Discussion: Transition to new Key4hep Tools

Modernising ILD software
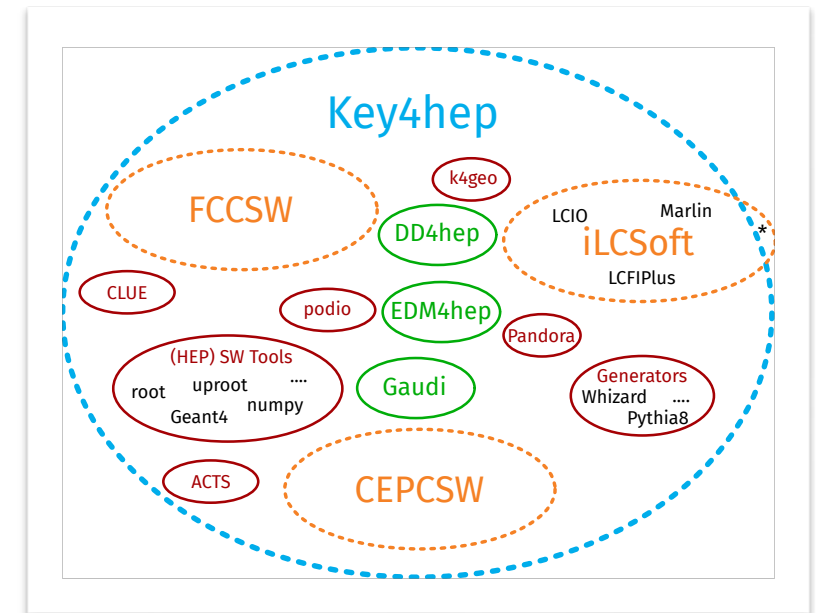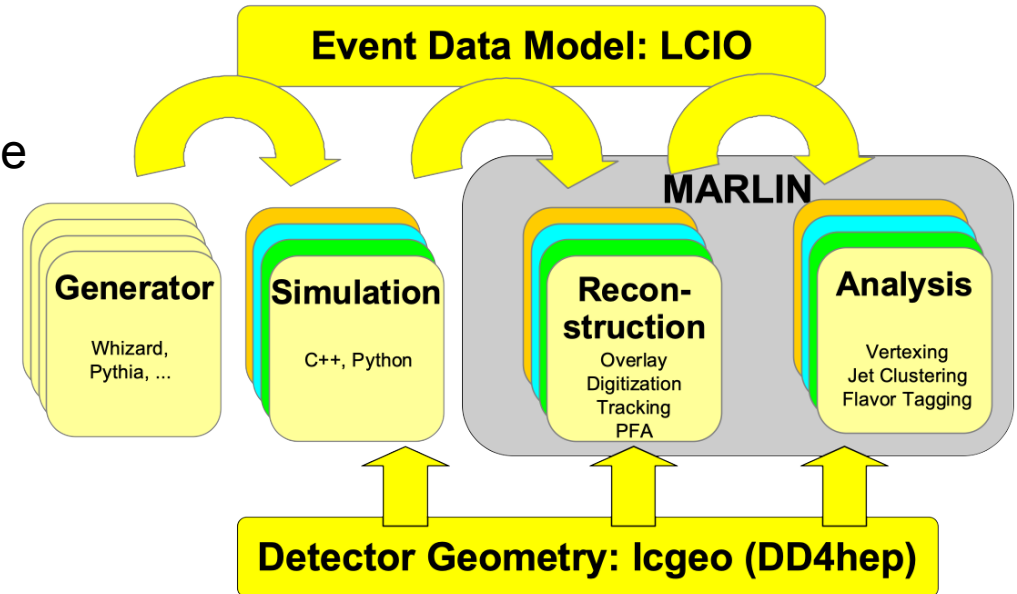
17.01.2024

Frank Gaede, Thomas Madlener, DESY
ILD Meeting 2024, CERN

ILD software in Key4hep

iLCSoft as integral part of Key4hep

- the full ILD software chain - in iLCSoft - is integral part of the Key4hep software stack

  - can run ILD full reconstruction in Gaudi via **MarlinWrapper** - producing LCIO

- **EDM4hep** is the *designated successor* of LCIO

  - similar EDM but more consistent, better performance and easier to use

- we have a **smooth transition path** build into the current Key4hep stack

Today discuss taking the next step for the next ILD projects

ILD software in Key4hep

iLCSoft as integral part of Key4hep

- the full ILD software chain - in iLCSoft - is integral part of the Key4hep software stack

  - can run ILD full reconstruction in Gaudi via **MarlinWrapper** - producing LCIO

- **EDM4hep** is the *designated successor* of LCIO

  - similar EDM but more consistent, better performance and easier to use

- we have a **smooth transition path** build into the current Key4hep stack
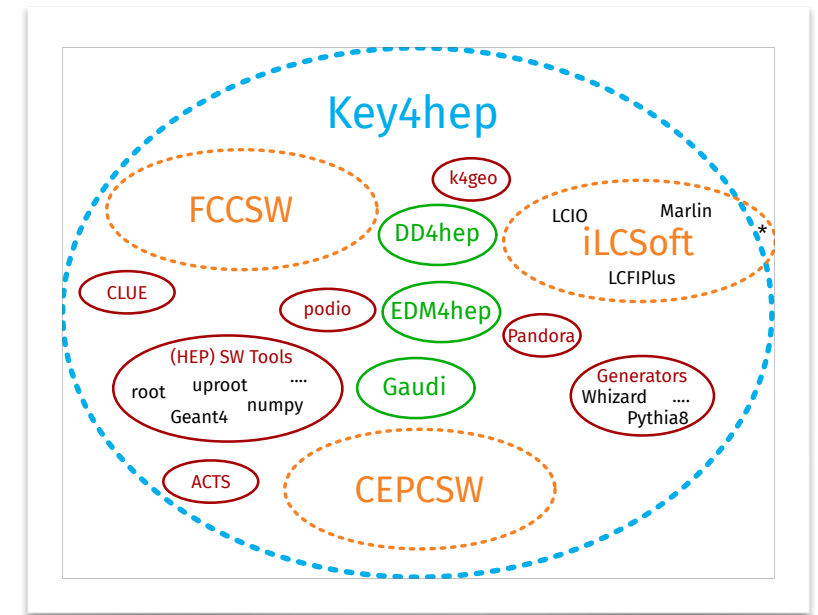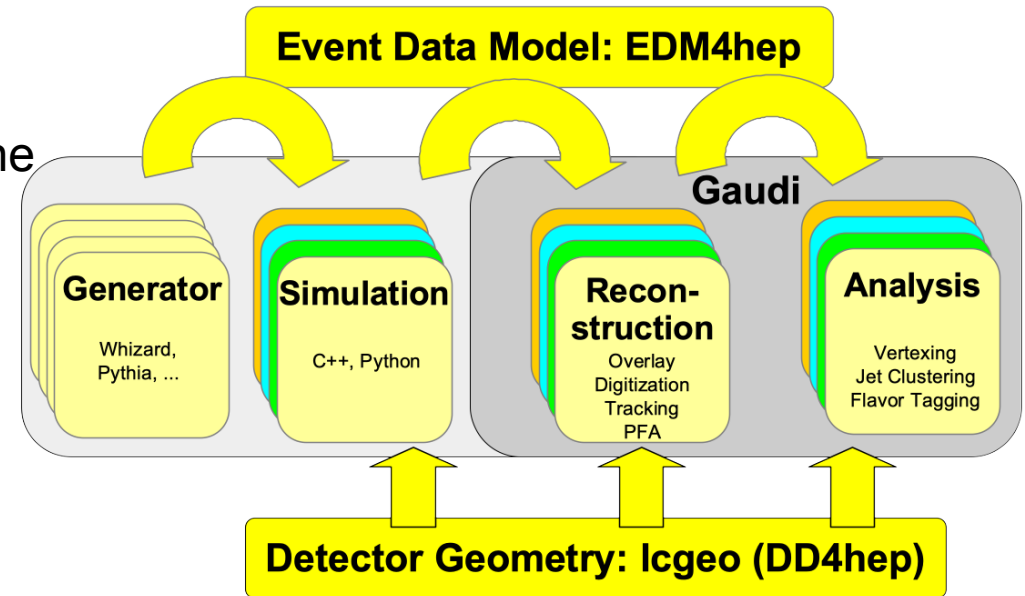
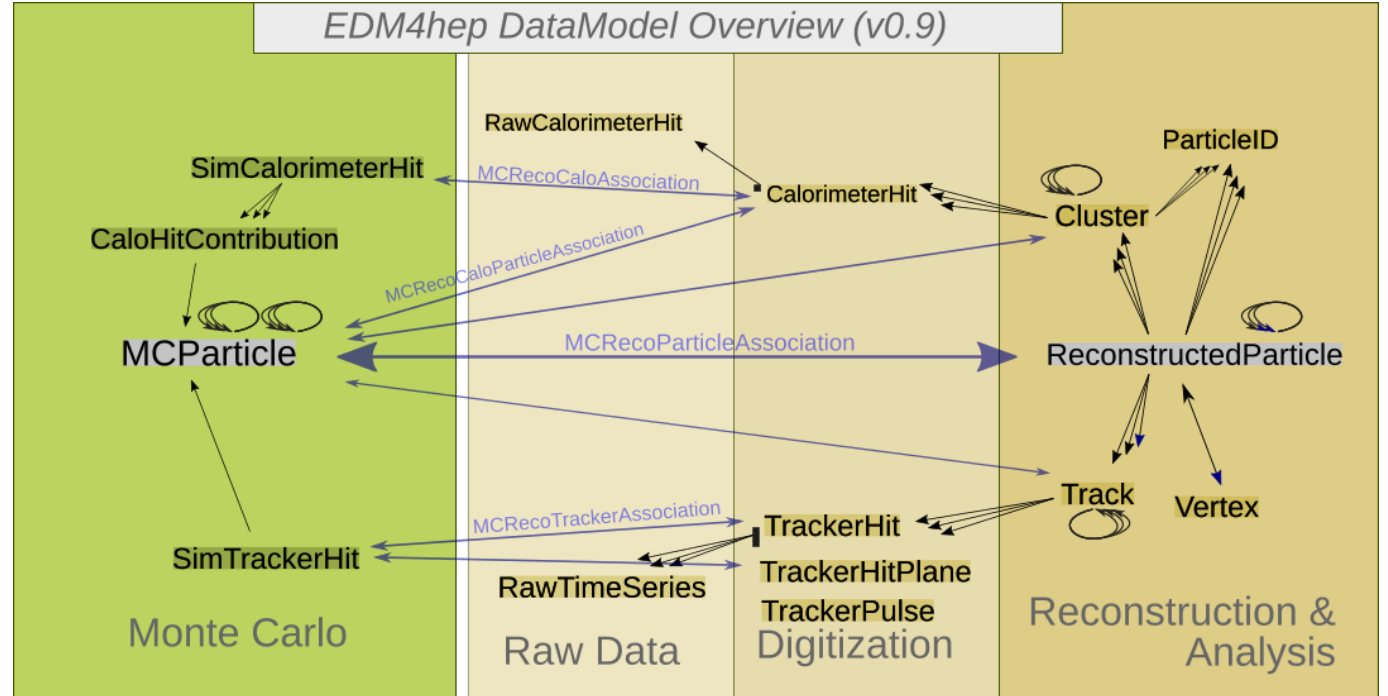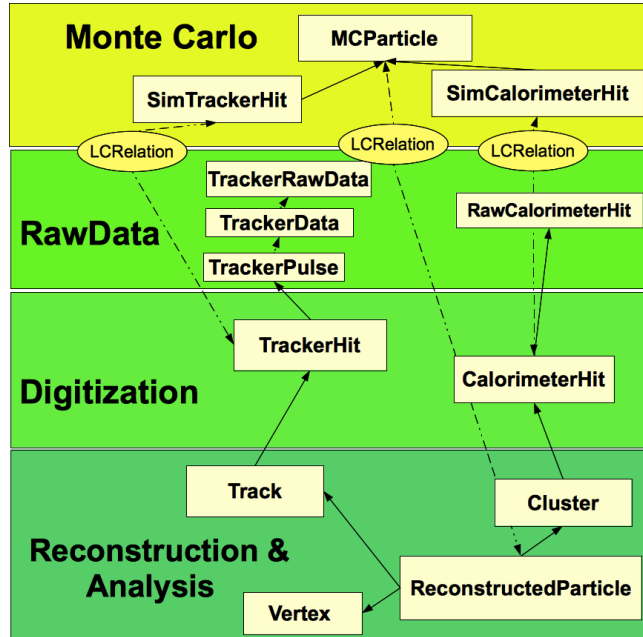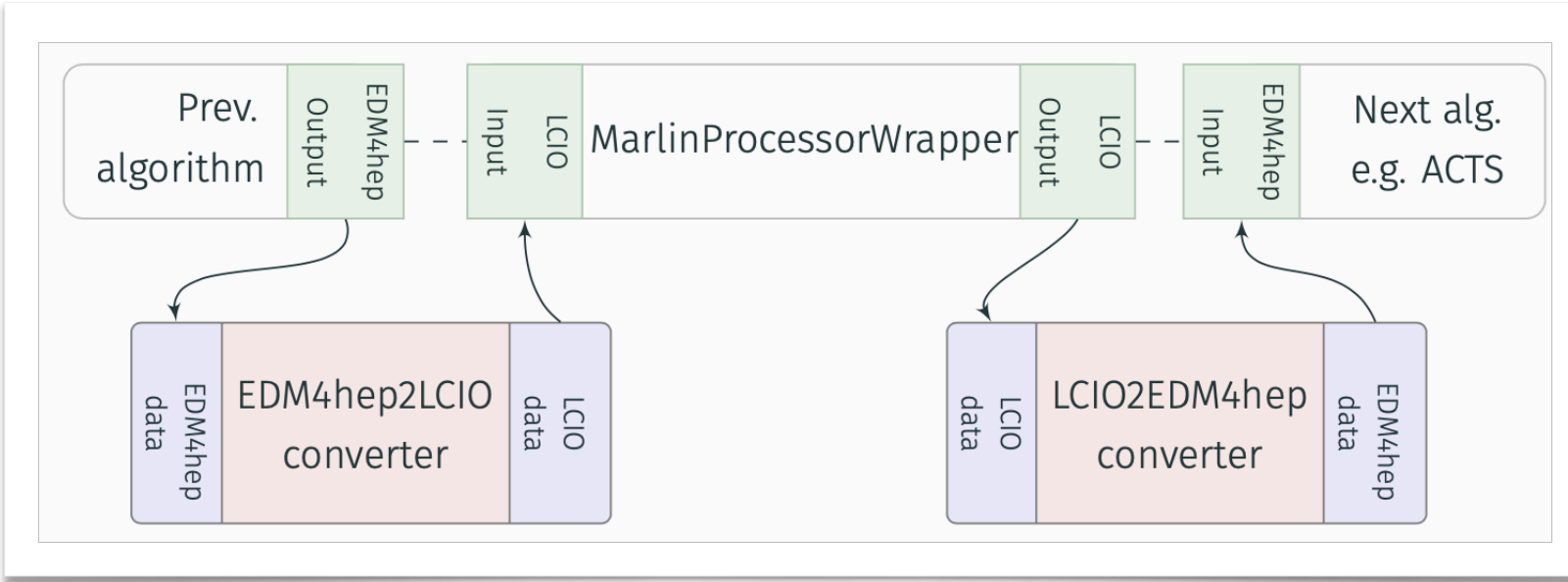Today discuss taking the next step for the next ILD projects

Event Data Model: EDM4hep

Gaudi

Generator — Whizard, Pythia, ...

Simulation — C++, Python

Recon-struction — Overlay Digitization Tracking PFA

Analysis — Vertexing Jet Clustering Flavor Tagging

Detector Geometry: lcgeo (DD4hep)

Key4hep

FCCSW — k4geo — DD4hep — LCIO — Marlin — iLCSoft — LCFIPlus — CLUE — podio — EDM4hep — Pandora — (HEP) SW Tools — root — uproot — .... — numpy — Geant4 — Gaudi — Generators — Whizard — .... — Pythia8 — ACTS — CEPCSW

EDM4hep

designated successor of LCIO

- EDM4hep largely based on (battle-proven) LCIO EDM

- EDM4hep is fast and efficient, supports multithreading and more consistent EDM w/ ROOT files

- *k4EDM4hep2LcioConv* provides consistent conversion between the two

K4MarlinWrappper

mix and match Marlin and Gaudi algorithms

courtesy. C. Hensel

- 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

Proposal
for advancing the smooth transition

- run **MarlinWrapper** for **central ILD reconstruction** from now on
- create **EDM4hep** output files - parallel to LCIO
    - encourage **analyses in EDM4hep**

- encourage people to **develop new algorithms in Gaudi/ EDM4hep**
    - in this phase, both LCIO and Marlin fully supported

- the next bigger **ILD production** (document ?) should be done w/ Gaudi/EDM4hep in any case
    - (still making use of MarlinWrapper and existing ILD reco tools)

additional material

Gaudi
the application framework

- C++ application framework for HEP

- developed at CERN

- used in production for

  - LHCb and ATLAS (*battle-proven*)

  - FCC-SW and smaller experiments

  - and now in Key4HEP

- highly configurable

  - EDM, workflows (algorithms)

- allows parallelisation through multi-threading

- integration of heterogeneous resources

  - CPUs, GPUs, FPGAs,…

🏠 Gaudi

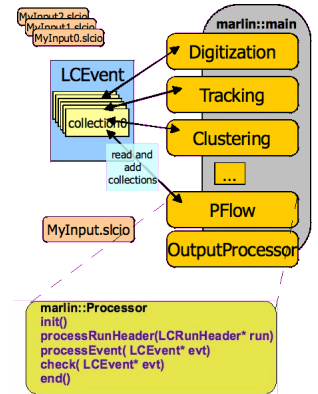|	Marlin	Gaudi
language	C++	C++
working unit	Processor	Algorithm
config language	XML	Python
transient data format	LCIO	anything
set up function	init	initialize
work function	processEvent	execute
wrap up function	end	finalize

similar to MARLIN framework
yet more powerful and larger user basis
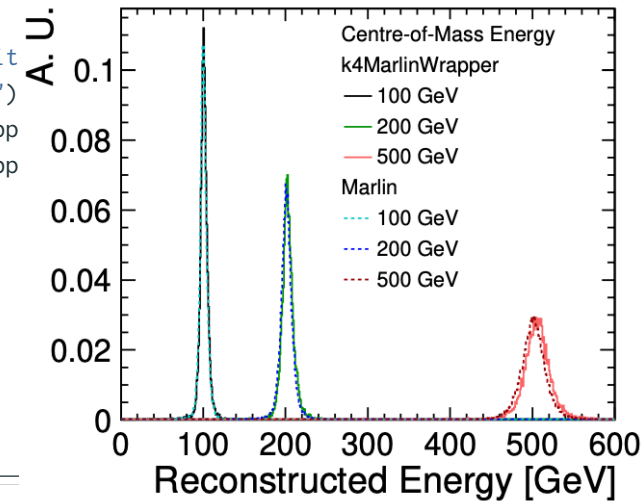
k4MarlinWrapper

running Marlin processors in Gaudi (Key4hep)

- set of Gaudi algorithms that wrap Marlin processors

  - developed by CERN-SFT

  - automatic XML to Python steering file conversion

- tools for automatic in-memory, on-demand conversion between LCIO and EDM4hep

  - developed by IHEP, CERN, DESY

  - possibility to mix Marlin processors with genuine Gaudi algorithms

- this is the intended **working horse for a smooth transition** from iLCSoft to Key4hep

- CLIC and ILD **full reconstruction** run as *proof-of-concept*

```
MyTPCDigiProcessor = MarlinProcessorWrapper("MyTPCDigiProcessor"
MyTPCDigiProcessor.OutputLevel = INFO
MyTPCDigiProcessor.ProcessorType = "DDTPCDigiProcessor"
MyTPCDigiProcessor.Parameters = [
            "DiffusionCoeffRPhi", "0.025", END_TAG,
            "DiffusionCoeffZ", "0.08", END_TAG,
            "DoubleHitResolutionRPhi", "2", END_TAG,
            "DoubleHitResolutionZ", "5", END_TAG,
            "HitSortingBinningRPhi", "2", END_TAG,
            "HitSortingBinningZ", "5", END_TAG,
            "MaxClusterSizeForMerge", "3", END_TAG,
            "N_eff", "22", END_TAG,
            # ...
            ]
algList.append(MyTPCDigiProcessor)
```
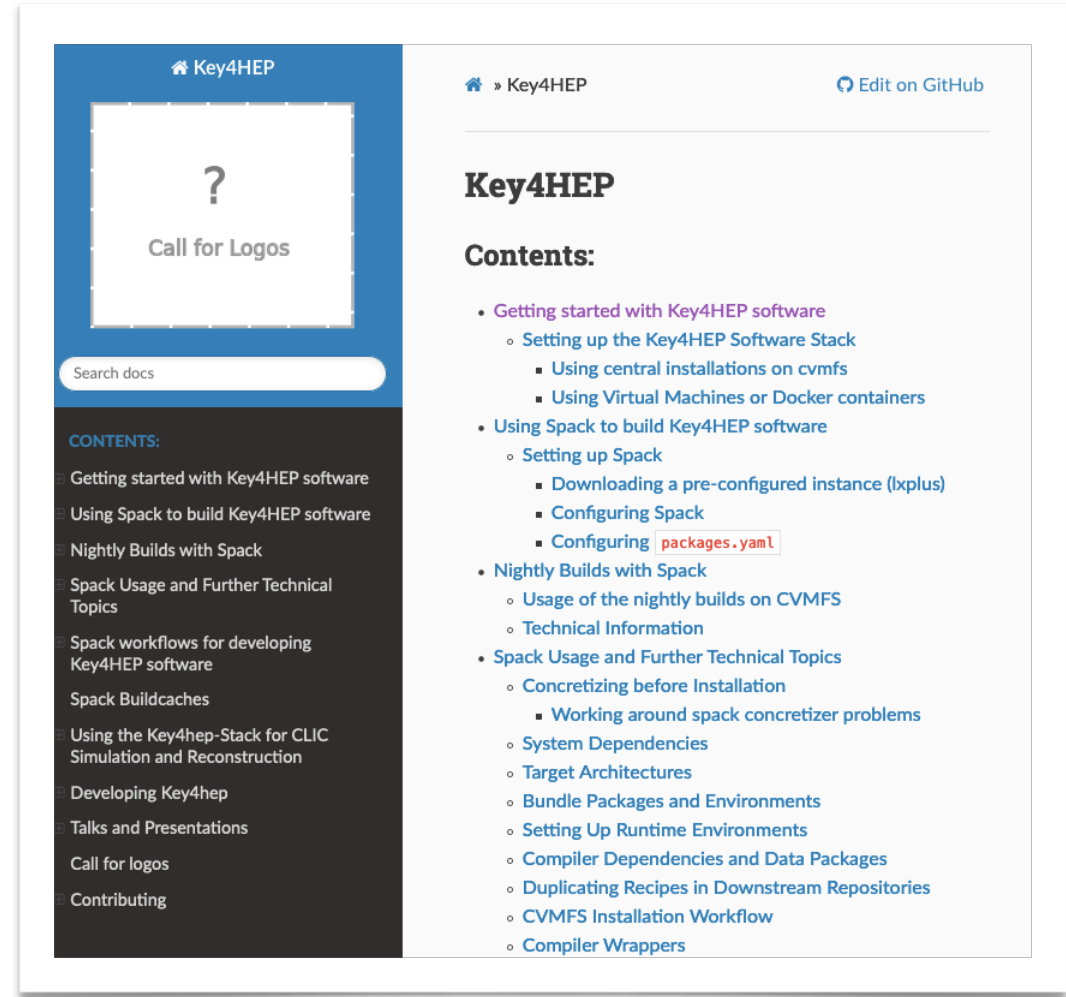
```
from Gaudi.Configuration import *
CONSTANTS = {'BCReco': "3TeV",}
parseConstants(CONSTANTS)
# ...
read = LcioEvent()
InitDD4hep = MarlinProcessorWrapper("Init
Config = MarlinProcessorWrapper("Config")
VXDBarrelDigitiser = MarlinProcessorWrapp
VXDEndcapDigitiser = MarlinProcessorWrapp
# ...
algList.append(InitDD4hep)
algList.append(Config)
# algList.append(OverlayFalse)
# algList.append(Overlay350GeV_CDR)
algList.append(VXDBarrelDigitiser)
algList.append(VXDEndcapDigitiser)
# ...
```

pointers to documentation
entry points to Key4hep

- Key4hep GitHub Project

  - https://github.com/key4hep

- Key4hep main documentation page
  - https://key4hep.github.io/key4hep-doc/

- Doxygen available., e.g. for EDM4hep

  - https://edm4hep.web.cern.ch/

- iLCSoft Github Project

  - https://github.com/ilcsoft