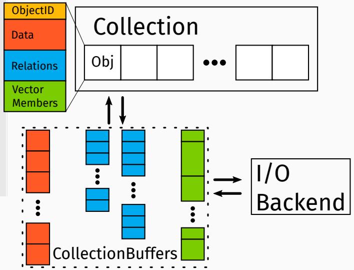
podio supports different I/O backends

- Default **ROOT** backend
  - POD buffers are stored as branches in a `TTree`
  - Files can be interpreted **without EDM library**(!)
  - Can be used in `RDataFrame` or with `uproot`
- Alternative **SIO** backend
  - Persistency library used in `LCIO`
  - Complete events are stored as binary records
- Adding more I/O backends is possible



```
pythia_converter_output_ee_Z_bbbar.root
  events;1
    ReconstructedParticles
      ReconstructedParticles.type
      ReconstructedParticles.energy
      ReconstructedParticles.momentum.x
      ReconstructedParticles.momentum.y
      ReconstructedParticles.momentum.z
      ReconstructedParticles.referencePoint.x
      ReconstructedParticles.referencePoint.y
      ReconstructedParticles.referencePoint.z
      ReconstructedParticles.charge
      ReconstructedParticles.mass
      ReconstructedParticles.goodnessOfPID
      ReconstructedParticles.covMatrix[10]
      ReconstructedParticles.clusters_begin
      ReconstructedParticles.clusters_end
      ReconstructedParticles.tracks_begin
      ReconstructedParticles.tracks_end
      ReconstructedParticles.particles_begin
      ReconstructedParticles.particles_end
      ReconstructedParticles.particleIDs_begin
      ReconstructedParticles.particleIDs_end
      @size
    ReconstructedParticles#0
    ReconstructedParticles#1
    ReconstructedParticles#2
    ReconstructedParticles#3
    ReconstructedParticles#4
    ReconstructedParticles#5
    Particle
    Particle#0
    Particle#1
  metadata;1
  run_metadata;1
  evt_metadata;1
  col_metadata;1
```


ReconstructedParticles.energy
htemp
Entries 666451
Mean 5.799
Std Dev 12.34
10^5
10^4
10^3
10^2
ReconstructedParticles.energy


ObjectID
Data
Relations
Vector Members
Collection
Obj
•••
CollectionBuffers
I/O Backend