Overview Key4hep
with a focus on reconstruction

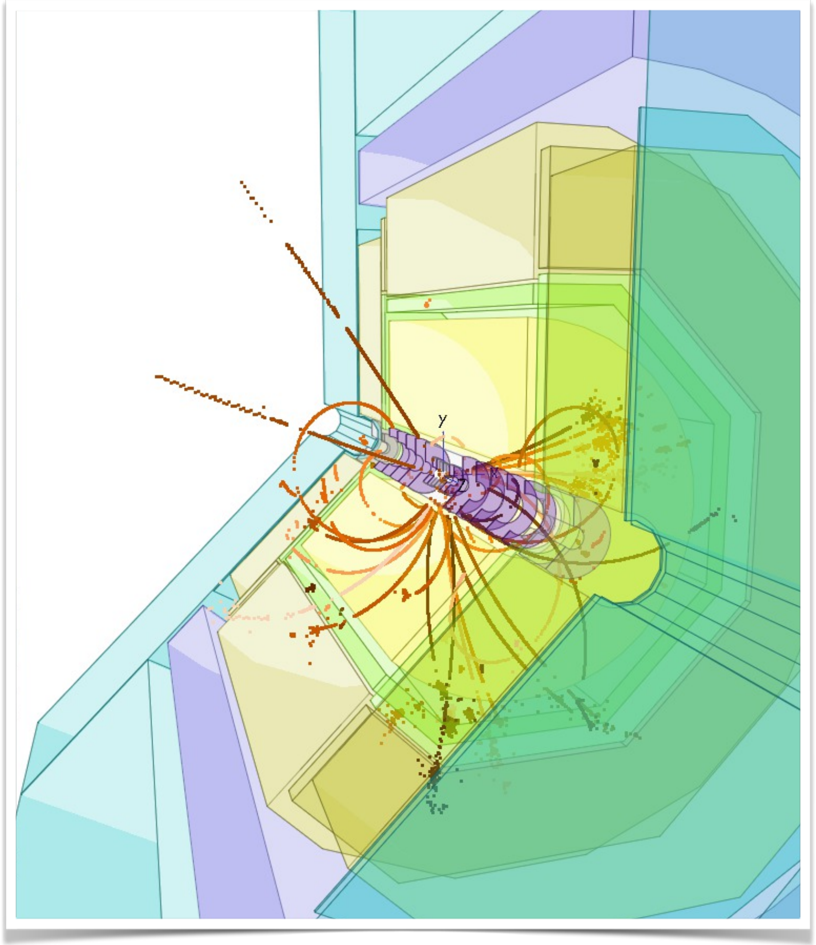
2nd ECFA Topical Workshop on Reconstruction

11.07.2023

Frank Gaede, DESY
for the Key4hep team

Outline

- Introduction
- core components of Key4hep
  - DD4hep, Gaudi, PODIO/EDM4hep
- reconstruction tools in Key4hep
  - MarlinWrapper - interplay iLCSoft and Gaudi
  - genuine Key4hep algorithms
- Conclusion and Outlook



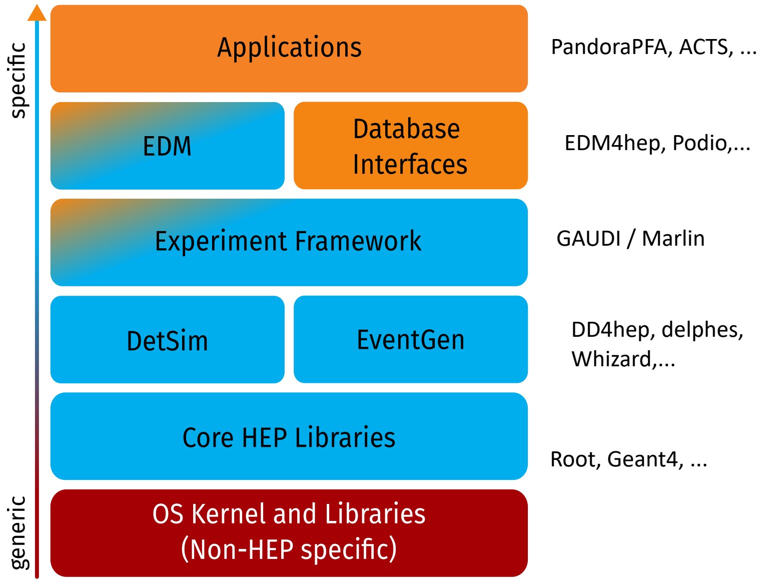
Introduction to Key4hep

turnkey software stack for all future colliders

- HEP community decided to develop a **common turnkey software stack** – for future collider studies

- create a software ecosystem integrating in an **optimal way the best software components** to provide a **ready-to-use full-fledged solution** for data processing of **HEP** experiments (with initial focus on future colliders)

  - similar to what was done with **iLCSoft** for the linear collider community >15 years ago



specific

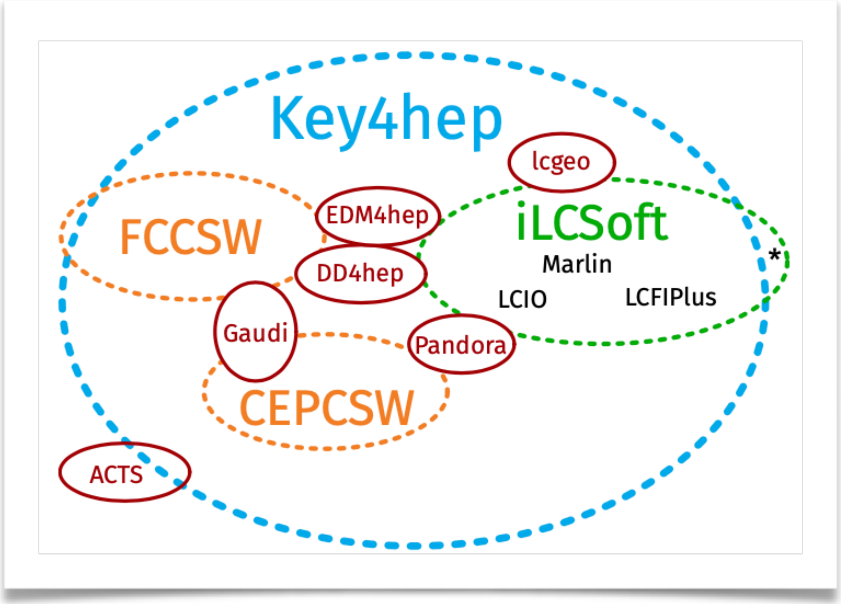
generic

Applications	PandoraPFA, ACTS, ...
EDM Database Interfaces	EDM4hep, Podio,...
Experiment Framework	GAUDI / Marlin
DetSim EventGen	DD4hep, delphes, Whizard,...
Core HEP Libraries	Root, Geant4, ...
OS Kernel and Libraries (Non-HEP specific)	

- supported by **HSF** and **CERN EP-R&D** and *AIDAinnova*

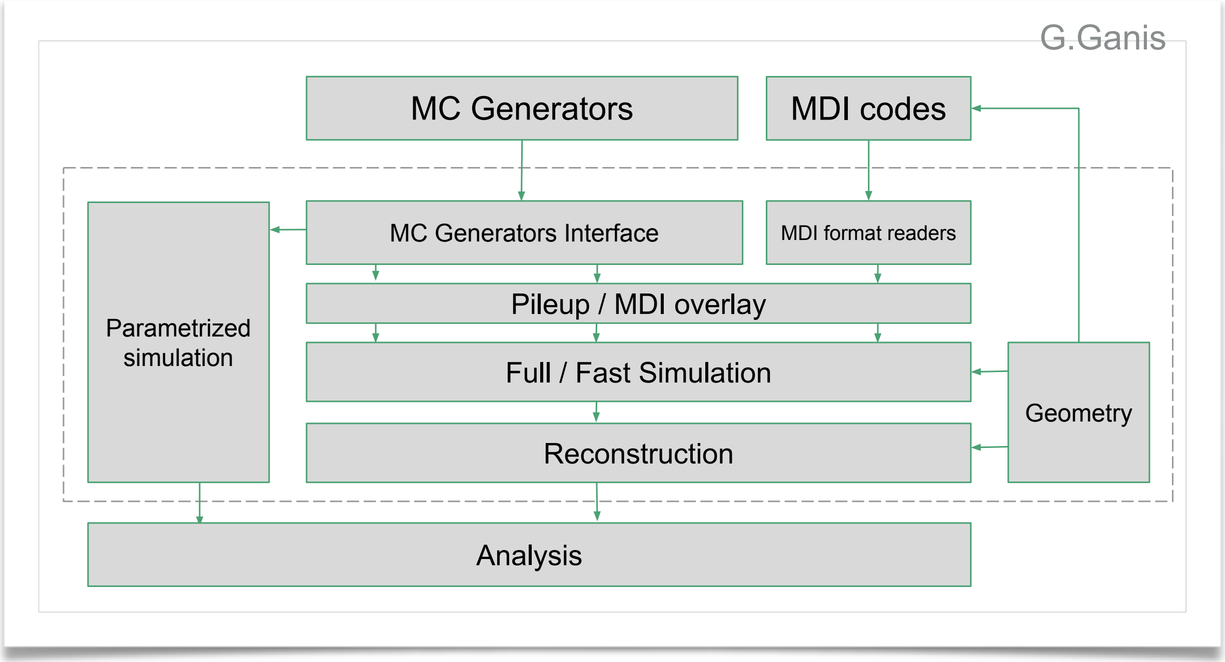
- involved communities/contributors:

  - CEPC, CLIC, FCC, EIC, ILC, LUXE, Muon Collider …



Workflows in HEP and Interoperability

the analyst and developers view



G.Ganis

MC Generators

MDI codes

MC Generators Interface

MDI format readers

Parametrized simulation

Pileup / MDI overlay

Full / Fast Simulation

Geometry

Reconstruction

Analysis

- top:typically workflows as seen by the analyst
- right: software techniques to enable these workflows in a turnkey stack

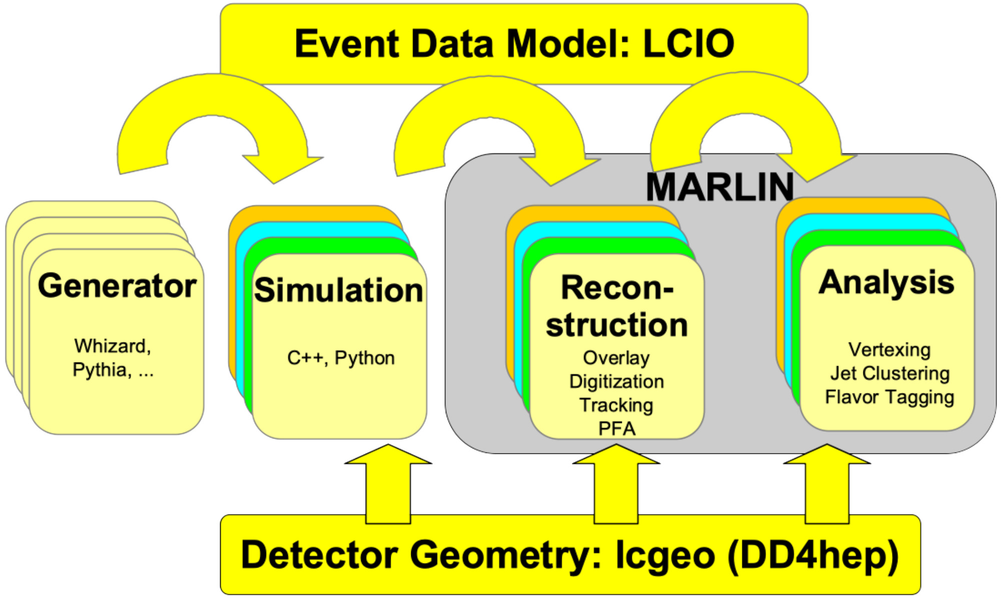
- Level 0 - **Common Data Formats**
  - Maximal interoperability, even on different hardware
- Level 1 - **Callable Interfaces**
  - Defined for one or more programming languages
  - Implementation quality of interfaced components important
  - Required to define plugins
- Level 2 - **Introspection Capabilities**
  - Software elements to facilitate the interaction of objects in a generic manner such as Dictionaries and Scripting interfaces
  - Language bindings, e.g. PyROOT
- Level 3 - **Component Level**
  - Software components are part of a common framework, optimal interplay
  - Common configuration, log and error reporting, plug-in management, ...

The common software vision

the high altitude view

- complete set of tools for
  - **generation, simulation, reconstruction, analysis**
  - build, package, test, deploy

- core ingredients of current **Key4hep**
  - **PODIO** for **EDM4hep** (based on LCIO and FCC-edm)
  - **Gaudi** framework, devel/used for (HL-)LHC
  - **DD4hep** for geometry
    - originally developed for LC now adopted by community
  - **spack** package manager



- generic scheme of iLCSoft framework and core tools

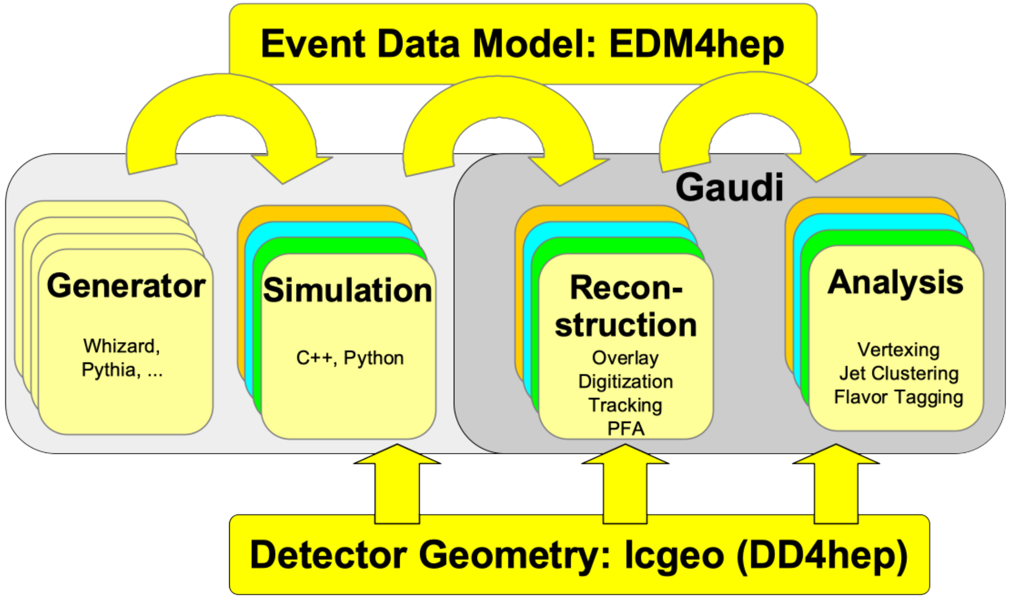
- very similar now in Key4hep:
  - Marlin -> Gaudi
  - LCIO -> EDM4hep

The common software vision

the high altitude view

- complete set of tools for
  - **generation, simulation, reconstruction, analysis**
  - build, package, test, deploy

- core ingredients of current **Key4hep**
  - **PODIO** for **EDM4hep** (based on LCIO and FCC-edm)
  - **Gaudi** framework, devel/used for (HL-)LHC
  - **DD4hep** for geometry
    - originally developed for LC now adopted by community
  - **spack** package manager



- generic scheme of iLCSoft framework and core tools

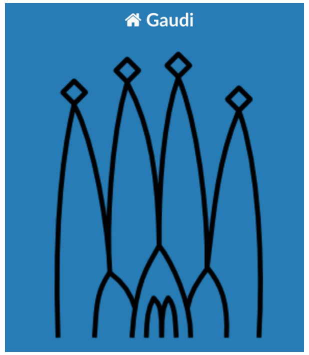
- very similar now in Key4hep:
  - Marlin -> Gaudi
  - LCIO -> EDM4hep

- re-use tools and algorithms from iLCSoft - using MarlinWrapper (see later)

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,…



	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

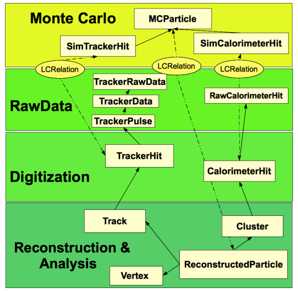
EDM4hep

generic HEP event data model

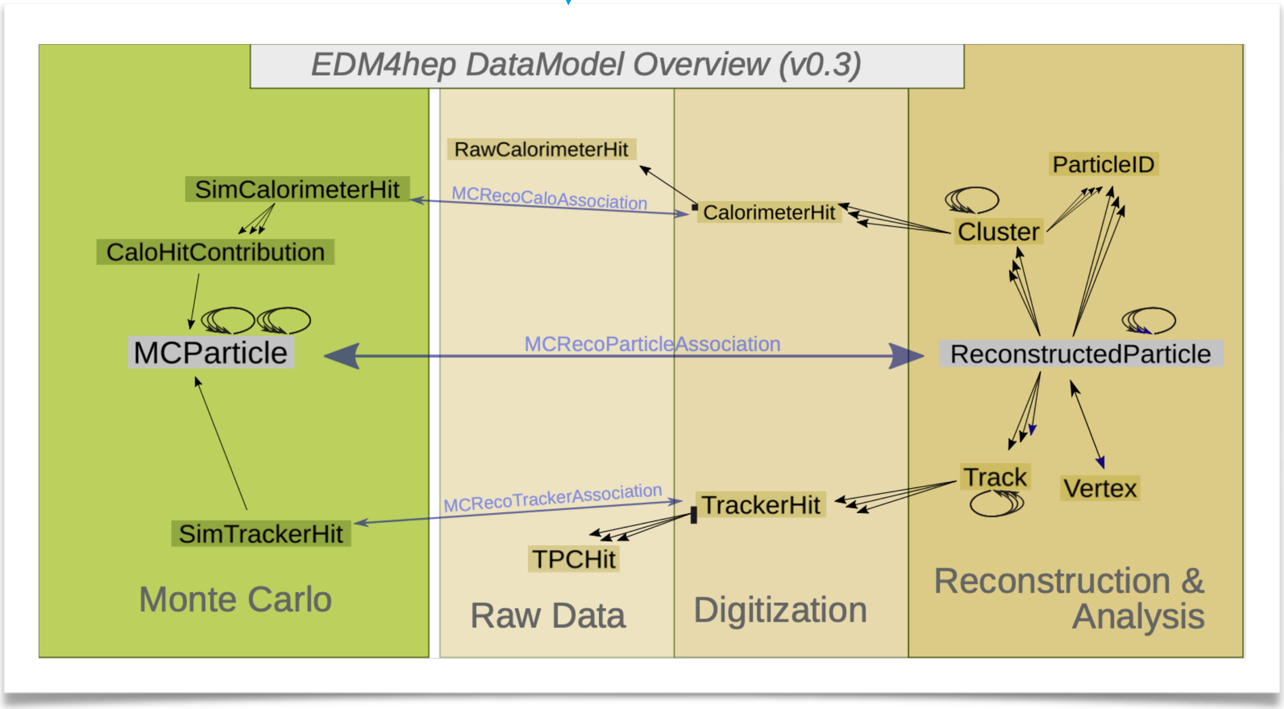
- the event data model defines the language for all data processing tasks in HEP
- EDM4hep aims at getting this right for all future collider projects - independent of the type of collider
    - (ee, mumu, pp,…)
- largely based on (battle-proven) LCIO and FCC-EDM
- first example analyses for FCC-hh successful
- EDM access and I/O part needs to be
    - fast and efficient
    - support multithreading
    - transparent choice of actual I/O system
- implemented with PODIO

see more on EDM4hep in next talk by Thomas

LCIO



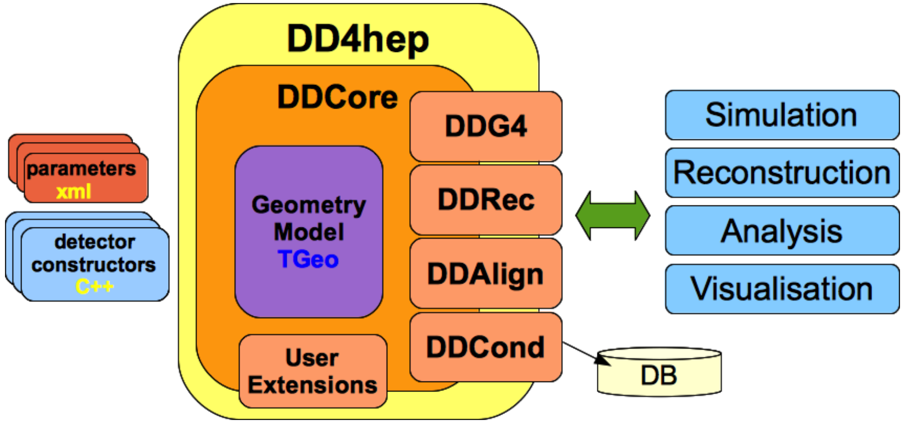
FCC-EDM



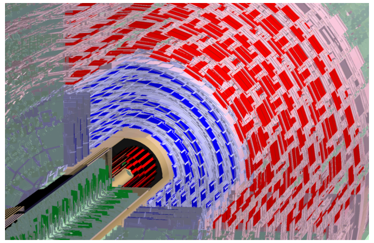
DD4hep geometry toolkit

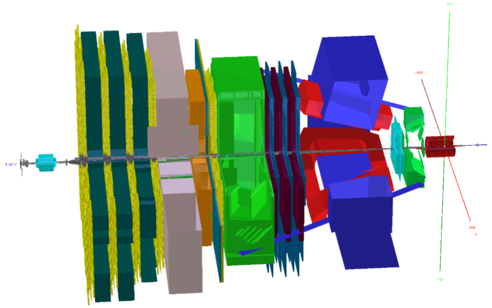
defining the detector geometry and different views on it

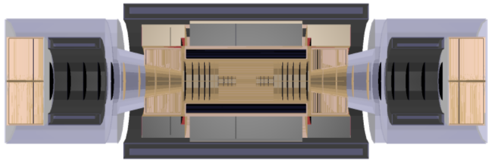
- LC community and CERN have developed a generic detector geometry system - based on best practises by ILC, CLIC, LHCb (*in AIDA, AIDA2020*)
- supporting the full life cycle of the experiment
- providing components and interfaces for
  - full simulation, reconstruction, conditions, alignment, visualisation and analysis
- adopted also by CMS and LHCb

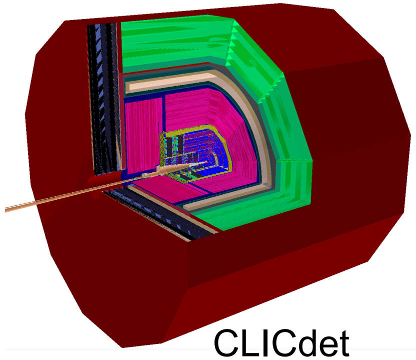


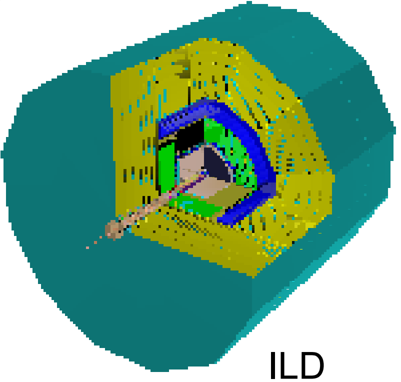
DD4hep: de facto industry standard

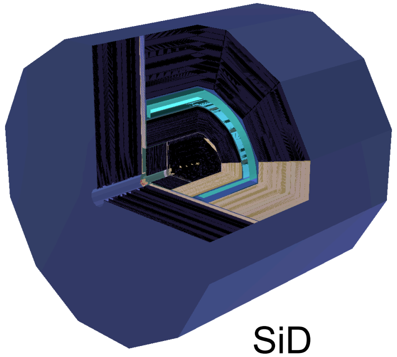

CMS


LHCb


FCC-hh

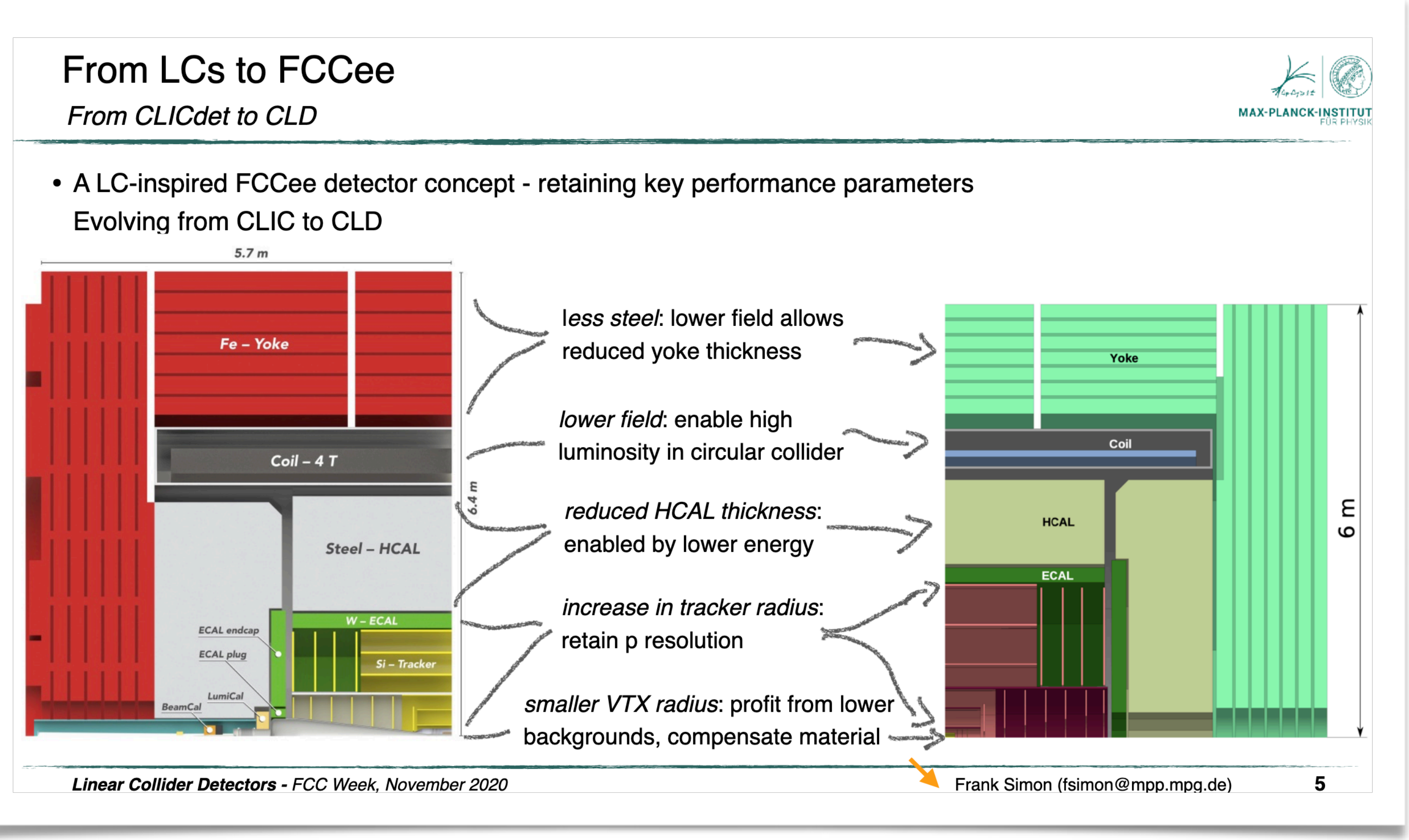

CLICdet


ILD


SiD

DD4hep detector models for FCCee

all Higgs factory detectors in new package k4geo



From LCs to FCCee

From CLICdet to CLD

- A LC-inspired FCCee detector concept - retaining key performance parameters
  Evolving from CLIC to CLD

less steel: lower field allows reduced yoke thickness

lower field: enable high luminosity in circular collider

reduced HCAL thickness: enabled by lower energy

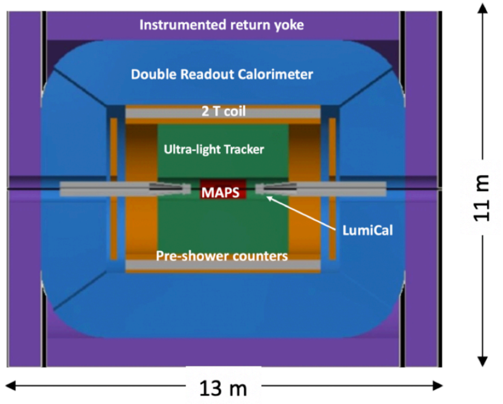
increase in tracker radius: retain p resolution

smaller VTX radius: profit from lower backgrounds, compensate material

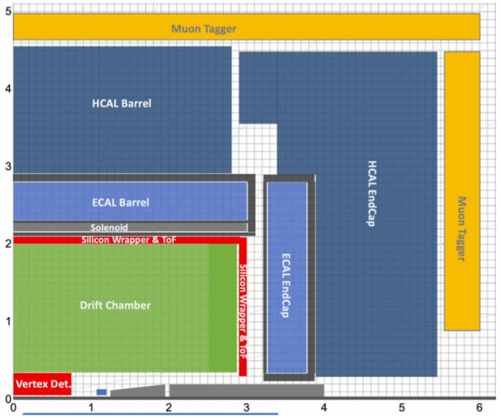
Linear Collider Detectors *- FCC Week, November 2020* Frank Simon (fsimon@mpp.mpg.de) 5

- CLD detector: based on CLICdet
  - adjusted for FCCee at lower energies and lower B-field:
  - larger tracker, thinner calorimeters,….

IDEA



Noble Liquid ECAL based

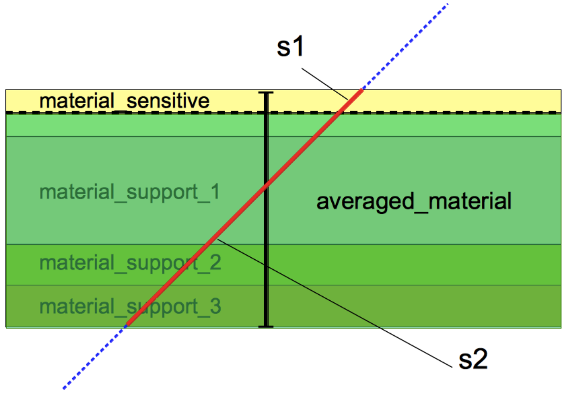


- ongoing work to implement the other two FCCee detector models in DD4hep:
  - IDEA w/ drift chamber and
    - dual readout calorimeter
    - LAr/Noble Liquid calorimeter

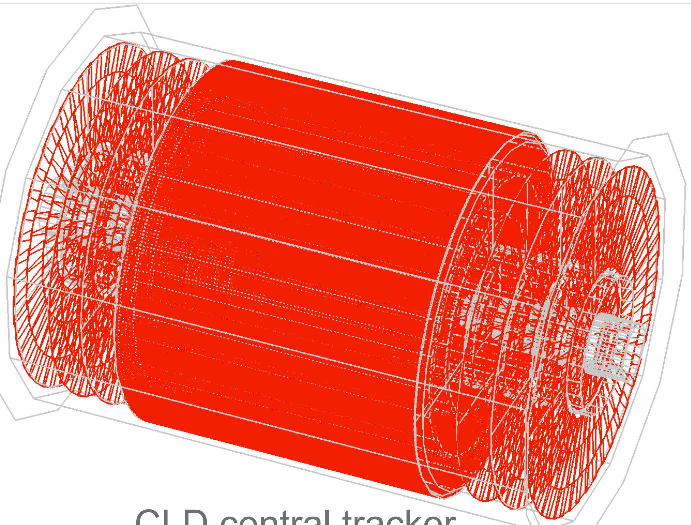
DDRec

interfaces for reconstruction in DD4hep

- DD4hep provides access to the detector geometry as needed in typical reconstruction algorithms in DDRec:
  - **tracking surfaces** attached to sensitive and dead material volumes in detailed model
  - material *automatically* averaged for multiple scattering and E-loss
  - measurement directions on surface
- dedicated **high level reco API** for common sub detectors:
  - e.g. *LayeredCalorimeterData*:
    - positions of absorber and sensitive layers
    - cell dimensions
    - symmetry (barrel, endcap)
  - also automatically extracted from detailed model
- used for example in MarlinPandora to describe the calo geometry



ILD inner tracker

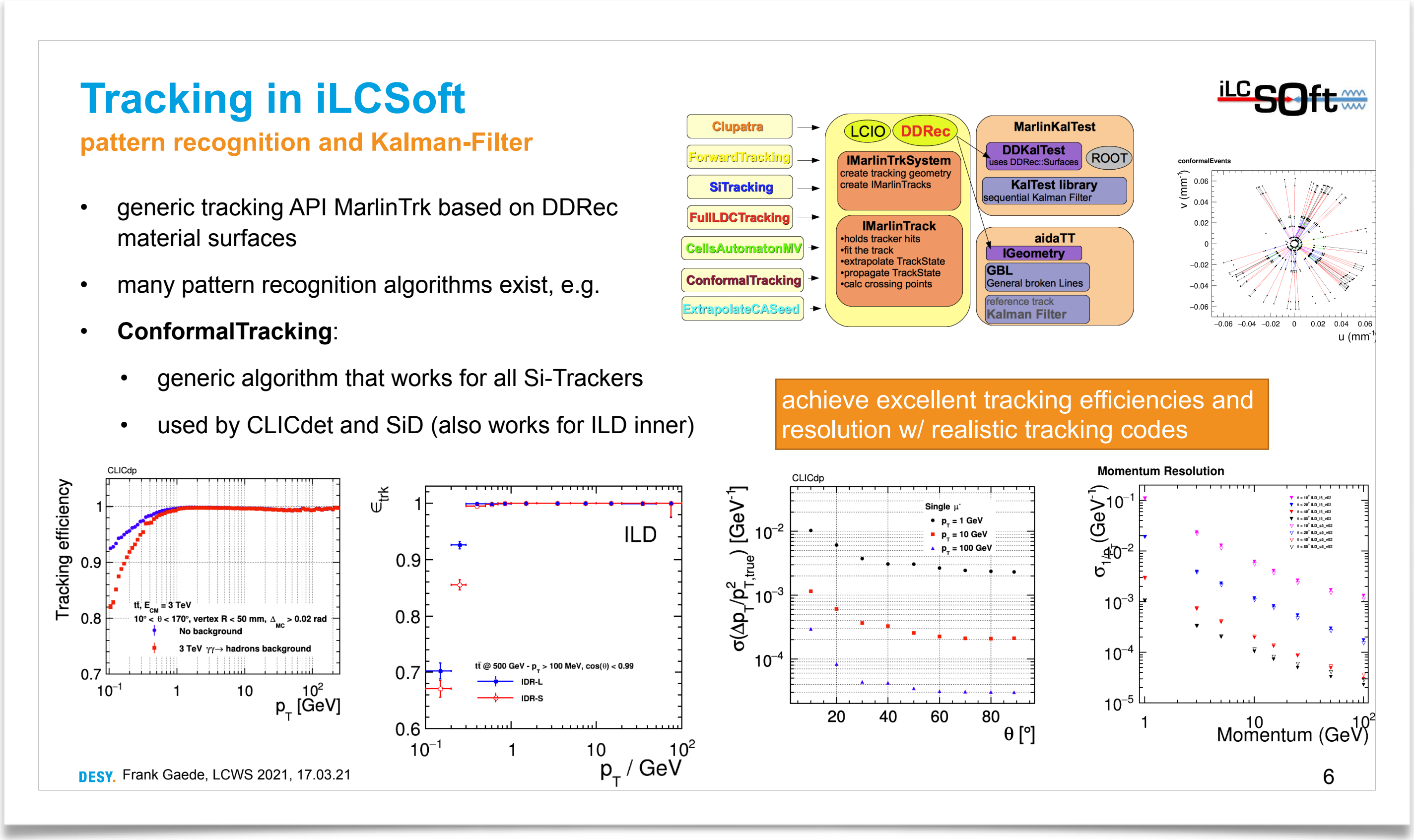


CLD central tracker

large reconstruction code base in iLCSoft

Developed over >15 years for (linear) lepton colliders

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
  - generic API for fitting algorithms
  - large number of pattern recognition algorithms



Tracking in iLCSoft

pattern recognition and Kalman-Filter

- generic tracking API MarlinTrk based on DDRec material surfaces
- many pattern recognition algorithms exist, e.g.
- **ConformalTracking**:
  - generic algorithm that works for all Si-Trackers
  - used by CLICdet and SiD (also works for ILD inner)

Clupatra
ForwardTracking
SiTracking
FullLDCTracking
CellsAutomatonMV
ConformalTracking
ExtrapolateCASeed

LCIO DDRec

IMarlinTrkSystem
create tracking geometry
create IMarlinTracks

IMarlinTrack
•holds tracker hits
•fit the track
•extrapolate TrackState
•propagate TrackState
•calc crossing points

MarlinKalTest
DDKalTest
uses DDRec::Surfaces
ROOT
KalTest library
sequential Kalman Filter

aidaTT
IGeometry
GBL
General broken Lines
reference track
Kalman Filter

achieve excellent tracking efficiencies and resolution w/ realistic tracking codes

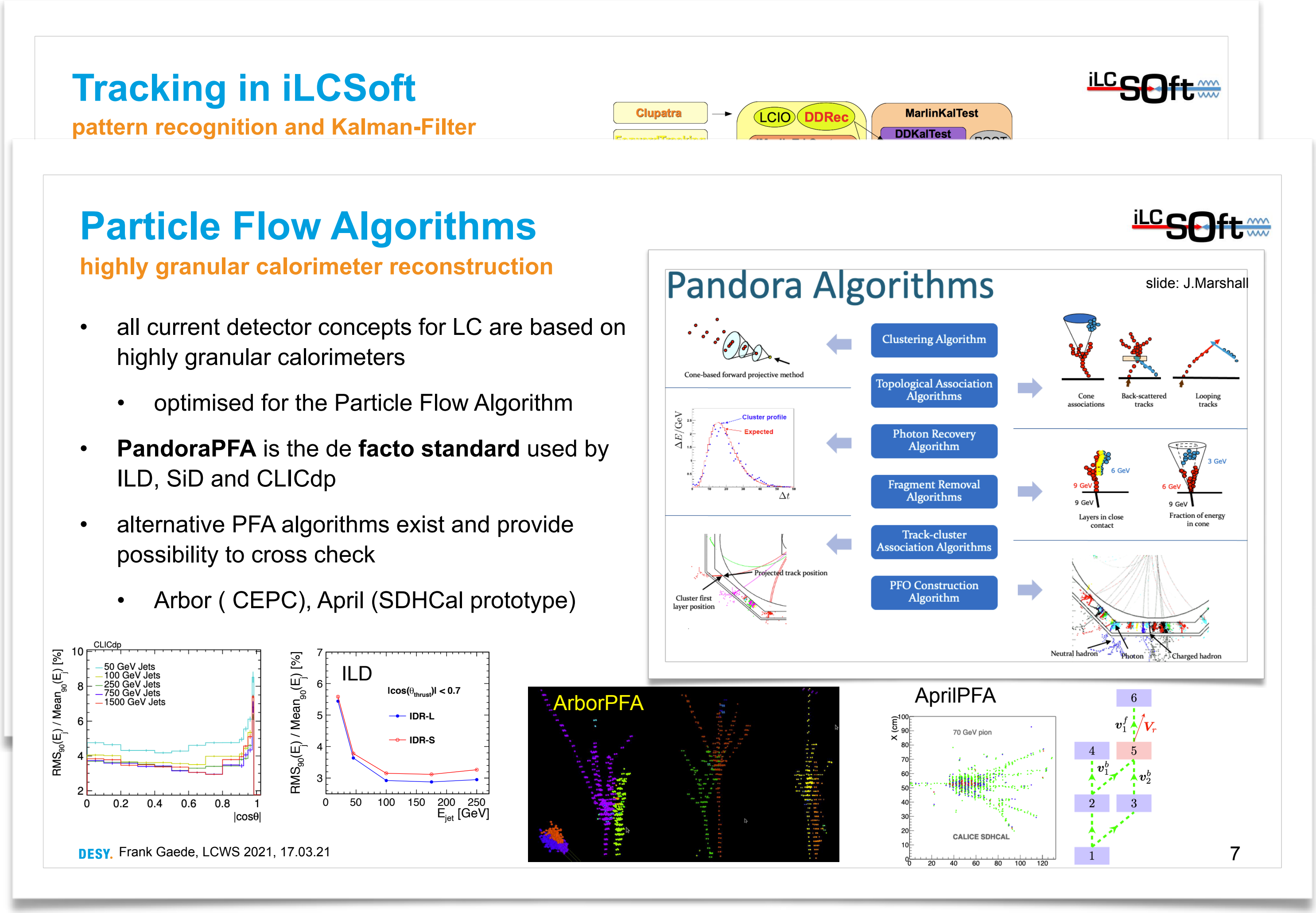
Frank Gaede, LCWS 2021, 17.03.21

6

large reconstruction code base in iLCSoft

Developed over >15 years for (linear) lepton colliders

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
  - generic API for fitting algorithms
  - large number of pattern recognition algorithms
- particle flow algorithms
  - PandoraPFA ans Arbor, AprilPFA



Tracking in iLCSoft

pattern recognition and Kalman-Filter

Particle Flow Algorithms

highly granular calorimeter reconstruction

- all current detector concepts for LC are based on highly granular calorimeters
  - optimised for the Particle Flow Algorithm
- **PandoraPFA** is the de **facto standard** used by ILD, SiD and CLICdp
- alternative PFA algorithms exist and provide possibility to cross check
  - Arbor (CEPC), April (SDHCal prototype)

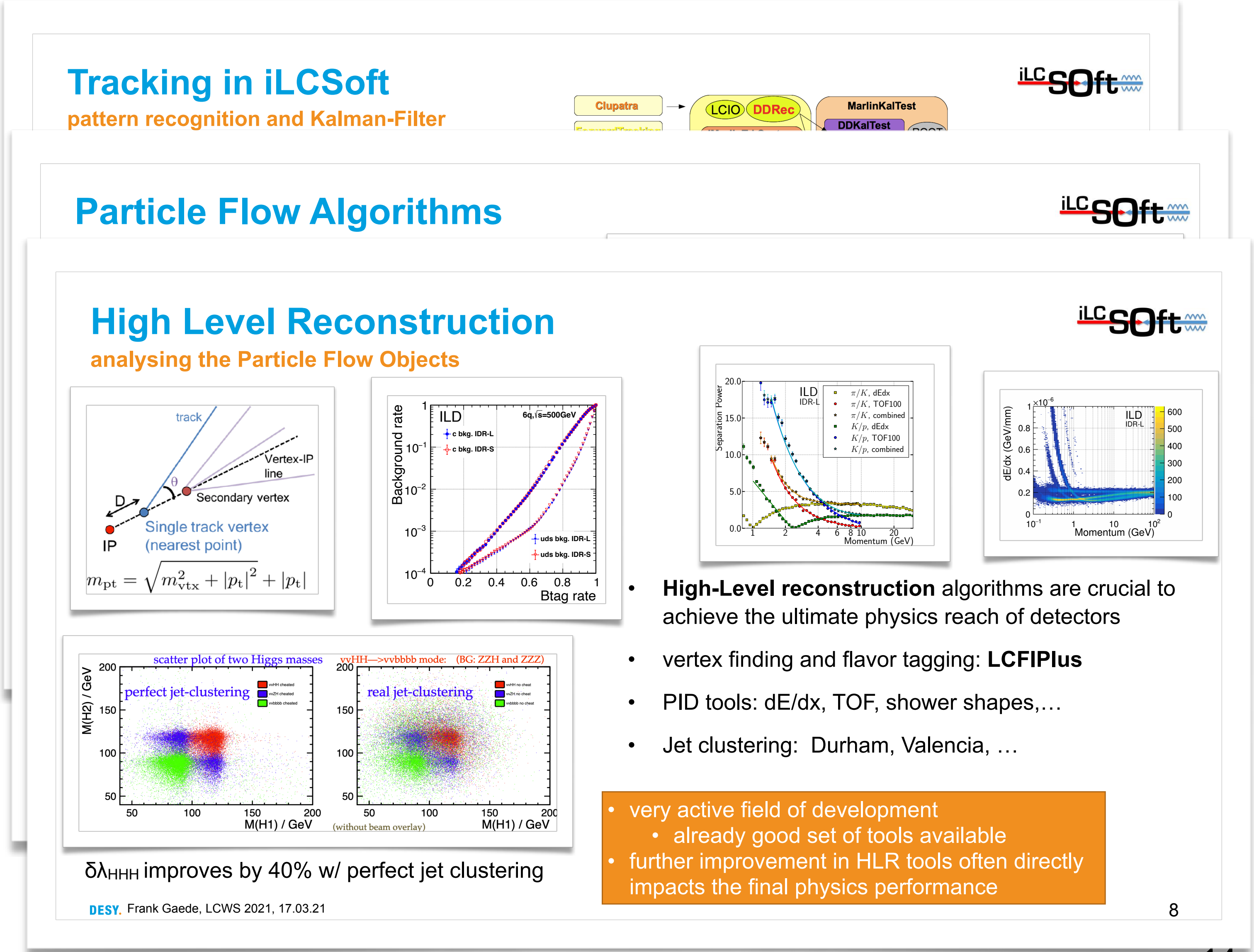
Frank Gaede, LCWS 2021, 17.03.21

7

large reconstruction code base in iLCSoft

Developed over >15 years for (linear) lepton colliders

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
  - generic API for fitting algorithms
  - large number of pattern recognition algorithms
- particle flow algorithms
  - PandoraPFA ans Arbor, AprilPFA
- high level reconstruction
  - jet finding, flavor tagging, PID, TOF,…



Tracking in iLCSoft

pattern recognition and Kalman-Filter

Particle Flow Algorithms

High Level Reconstruction

analysing the Particle Flow Objects

δλ$_{HHH}$ improves by 40% w/ perfect jet clustering

- **High-Level reconstruction** algorithms are crucial to achieve the ultimate physics reach of detectors
- vertex finding and flavor tagging: **LCFIPlus**
- PID tools: dE/dx, TOF, shower shapes,…
- Jet clustering: Durham, Valencia, …

- very active field of development
  - already good set of tools available
- further improvement in HLR tools often directly impacts the final physics performance

DESY. Frank Gaede, LCWS 2021, 17.03.21

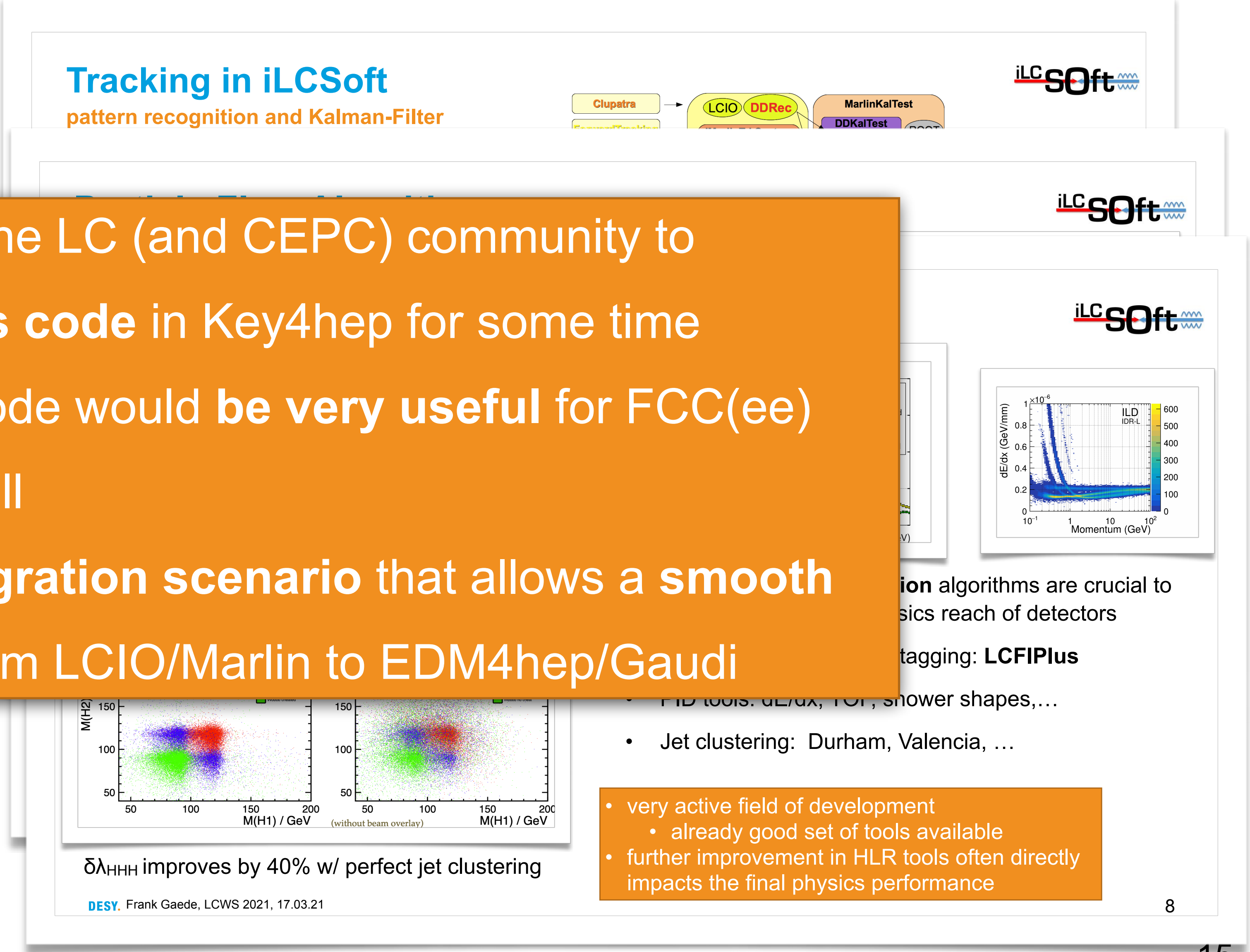
8

large reconstruction code base in iLCSoft

Developed over >15 years for (linear) lepton colliders

- realistic detector models for incl. tracking/reconstruction geometry
- track reconstruction
  - generic API for f
  - large number of recognition algo
- particle flow algorith
  - PandoraPFA ar
- high level reconstru
  - jet finding, flavor tagging, PID, TOF,…

- it is **vital** for the LC (and CEPC) community to **preserve this code** in Key4hep for some time
- a lot of this code would **be very useful** for FCC(ee) studies as well
- -> need a **migration scenario** that allows a **smooth transition** from LCIO/Marlin to EDM4hep/Gaudi



Tracking in iLCSoft

pattern recognition and Kalman-Filter

ion algorithms are crucial to
sics reach of detectors

tagging: **LCFIPlus**

- PID tools: dE/dx, TOF, shower shapes,…
- Jet clustering: Durham, Valencia, …

- very active field of development
  - already good set of tools available
- further improvement in HLR tools often directly impacts the final physics performance

$\delta\lambda_{HHH}$ improves by 40% w/ perfect jet clustering

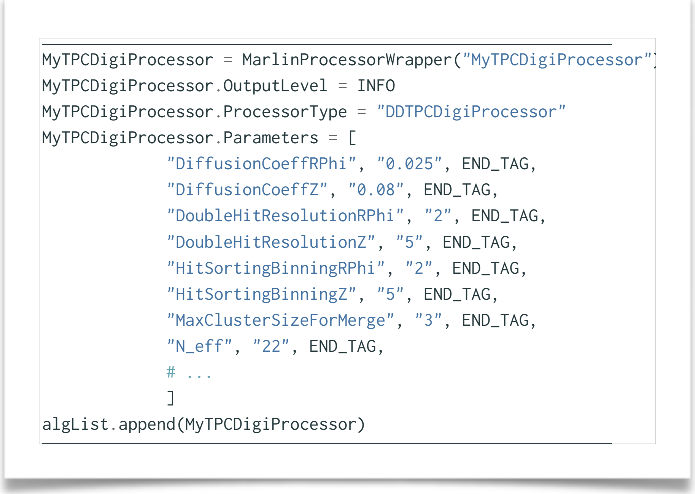
DESY. Frank Gaede, LCWS 2021, 17.03.21 8

k4MarlinWrapper

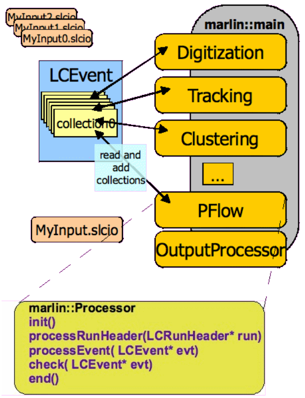
running Marlin processors in Gaudi (Key4hep)

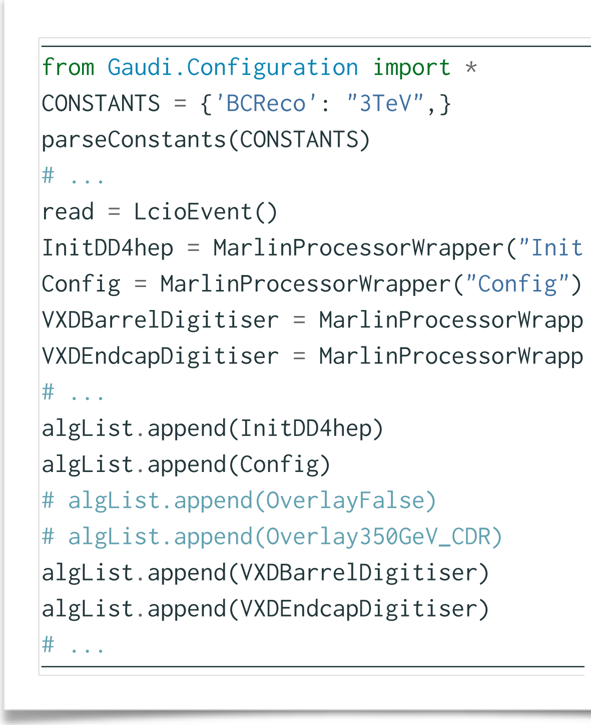
- set of Gaudi algorithms that wrap Marlin processors
  - developed by CERN-SFT
  - automatic XML to Python steering file conversion
- tools for automatic in-memory, on-demand conversion between LCIO and EDM4hep
  - developed by IHEP, CERN, DESY
  - possibility to mix Marlin processors with genuine Gaudi 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*

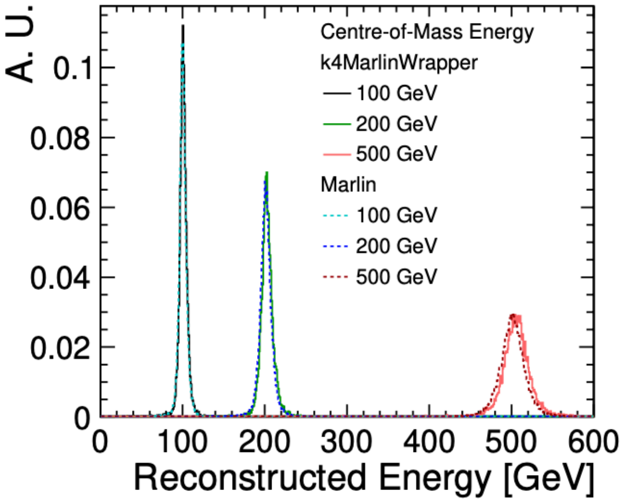


```
MyTPCDigiProcessor = MarlinProcessorWrapper("MyTPCDigiProcessor")
MyTPCDigiProcessor.OutputLevel = INFO
MyTPCDigiProcessor.ProcessorType = "DDTPCDigiProcessor"
MyTPCDigiProcessor.Parameters = [
                    "DiffusionCoeffRPhi", "0.025", END_TAG,
                    "DiffusionCoeffZ", "0.08", 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)
```



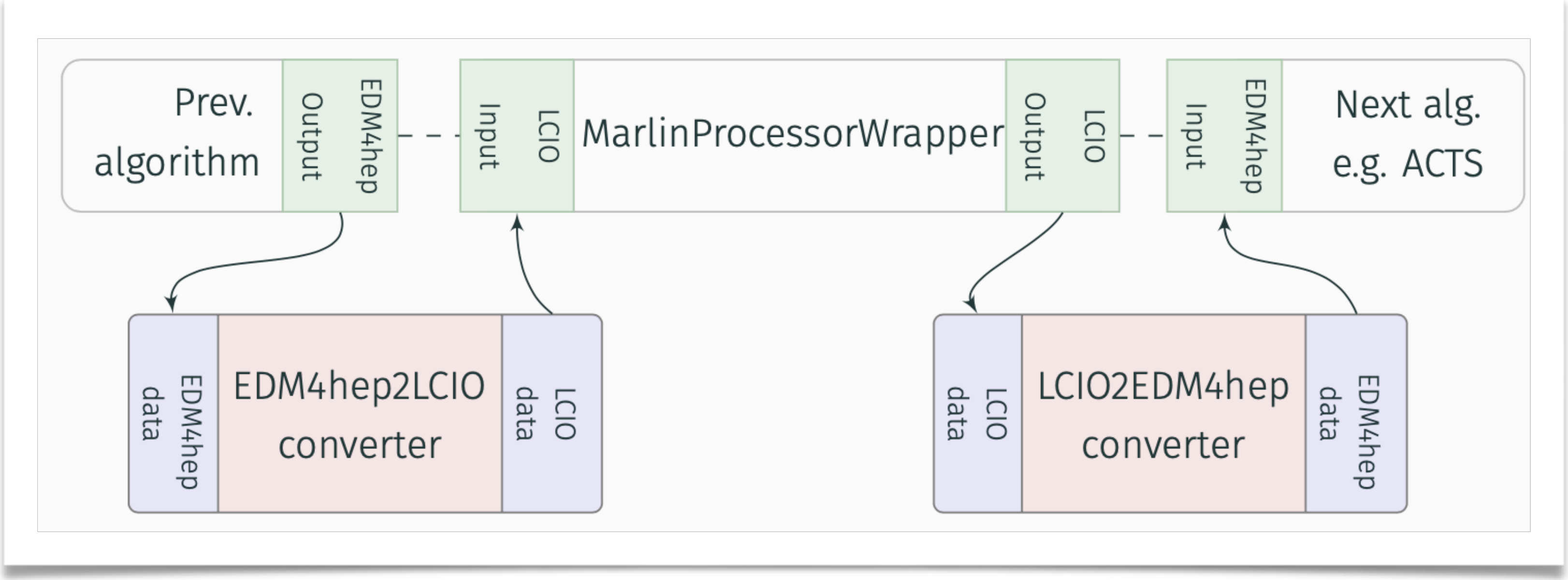


```
from Gaudi.Configuration import *
CONSTANTS = {'BCReco': "3TeV",}
parseConstants(CONSTANTS)
# ...
read = LcioEvent()
InitDD4hep = MarlinProcessorWrapper("Init
Config = MarlinProcessorWrapper("Config")
VXDBarrelDigitiser = MarlinProcessorWrapp
VXDEndcapDigitiser = MarlinProcessorWrapp
# ...
algList.append(InitDD4hep)
algList.append(Config)
# algList.append(OverlayFalse)
# algList.append(Overlay350GeV_CDR)
algList.append(VXDBarrelDigitiser)
algList.append(VXDEndcapDigitiser)
# ...
```



K4MarlinWrappper

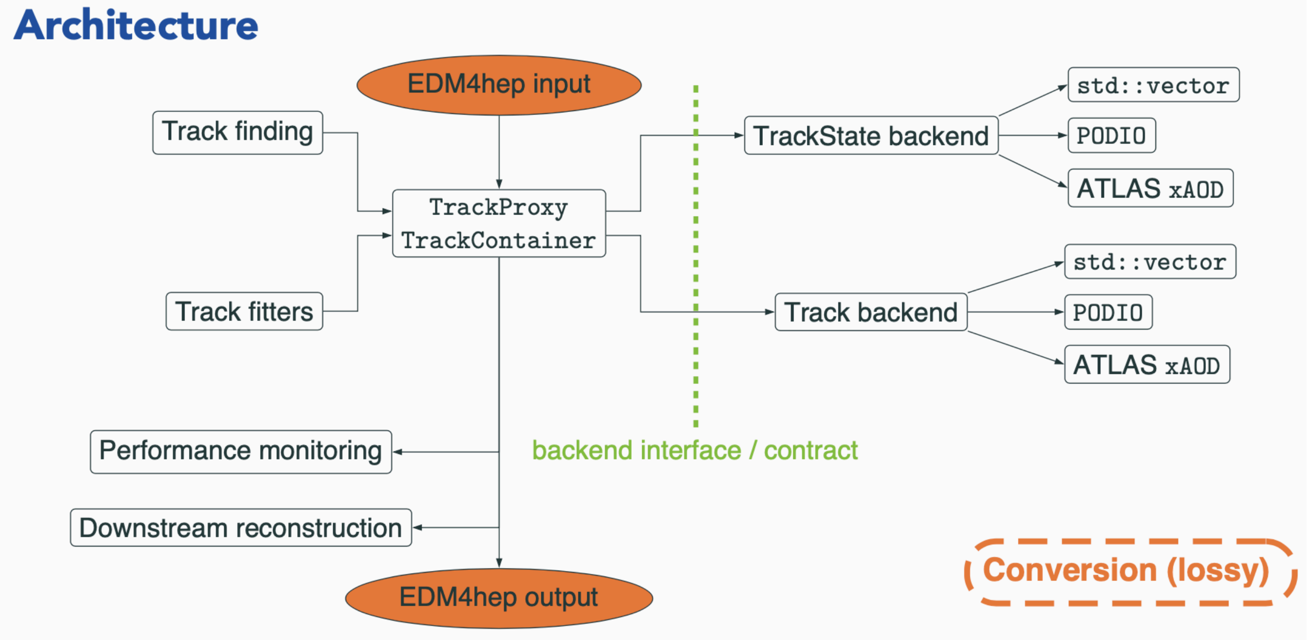
the vision: mix and match Marlin and Gaudi algorithms



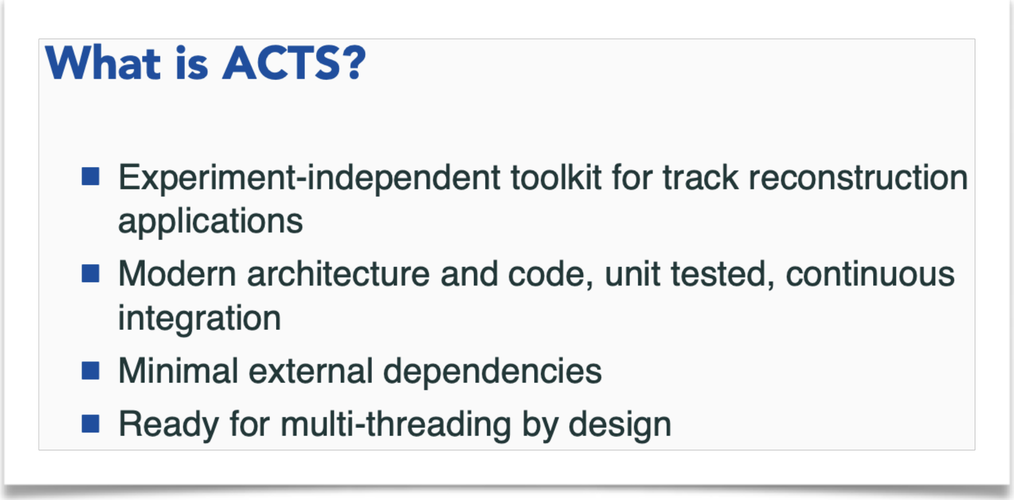
- in a transition phase algorithms developed in the new EDM4hep/Gaudi world can gradually replace older algorithms
    - e.g. eventually one might want to replace track fitting with **ACTS** also for LC detectors
- some technicalities have to be address *under-the-hood*
    - see also next talk by T. Madlener

ACTS

a common tracking toolkit



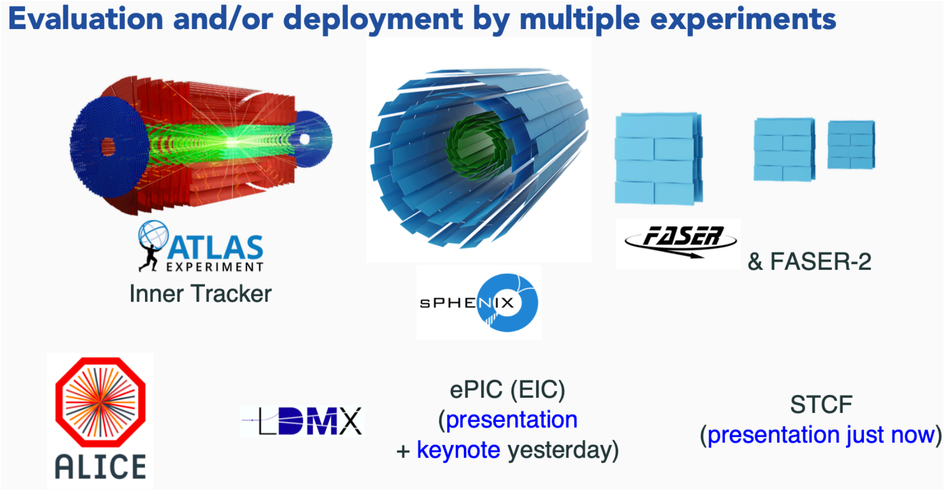
- ACTS tracking toolkit is the the current choice for track fitting (and finding !?) in Key4hep
- recently implemented interface to write out EDM4hep Tracks
  - some discussion needed w/ tracking and ACTS experts on details of the tracking data model
  - perigee vs. on-surface parameterisation …



What is ACTS?

- Experiment-independent toolkit for track reconstruction applications
- Modern architecture and code, unit tested, continuous integration
- Minimal external dependencies
- Ready for multi-threading by design

P.Gessinger, CHEP 2023



Evaluation and/or deployment by multiple experiments

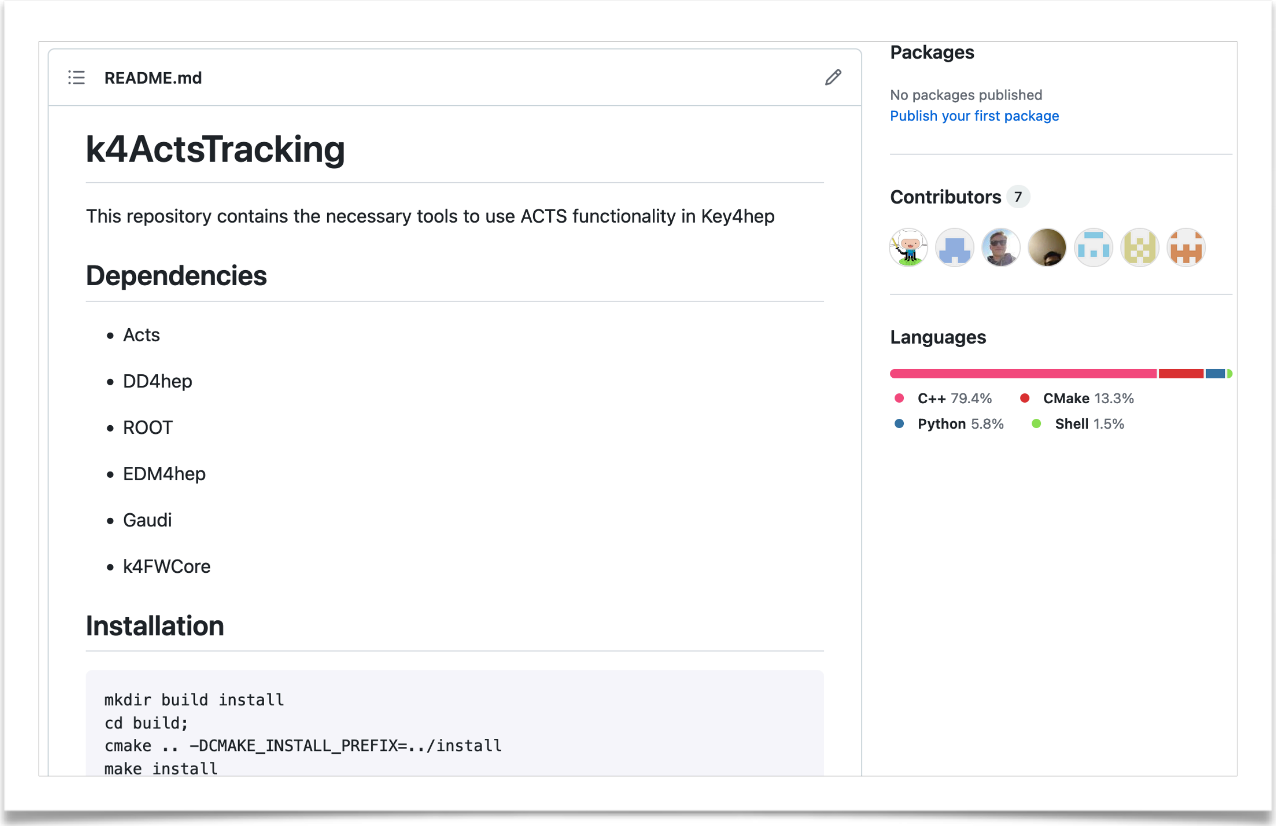
ePIC (EIC)
(presentation
+ keynote yesterday)

STCF
(presentation just now)

k4ACTS

integration of ACTS in Key4hep

- basic package infrastructure exists
- no real implementation of tracking example in key4hep yet
- some work planed at CERN (L.Reichenbach) in context of electron reconstruction w/ ACTS
  - some examples for track fitting in ACTS w/ a DD4hep detector (OpenDetector) exist
- crucial is the interface to the tracking geometry
  - ACTS has interface to DD4hep to extract surface geometry
  - need to check compatibility w/ *ddrec::Surface* used in LC tracking



README.md

k4ActsTracking

This repository contains the necessary tools to use ACTS functionality in Key4hep

Dependencies

- Acts
- DD4hep
- ROOT
- EDM4hep
- Gaudi
- k4FWCore

Installation

```
mkdir build install
cd build;
cmake .. -DCMAKE_INSTALL_PREFIX=../install
make install
```

Packages

No packages published
Publish your first package

Contributors 7

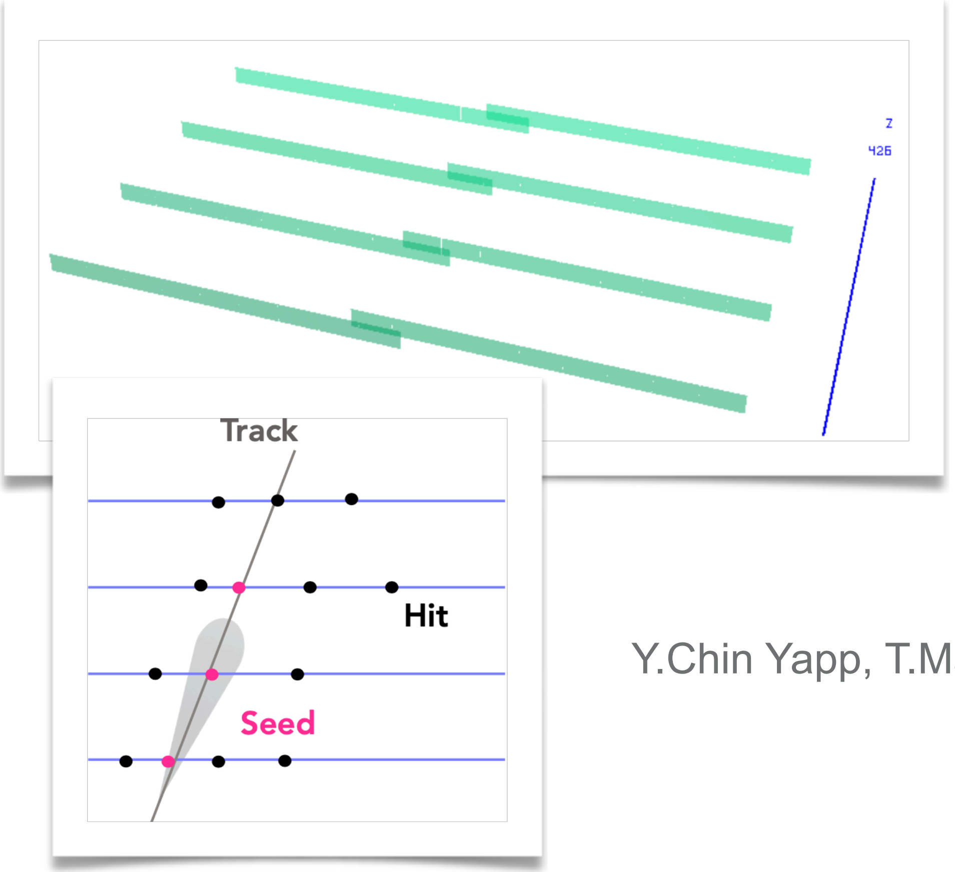
Languages

● C++ 79.4% ● CMake 13.3%
● Python 5.8% ● Shell 1.5%

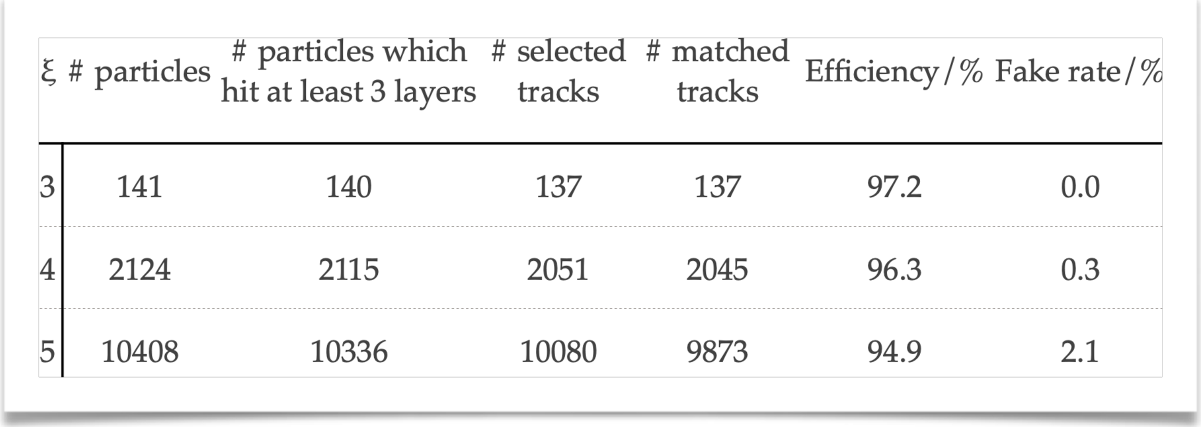
see also talk: K.Krizka on ACTS tracking for muon collider later today

ACTS for LUXE

- LUXE is a planned high field QED experiment at DESY
  - colliding a strong laser with e- beam from EU-XFEL
- LUXE will use the Key4hep stack
  - implemented the rather simple tracker - 4 layers of Si-sensors in spectrometer setup
  - use of tracking geometry conversion provided by ACTS
  - using CKF code from ACTS achieve good tracking efficiencies at “high” multiplicities
- ideal testing ground for ACTS integration in Key4hep - tracking geometry and EDM4hep



Y.Chin Yapp, T.Madlener

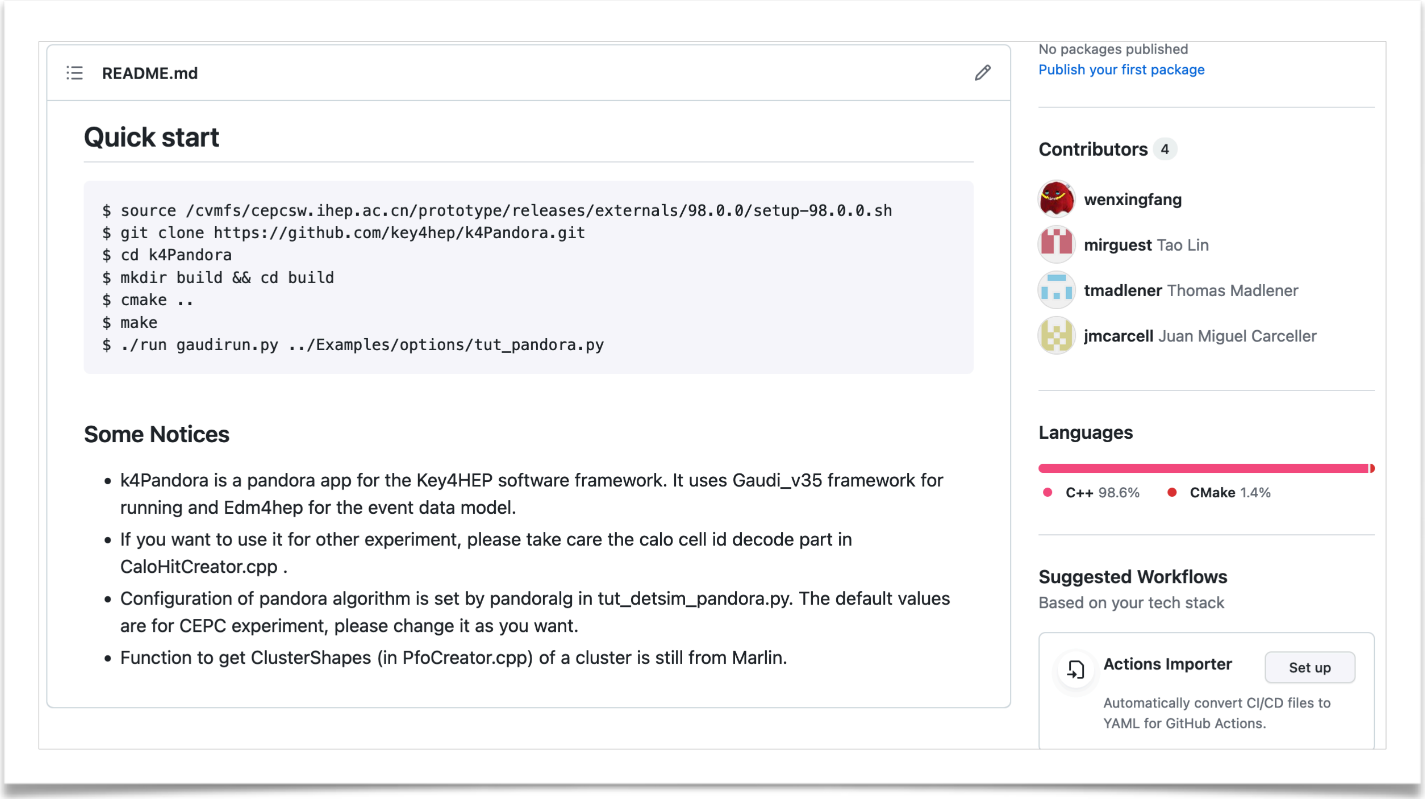


ξ	# particles	# particles which hit at least 3 layers	# selected tracks	# matched tracks	Efficiency / %	Fake rate / %
3	141	140	137	137	97.2	0.0
4	2124	2115	2051	2045	96.3	0.3
5	10408	10336	10080	9873	94.9	2.1

k4Pandora

interfacing PandoraPFA to Key4hep

- first implementation exists
- implemented by CEPC colleagues
  - currently only targeted at CEPC
  - uses old GEAR geometry description
- need to generalise for all future collider detectors that have a DD4hep geometry model
- should use *ddrec::LayeredCalorimeterData* classes consistently
- work started/planed at CERN (S.Sassikumar):
- investigate PandoraPFA for LAr calorimeter reconstruction
  - add LAr calorimeter to CLD detector



README.md

Quick start

```
$ source /cvmfs/cepcsw.ihep.ac.cn/prototype/releases/externals/98.0.0/setup-98.0.0.sh
$ git clone https://github.com/key4hep/k4Pandora.git
$ cd k4Pandora
$ mkdir build && cd build
$ cmake ..
$ make
$ ./run gaudirun.py ../Examples/options/tut_pandora.py
```

Some Notices

- k4Pandora is a pandora app for the Key4HEP software framework. It uses Gaudi_v35 framework for running and Edm4hep for the event data model.
- If you want to use it for other experiment, please take care the calo cell id decode part in CaloHitCreator.cpp .
- Configuration of pandora algorithm is set by pandoralg in tut_detsim_pandora.py. The default values are for CEPC experiment, please change it as you want.
- Function to get ClusterShapes (in PfoCreator.cpp) of a cluster is still from Marlin.

No packages published
Publish your first package

Contributors 4

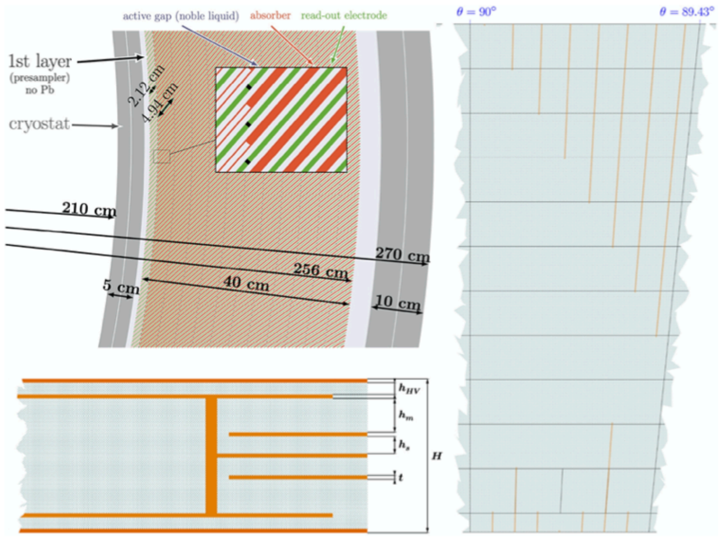
- wenxingfang
- mirguest Tao Lin
- tmadlener Thomas Madlener
- jmcarcell Juan Miguel Carceller

Languages

- C++ 98.6%
- CMake 1.4%

Suggested Workflows
Based on your tech stack

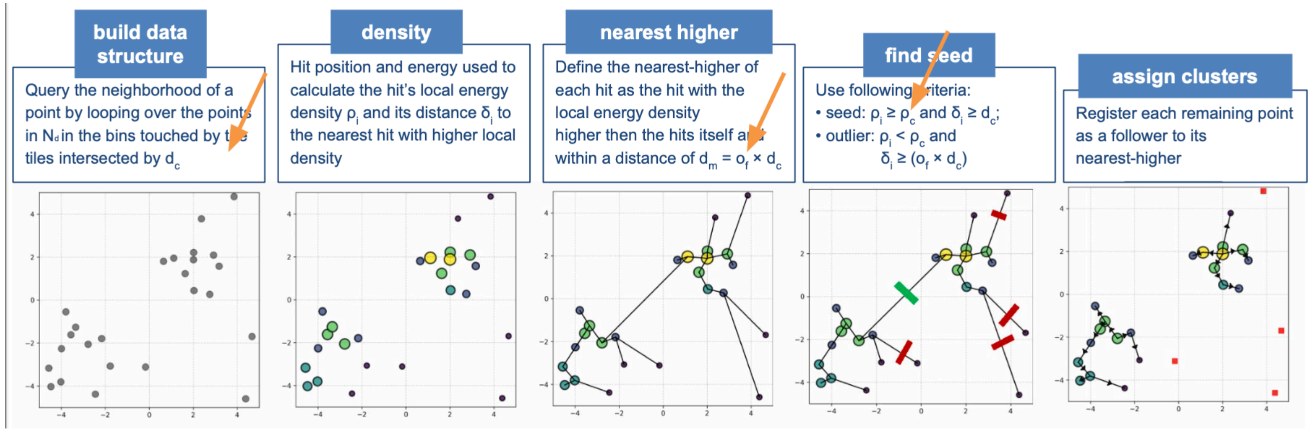
Actions Importer — Set up
Automatically convert CI/CD files to YAML for GitHub Actions.



k4Clue

CLUStering Energy

- originally developed for CMS HGCal:
  - fast clustering for high granular calorimeters - based on local energy density
- ported to Key4hep and extended to 4pi geometry
- dedicated tuning of clustering parameters for CLD and LAr ECal
  - shows performance comparable to pre-existing dedicated algorithms
- versatile clustering algorithm for a variety of different calorimeter technologies



build data structure
Query the neighborhood of a point by looping over the points in N_d in the bins touched by the tiles intersected by d_c

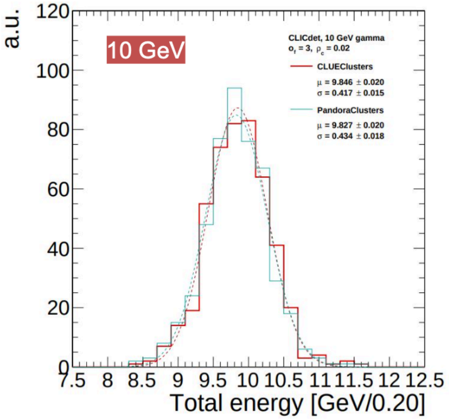
density
Hit position and energy used to calculate the hit's local energy density ρ_i and its distance δ_i to the nearest hit with higher local density

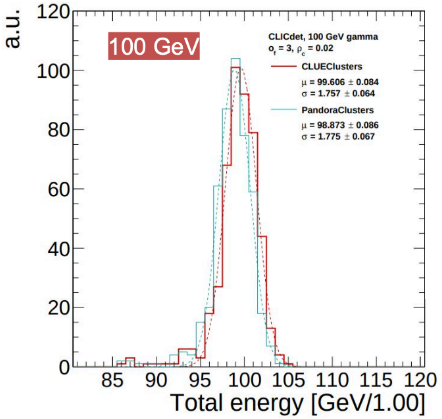
nearest higher
Define the nearest-higher of each hit as the hit with the local energy density higher then the hits itself and within a distance of $d_m = o_f \times d_c$

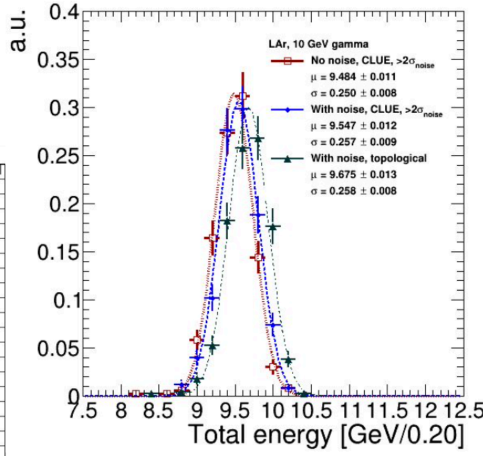
find seed
Use following criteria:
- seed: $\rho_i \geq \rho_c$ and $\delta_i \geq d_c$;
- outlier: $\rho_i < \rho_c$ and $\delta_i \geq (o_f \times d_c)$

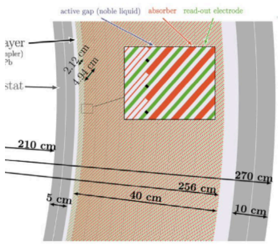
assign clusters
Register each remaining point as a follower to its nearest-higher

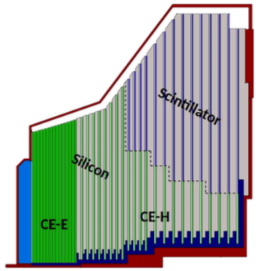
E.Brondolin, CHEP 2023

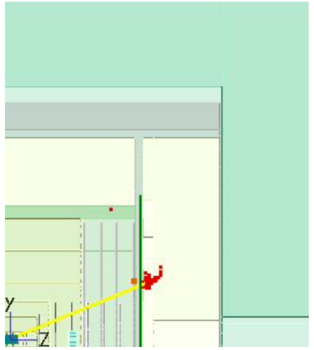






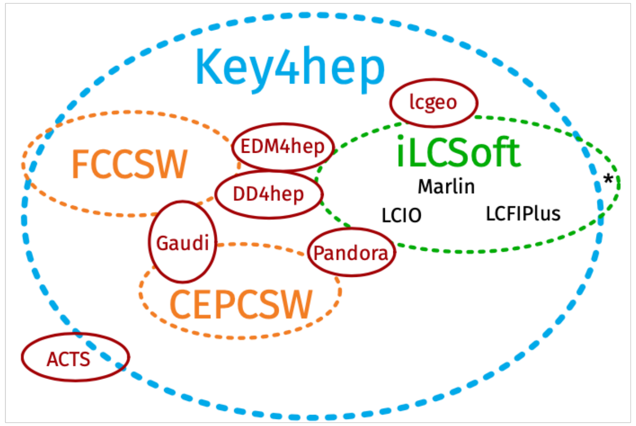






Summary and Outlook

- **Key4hep** started as a new future collider community wide effort in 2020 to put together a modern turnkey software stack
- with growing community of users and contributors: CEPC, CLIC, FCC, EIC, ILC, LUXE, Muon Collider …
    - core tools: DD4hep, EDM4hep(podio), Gaudi
- reconstruction (and simulation/analysis) tools and algorithm from CEPC, FCC and the linear colliders included in Key4hep stack
    - can run complete LC reconstruction w/ MarlinWrapper
    - first genuine Key4hep/EDM4hep/Gaudi reconstruction algorithm start to become available (k4Clue, k4ACTS, k4Pandora,….)

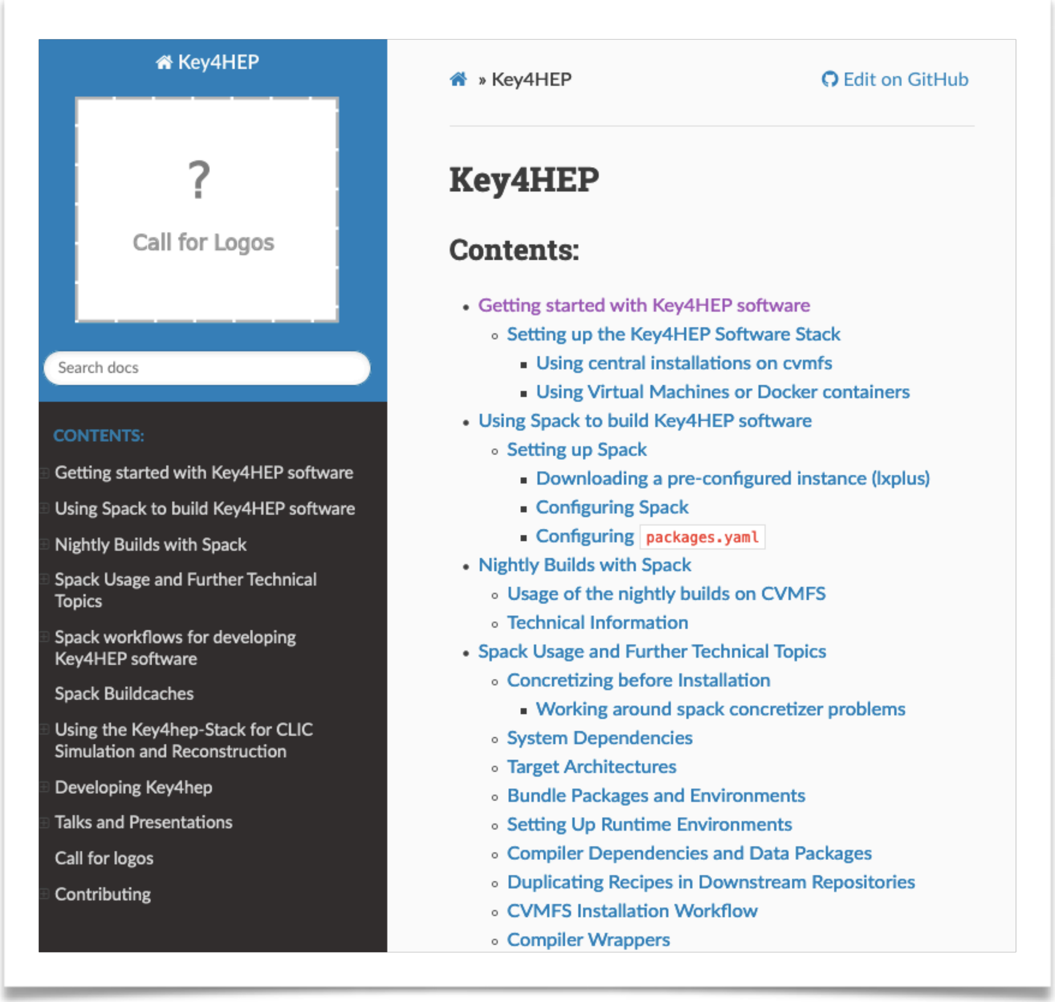


- **Key4hep project is the first time that such a large number of experiments develop a common software stack**
- **Progress crucially depends on contributing person power**
- **Now is a great time to get involved in contributing to Key4hep w/ (high level) reconstruction algorithms**

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



Key4HEP

?

Call for Logos

Search docs

CONTENTS:

- Getting started with Key4HEP software
- Using Spack to build Key4HEP software
- Nightly Builds with Spack
- Spack Usage and Further Technical Topics
- Spack workflows for developing Key4HEP software
- Spack Buildcaches
- Using the Key4hep-Stack for CLIC Simulation and Reconstruction
- Developing Key4hep
- Talks and Presentations
- Call for logos
- Contributing

» Key4HEP

Edit on GitHub

Key4HEP

Contents:

- Getting started with Key4HEP software
  - Setting up the Key4HEP Software Stack
    - Using central installations on cvmfs
    - Using Virtual Machines or Docker containers
- Using Spack to build Key4HEP software
  - Setting up Spack
    - Downloading a pre-configured instance (lxplus)
    - Configuring Spack
    - Configuring `packages.yaml`
- Nightly Builds with Spack
  - Usage of the nightly builds on CVMFS
  - Technical Information
- Spack Usage and Further Technical Topics
  - Concretizing before Installation
    - Working around spack concretizer problems
  - System Dependencies
  - Target Architectures
  - Bundle Packages and Environments
  - Setting Up Runtime Environments
  - Compiler Dependencies and Data Packages
  - Duplicating Recipes in Downstream Repositories
  - CVMFS Installation Workflow
  - Compiler Wrappers