

# (High Level) Reconstruction Tools for ILD

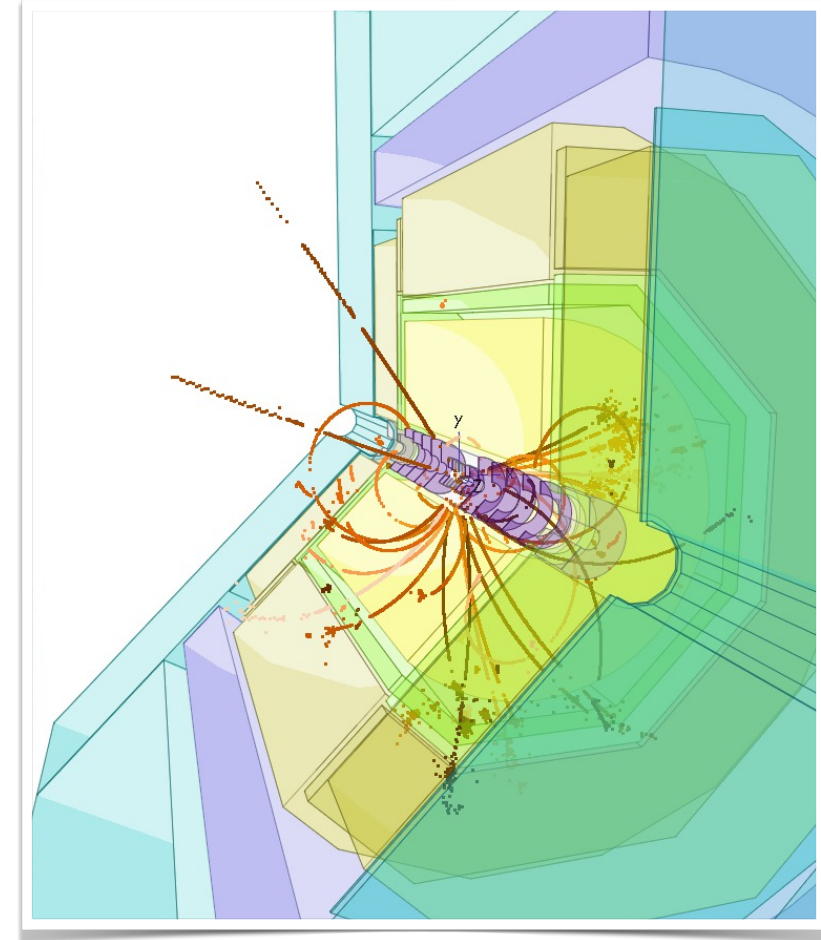
Towards “genuine” use of Key4hep ?

16.01.2024

Frank Gaede, DESY  
ILD Meeting 2024, CERN

# Outline

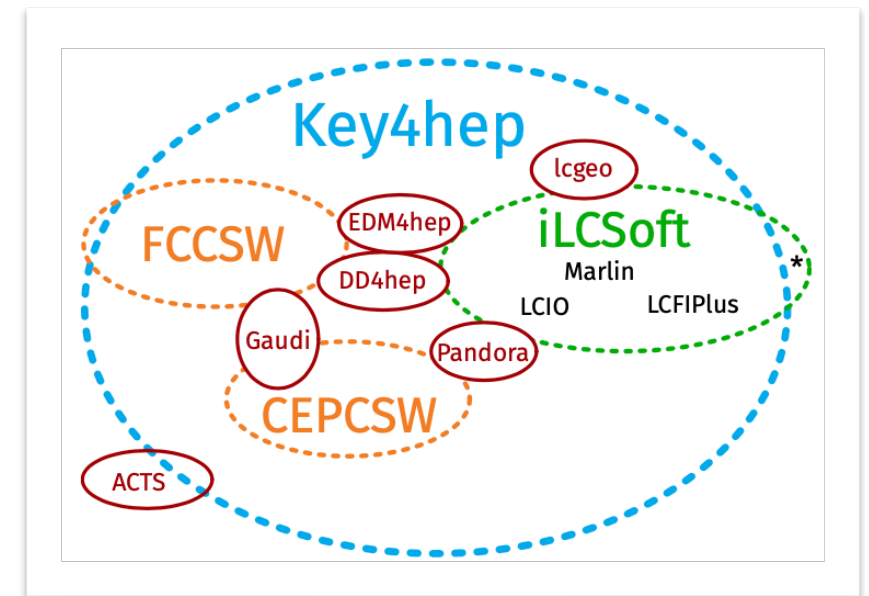
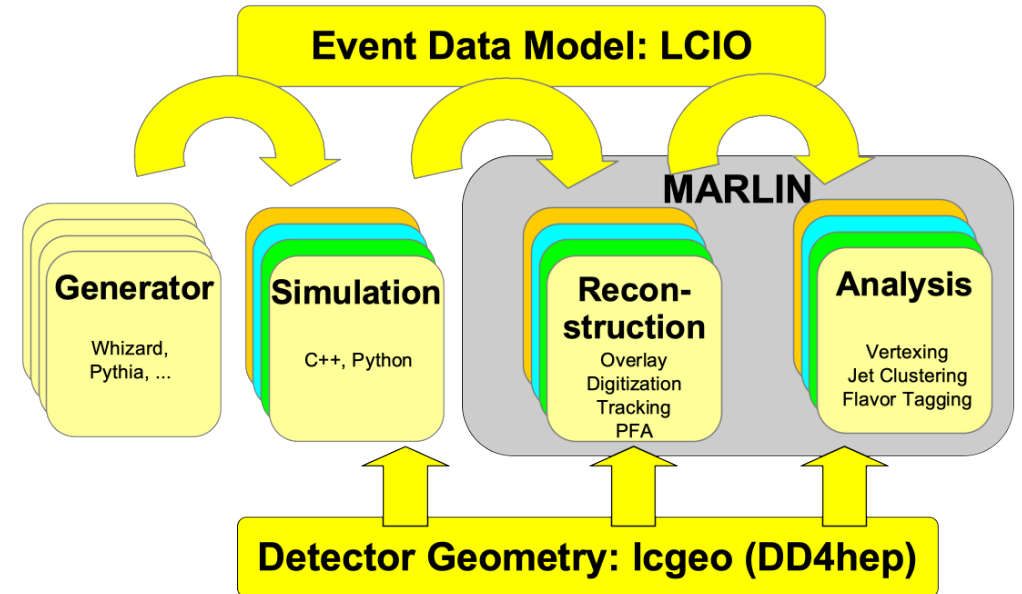
- Introduction and Reminder
  - iLCSoft  $\leftrightarrow$  Key4hep
  - DD4hep detector models and reconstruction
- Standard ILD reconstruction algorithms
- “Transition” to Key4hep
- Recent (HL)R developments
- Conclusion and Outlook



# The common software vision

## iLCSoft as integral part of Key4hep

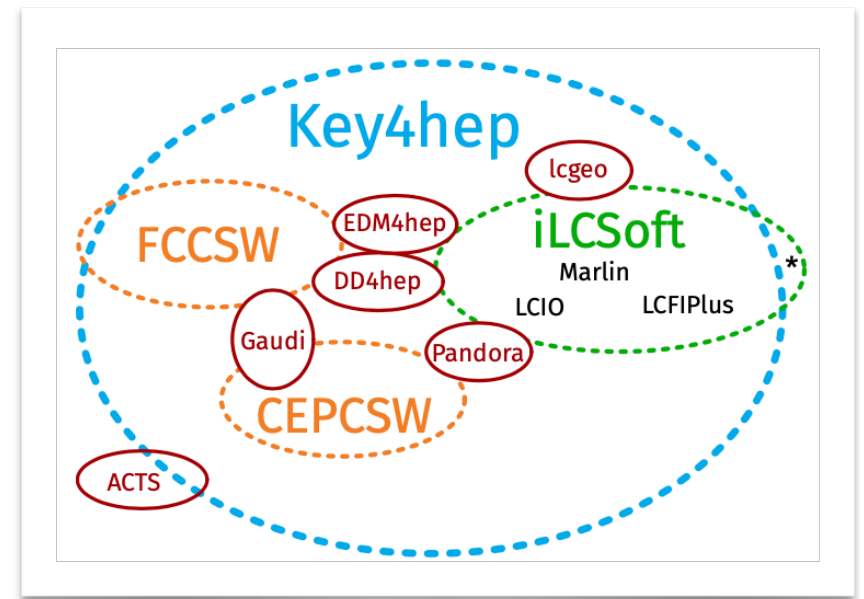
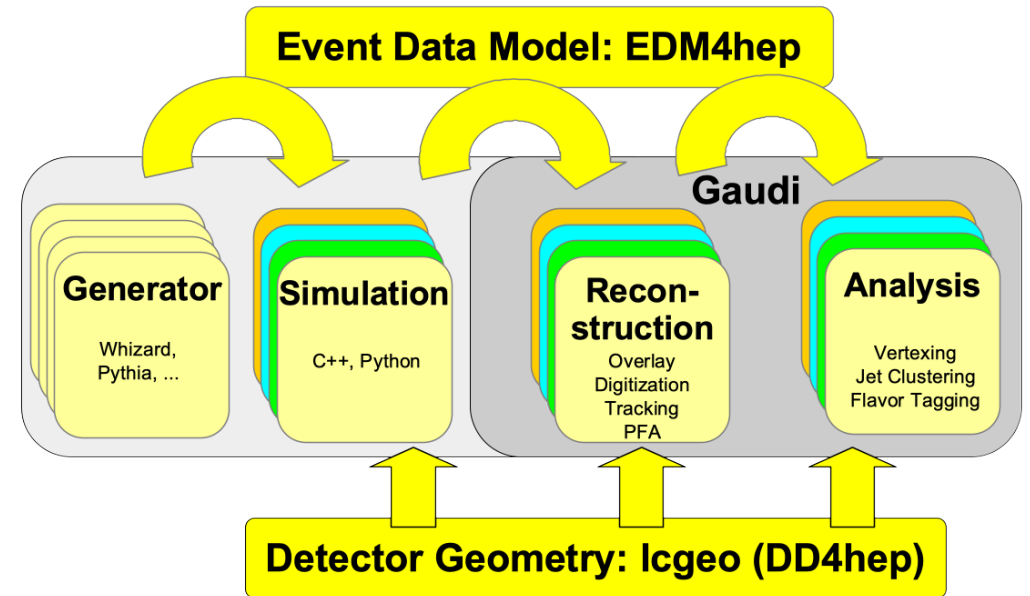
- complete set of tools for
  - **generation, simulation, reconstruction, analysis**
  - build, package, test, deploy
- core ingredients of current **Key4hep**
  - **PODIO** for **EDM4hep** (based on LCIO and FCC-edm)
  - **Gaudi** framework, devel/used for (HL-)LHC
  - **DD4hep** for geometry
    - originally developed for LC now adopted by community
  - **spack** package manager



# The common software vision

## iLCSoft as integral part of Key4hep

- 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

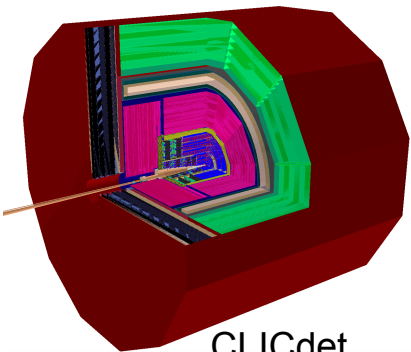




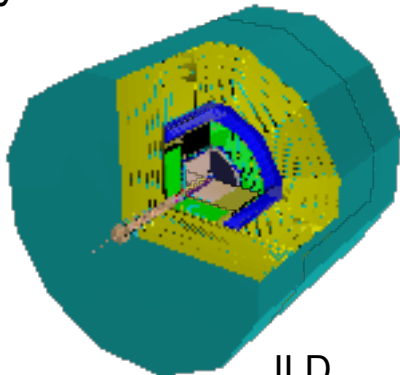
# DD4hep geometry toolkit

defining the detector geometry and different views on it

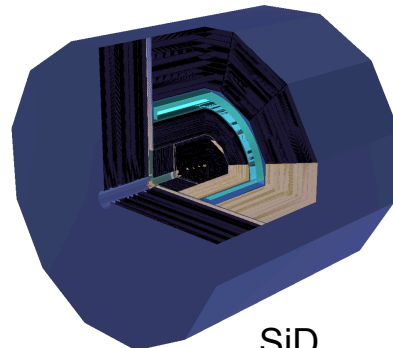
- LC community and CERN have developed a generic detector geometry system - based on best practises by ILC, CLIC, LHCb (*in AIDA, AIDA2020*)
- supporting the full life cycle of the experiment
- providing components and interfaces for
  - full simulation, reconstruction, conditions, alignment, visualisation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



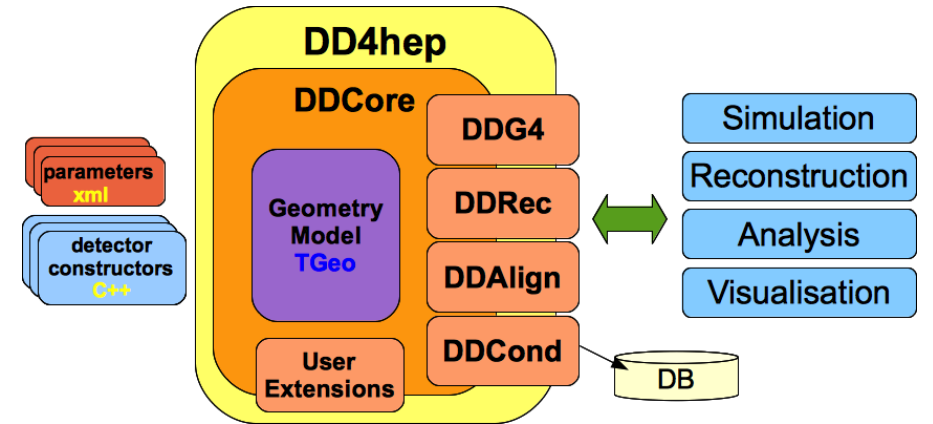
CLICdet



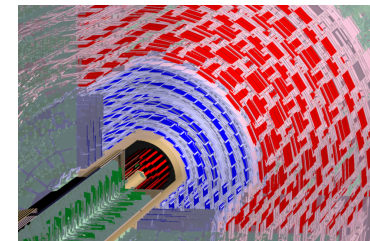
ILD



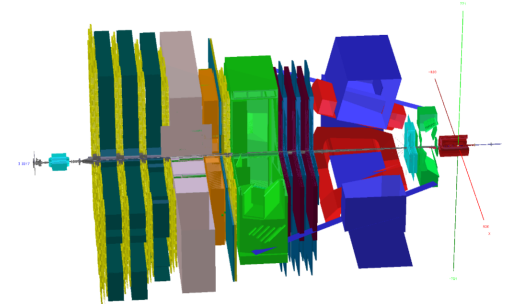
SiD



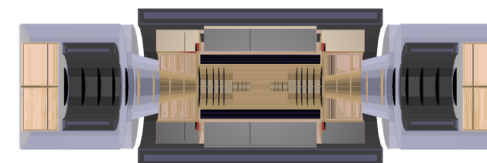
DD4hep: de facto industry standard



CMS



LHCb



FCC-hh

# DD4hep detector models for FCCee

all Higgs factory detectors in new package *k4geo*

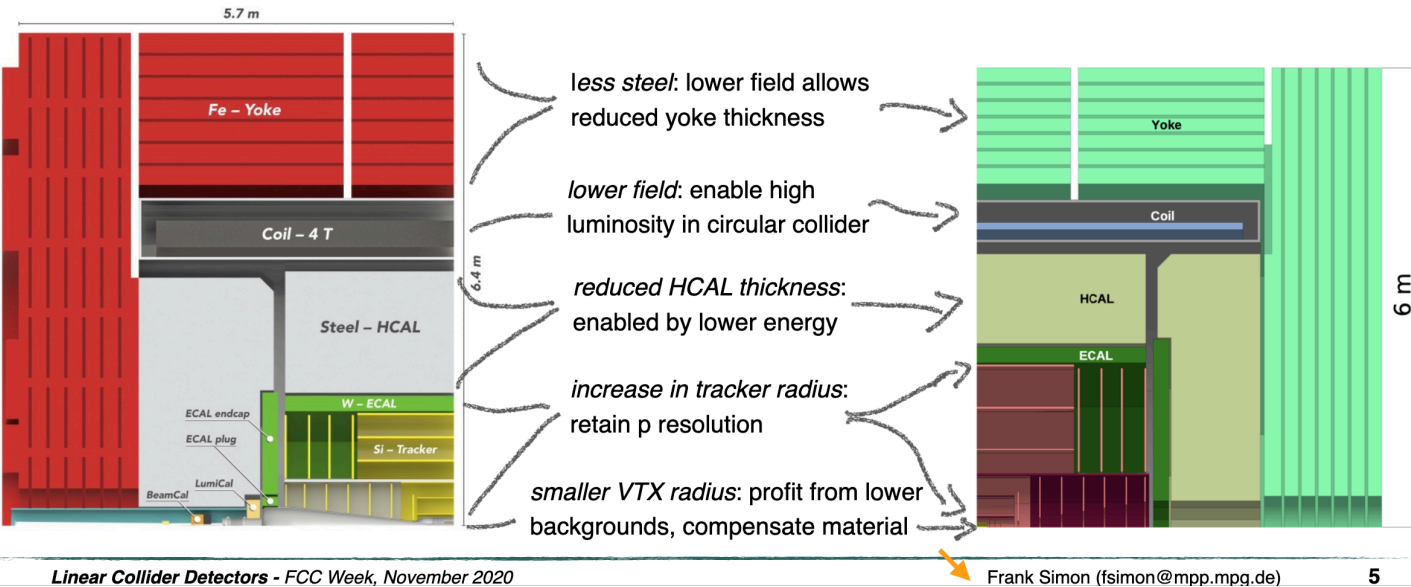


## From LCs to FCCee

From CLICdet to CLD

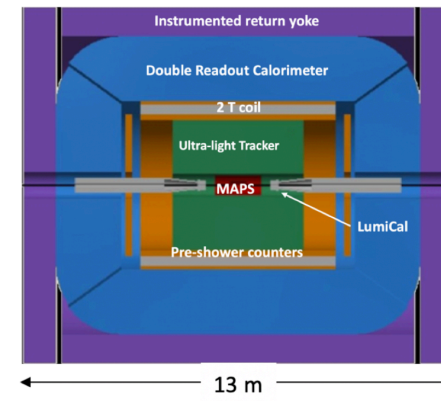


- A LC-inspired FCCee detector concept - retaining key performance parameters
- Evolving from CLIC to CLD

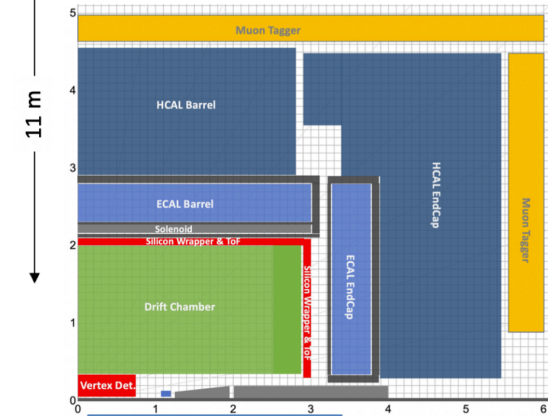


- CLD detector: based on CLICdet
  - adjusted for FCCee at lower energies and lower B-field:
  - larger tracker, thinner calorimeters,.....

## IDEA



## Noble Liquid ECAL based

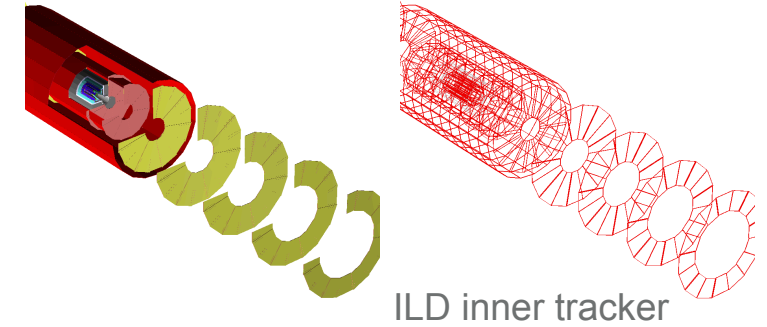
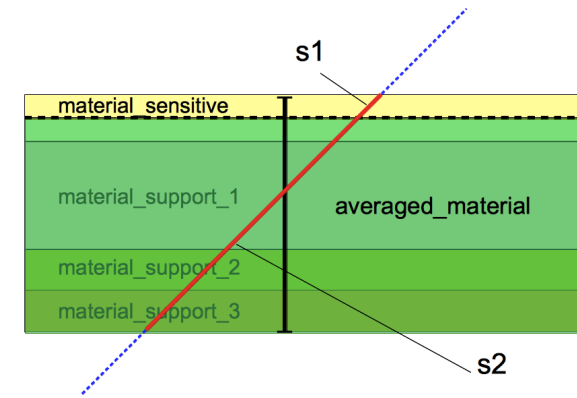


- ongoing work to implement the other two FCCee detector models in DD4hep
  - dual readout calorimeter
  - LAr/Noble Liquid calorimeter

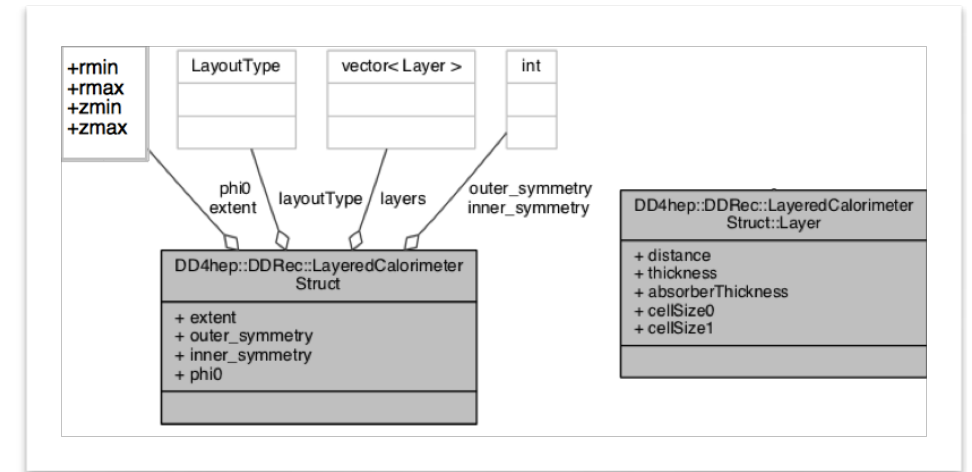
interoperability in **Key4hep** can re-use for **ILD**:

- sub detector components
  - see: talk D. Jeans
- (high level) reconstruction algorithms
  - e.g. Si-Tracking or ACTS

- 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)



can exchange (high level) reconstruction algorithms with other detectors if they also use DDRec



# large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- 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)

achieve excellent tracking efficiencies and resolution w/ realistic tracking codes

CLICdp  
Tracking efficiency vs  $p_T$  [GeV].  
Conditions:  $\sqrt{s} = 3$  TeV,  $10^\circ < \theta < 170^\circ$ , vertex R < 50 mm,  $\Delta_{IC} > 0.02$  rad.  
Series: No background (blue), 3 TeV  $\gamma\gamma \rightarrow$  hadrons background (red).

ILD  
 $\epsilon_{trk}$  vs  $p_T / \text{GeV}$ .  
Conditions:  $\sqrt{s} @ 500$  GeV -  $p_e > 100$  MeV,  $\cos(\theta) < 0.99$ .  
Series: IDR-L (blue), IDR-S (red).

CLICdp  
 $\sigma(Dp_T/p_{T,true})$  [GeV<sup>-1</sup>] vs  $\theta$  [°].  
Single  $\mu^+$ .  
Series:  $p_T = 1$  GeV (black),  $p_T = 10$  GeV (red),  $p_T = 100$  GeV (blue).

Momentum Resolution  
 $\sigma_{trk}$  [GeV<sup>-1</sup>] vs Momentum (GeV).  
Series:  $\sqrt{s} = 3$  TeV,  $10^\circ < \theta < 170^\circ$ , vertex R < 50 mm,  $\Delta_{IC} > 0.02$  rad (blue);  $\sqrt{s} = 3$  TeV,  $10^\circ < \theta < 170^\circ$ , vertex R < 50 mm,  $\Delta_{IC} > 0.02$  rad, hadrons background (red);  $\sqrt{s} = 3$  TeV,  $10^\circ < \theta < 170^\circ$ , vertex R < 50 mm,  $\Delta_{IC} > 0.02$  rad, hadrons background, no background (blue);  $\sqrt{s} = 3$  TeV,  $10^\circ < \theta < 170^\circ$ , vertex R < 50 mm,  $\Delta_{IC} > 0.02$  rad, hadrons background, no background, no material (red).

DESY. Frank Gaede, LCWS 2021, 17.03.21

# large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- 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 and 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)

### Pandora Algorithms

slide: J.Marshall

CLICdP  
50 GeV Jets  
100 GeV Jets  
250 GeV Jets  
750 GeV Jets  
1500 GeV Jets

ILD  
 $|\cos\theta_{hadron}| < 0.7$   
IDR-L  
IDR-S

ArborPFA

AprilPFA

70 GeV pion  
CALICE SDHCAL

DESY. Frank Gaede, LCWS 2021, 17.03.21



# large reconstruction code base in iLCSoft

Developed over >15 years for linear collider detectors - e.g. ILD

- 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 and 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

$m_{Pt} = \sqrt{m_{vtx}^2 + |p_t|^2 + |p_t|}$

- **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

$\delta_{LHHH}$  improves by 40% w/ perfect jet clustering

DESY. Frank Gaede, LCWS 2021, 17.03.21

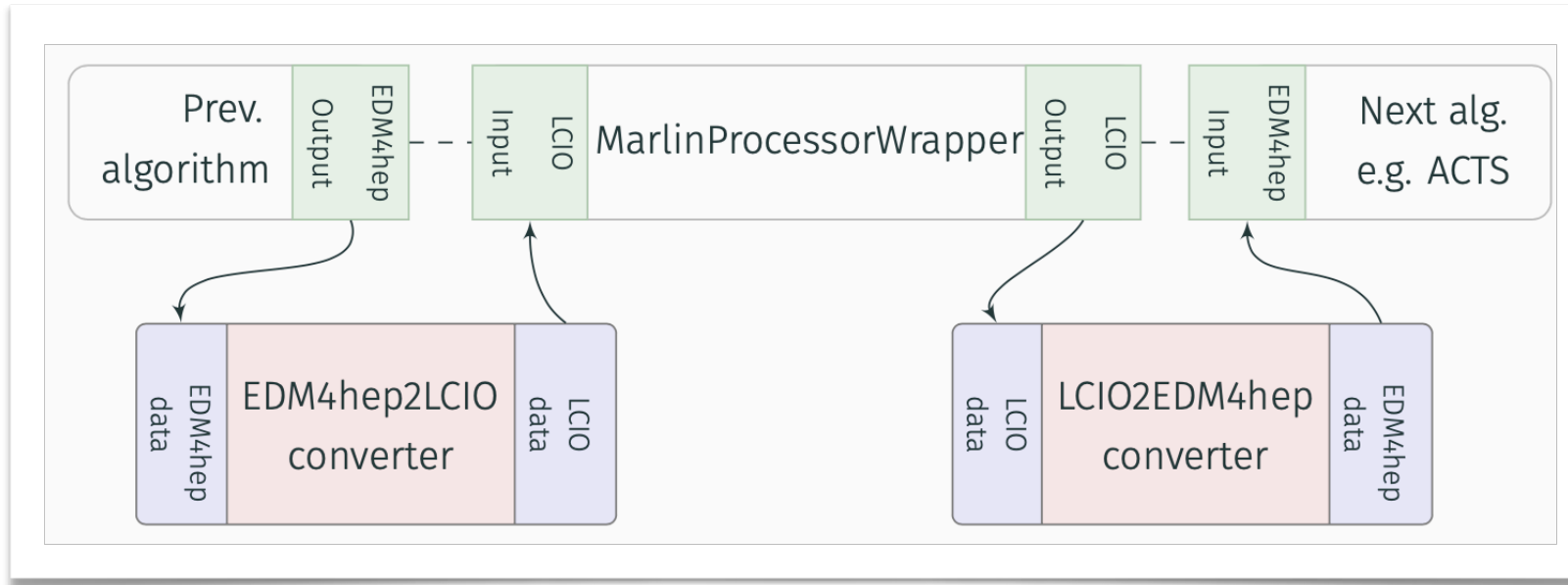
DESY. Frank Gaede, ILD Meeting 2024, CERN, 16.01.24

8



# K4MarlinWrapppper

the vision: mix and match Marlin and Gaudi algorithms



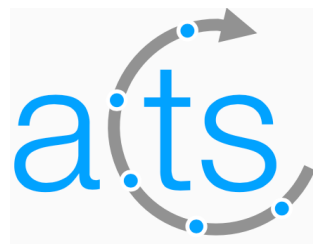
- in a transition phase algorithms developed in the new EDM4hep/Gaudi world can gradually replace older algorithms

- could start to think about actually developing new algorithms from the beginning in **Gaudi/EDM4hep** !
  - volunteers or candidates ?

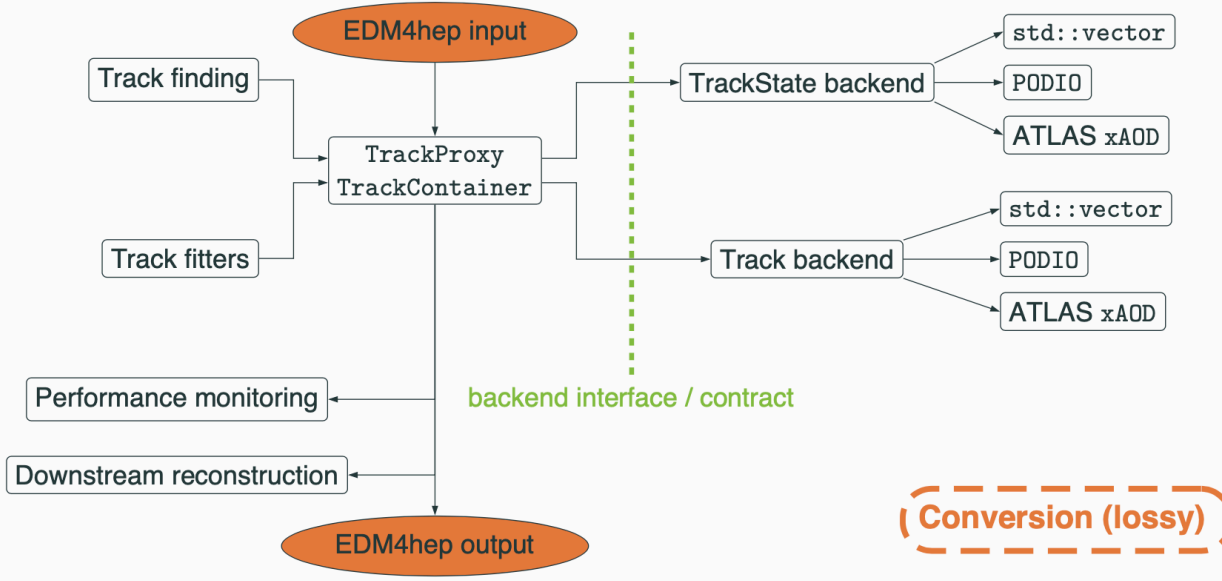


# ACTS

a common tracking toolkit



## Architecture

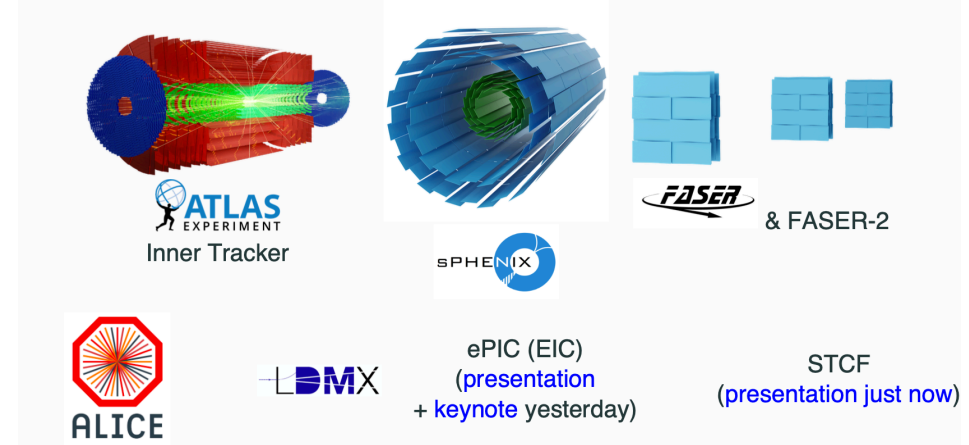


## What is ACTS?

- 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

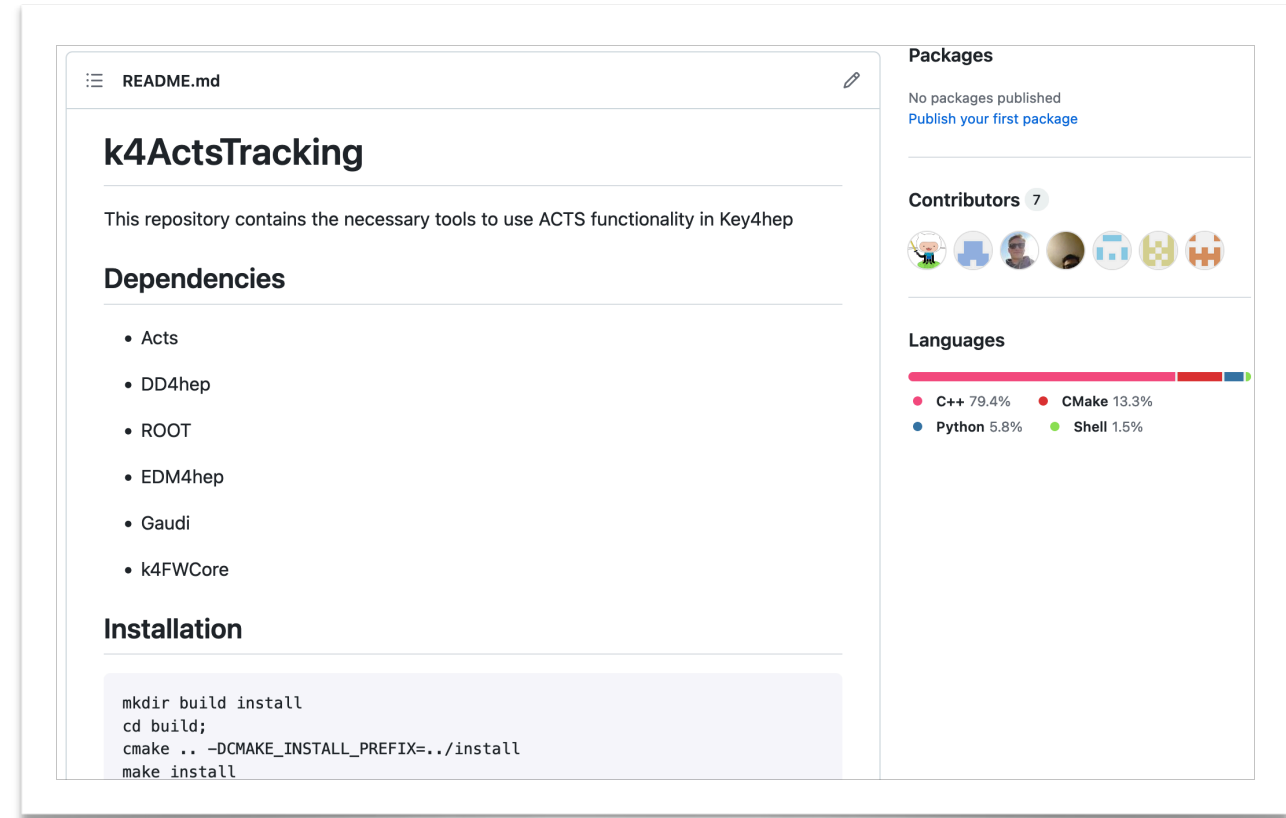


- ACTS tracking toolkit is the the current choice for track fitting (and finding !?) in Key4hep
- recently implemented interface to write out EDM4hep Tracks
  - some discussion needed w/ tracking and ACTS experts on details of the tracking data model
  - perigee vs. on-surface parameterisation ...

# k4ACTS

## integration of ACTS in Key4hep

- first major reconstruction algorithm in Key4hep/ Gaudi
- ongoing work at CERN (L.Reichenbach) in context of electron reconstruction w/ ACTS for CLD
- one crucial issue for ILD 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**

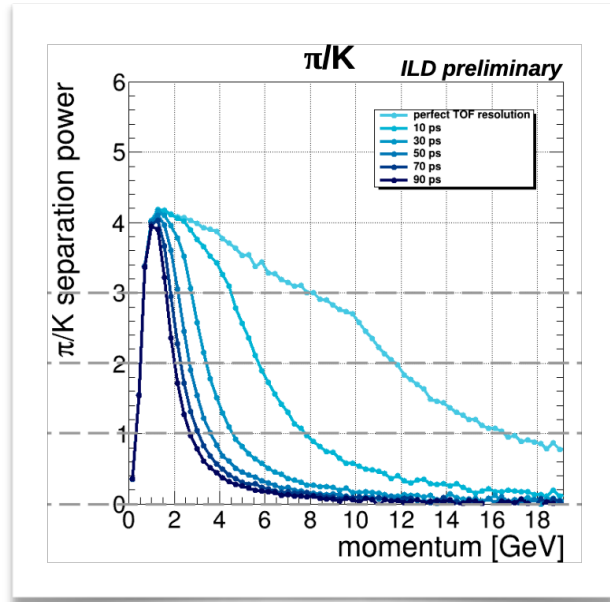
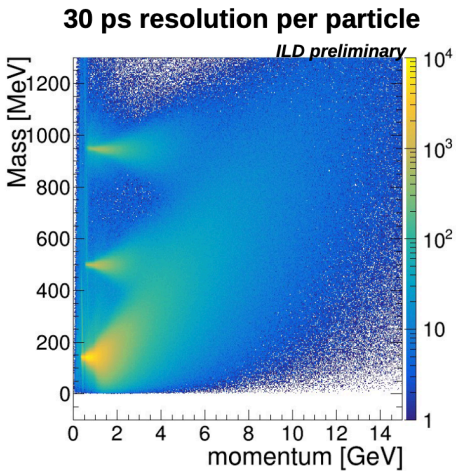
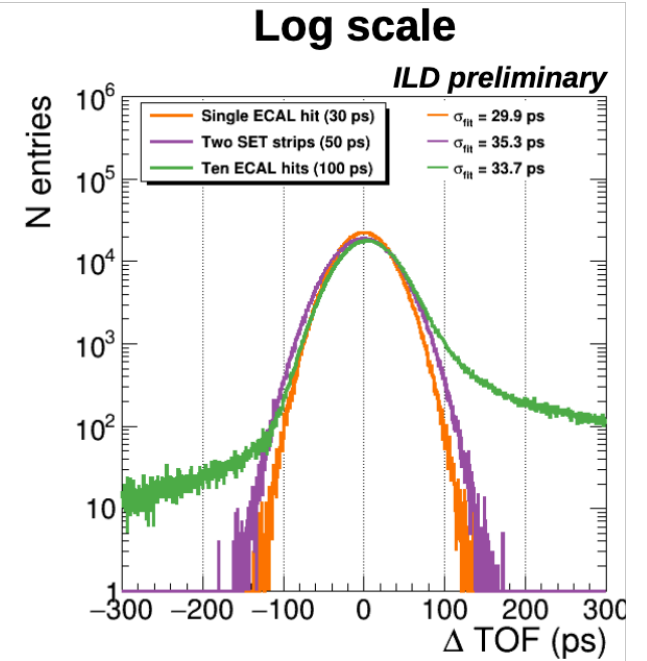
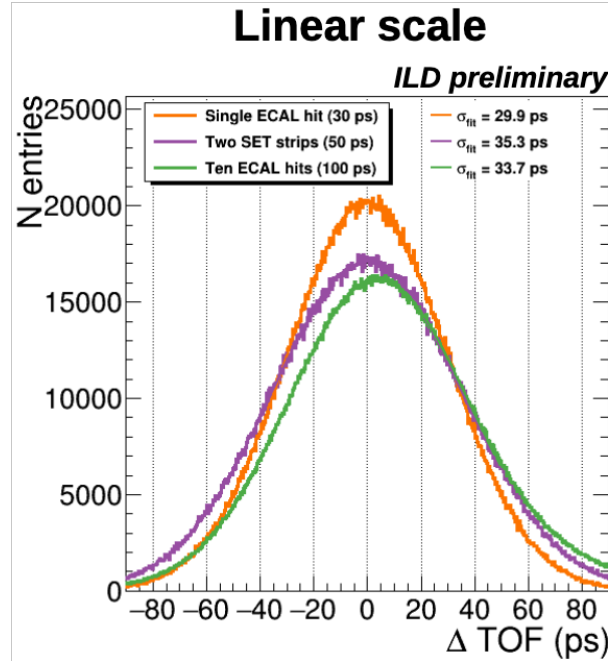
Language	Percentage
C++	79.4%
CMake	13.3%
Python	5.8%
Shell	1.5%

- might eventually benefit from this implementation
- expect significant effort to adapt to ILD tracking (w. TPC)
- probably more a midterm project ...

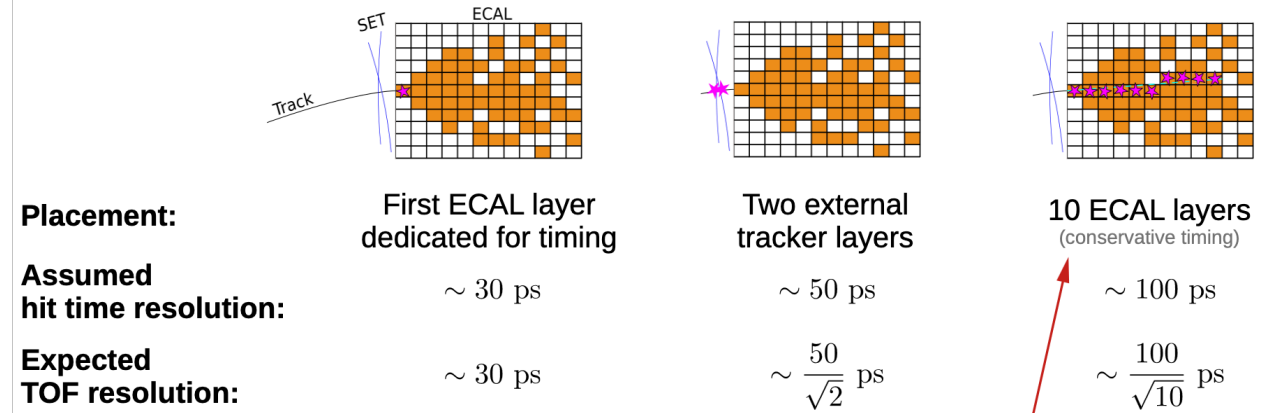
# TOF Estimators

## enhance particle identification

- studied various options and potential improvements for TOF in ILD
- track length calculation is important
- need to decide on ILD goal - and strategy for TOF !**



### Time-of-flight reconstruction in ILD

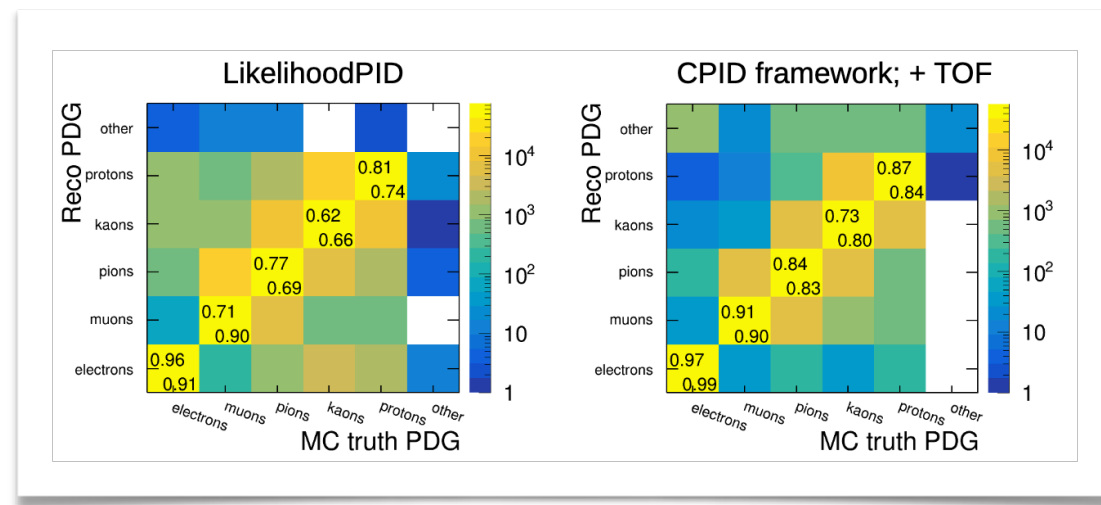
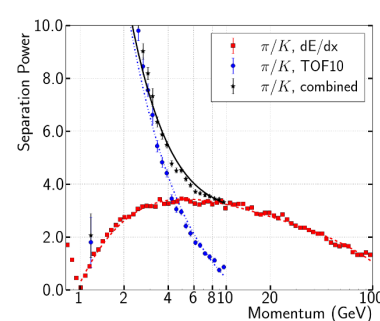
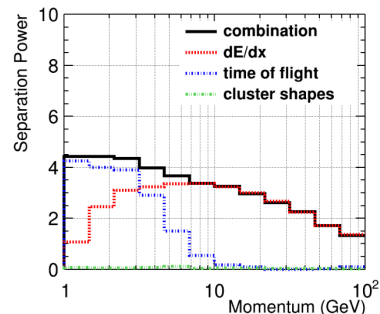
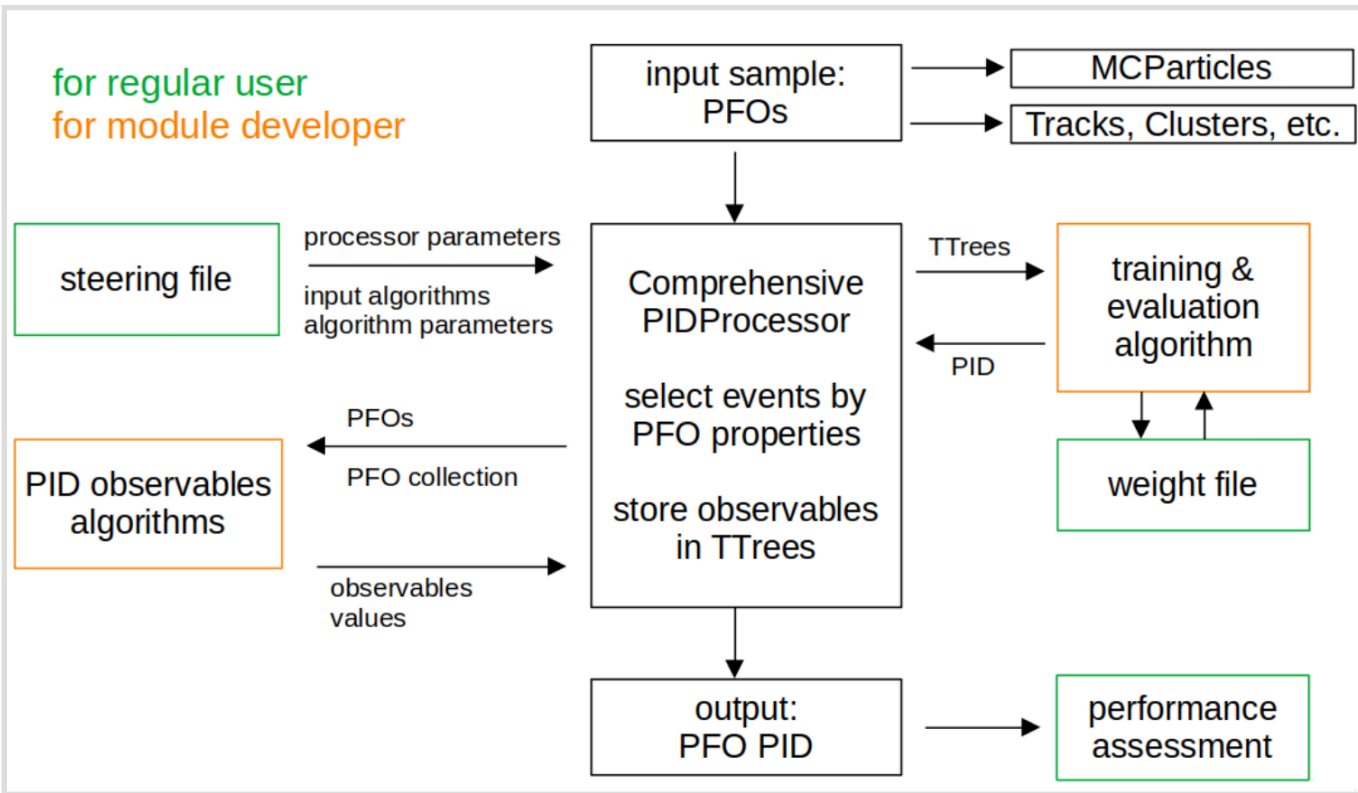


# General PID

## comprehensive particle identification CPID



- developed a new PID toolkit - release in MarlinReco
- can use to train/evaluate/test novel deep learning algorithm for PID
- allows much easier combination of different measurements (dE/dx, TOF, cluster-shape) and shows better results

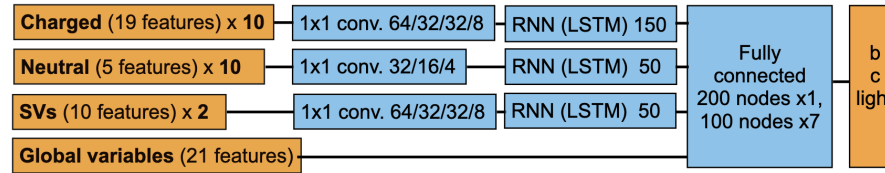




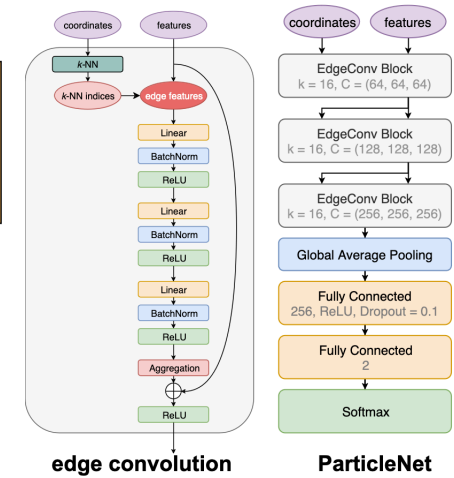
# Flavour tagging

with deep learning methods

- implemented DeepJet and ParticleNet flavour tagging for ILD
  - achieve mostly better results than LCFIPlus
- implemented in Marlin processor



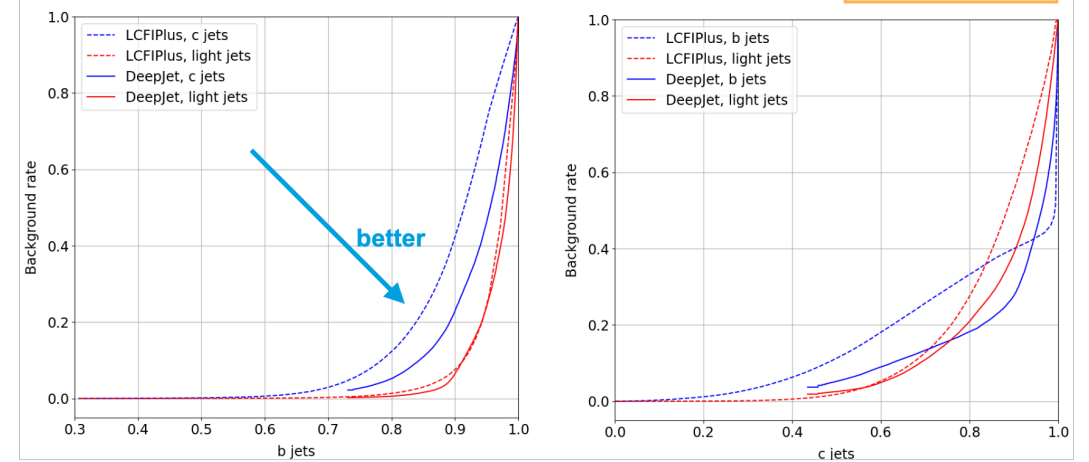
M.Meyer



## Workflow training (PyTorch) & inference (iLCSoft/Marlin)

- run PV & SV finder, jet clustering and vertex refinement of LCFIPlus
- run Marlin processor that calculates and stores features needed for the flavor taggers } iLCSoft/Marlin
- store variables in **root files** with four trees (charged, neutral, jets, sv) } iLCSoft/Marlin
- Training (python scripts & PyTorch):**
  - convert trees in root files to **pandas dataframes**, do some **checks** and **cleaning**, store dataframes in **hdf5-files**
  - do **further pre-processing** and training in **PyTorch**
  - use **torch library** to convert trained model into model that can be used in C++
- Inference (iLCSoft/Marlin)**
  - store variables via **PIDHandler** (not optimal in terms of memory, might be changed)
  - run **Marlin processor** for tagging with **ParticleNet Model**
    - read feature values from PIDHandler
    - store them in the vectors needed by the ParticleNet Model (coordinates of const., features of the const., coordinates of SV, features of SV)
    - convert vectors to torch tensors and do the pre-processing
    - do the inference with the converted model
    - store output again using PIDHandler
- run Marlin processor to store outputs in trees and histogram that can be used to calculate ROCs etc.

## DeepJet: ROC curves - comparison to LCFIPlus







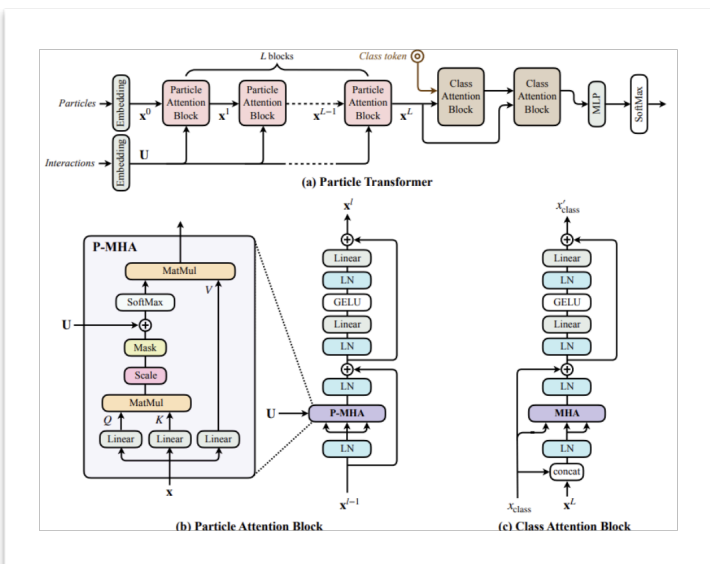
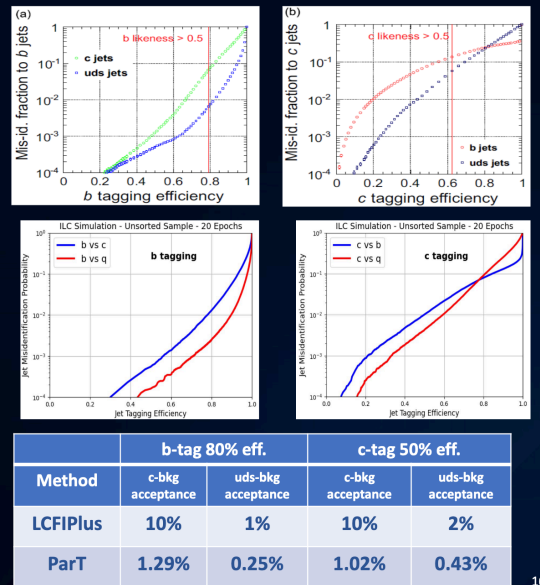
# Flavour tagging

## with deep learning methods

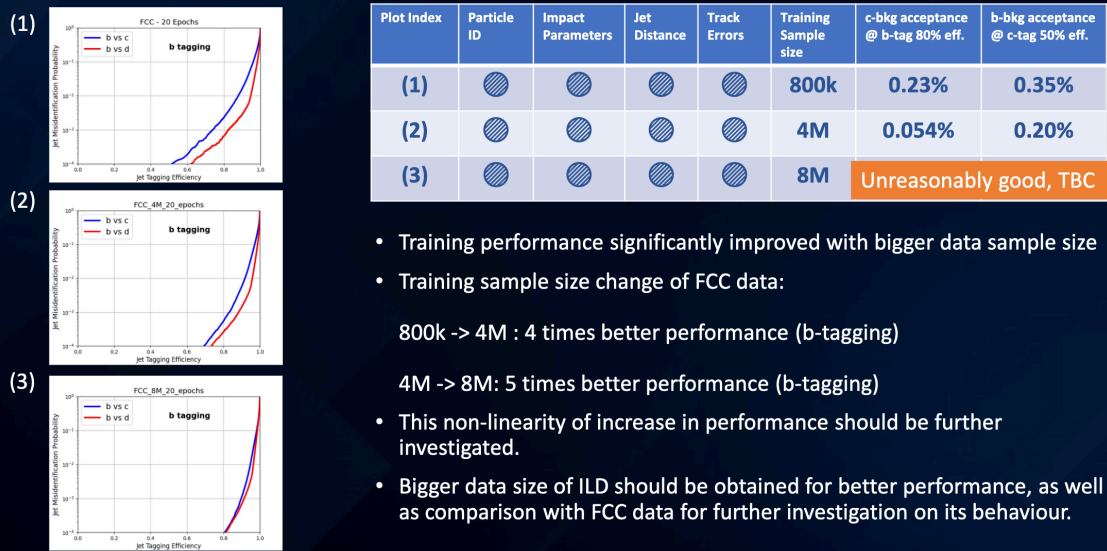
- implemented ParticleTransformer flavour tagging for ILD
  - achieve dramatically better results than LCFIPlus
- observe strange improvement w/ more training data at FCC -> to be studied
- framework inference not yet implemented
  - could do this in **Gaudi/EDM4hep**

### Application of ParT to ILD data (ILD qq 91 GeV, 0.8M jets for training)

- Jet tagging performance is greatly improved by ParT immediately.
- The performance is improved by 4.05 – 9.80 times compared to LCFIPlus with the same set of data.
- 20 epochs are taken, 200 epochs do not help improving performance but give overtraining



### Sample size affects performance (FCCee sample)



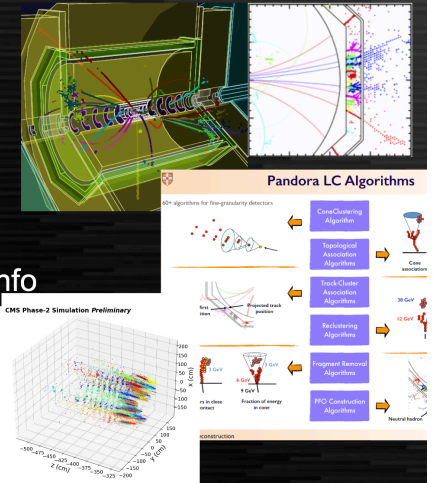
# Deep Learning for PFA



- started to develop deep neural networks for particle flow - partly based on CMS HGCal
- using GravNet and Object Condensation
- some early, promising results
- work in progress ...

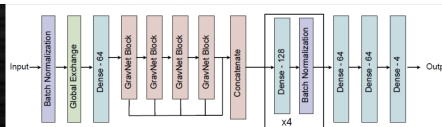
## Particle flow with DNN: introduction

- Separation of cluster at calorimeter
  - Charged or neutral cluster
- Essential for jet energy resolution
- Current algorithm: PandoraPFA
  - Combination of various process
  - Not easy to optimize or adding more info
- CMS HGCal clustering
  - Similar to ILD calo
  - Good for starting point



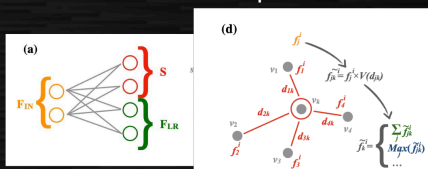
## PFA: clustering algorithm

- Input: position/energy/timing of each hit
- Output: virtual coordinate and  $\beta$  for each hit



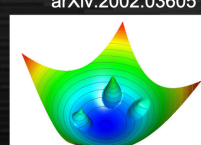
### GravNet [arXiv:1902.07987](https://arxiv.org/abs/1902.07987)

- The virtual coordinate (S) is derived from input variables with simple MLP
- Convolution using "distance" at S (bigger convolution with nearer hits)
- Concatenate the output with MLP



### Object Condensation (loss function) [arXiv:2002.03605](https://arxiv.org/abs/2002.03605)

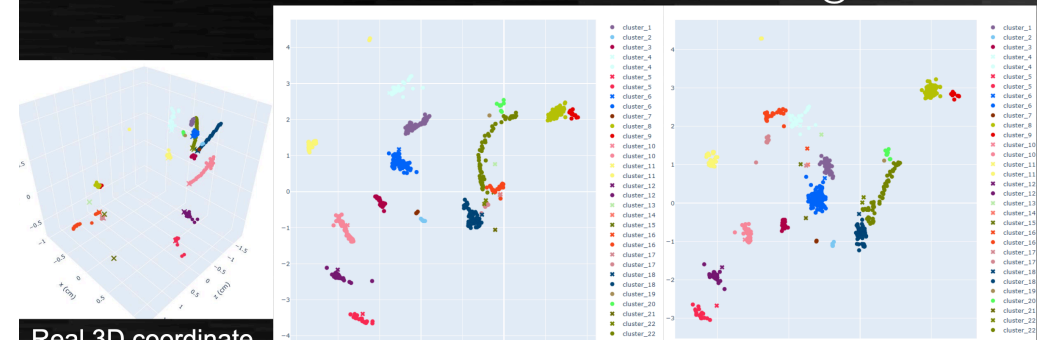
$$L = L_p + s_C(L_B + L_V)$$



- Condensation point: The hit with largest  $\beta$  at each (MC) cluster
- $L_V$ : Attractive potential to the condensation point of the same cluster and repulsive potential to the condensation point of different clusters
- $L_B$ : Pulling up  $\beta$  of the condensation point
- $L_p$ : Regression to output features

## Preliminary results – event sample

10 Taus @ 10 GeV each



Real 3D coordinate

Hits on the virtual coordinate – colored by MC truth clusters  
x refers virtual hits from tracks  
left with beta-track term, right without beta-track term

# AI in Key4hep

integrate ML inference smoothly in code base

- we see more and more developments using AI/ML for (HL) reconstruction, e.g.
  - *LCFIPlus* (TMVA)
  - *CPID* (TMVA et al)
  - *MarlinML* flavour tagging
  - DNN for PFA
  - ...
- should try and make an effort to unify and simplify the use of ML inference in Key4hep for reconstruction
  - some ideas developed in DDML for fast ML simulation

## Integration into the Full Simulation Chain

- Prototype library for running ML-based fast sim models: **DDFastShowerML**  
<https://gitlab.desy.de/ilcsoft/ddfastshowerml>
  - Use fast sim hooks in DDG4/Geant4
  - Use realistic, detailed detector models
  - Currently only supports CPU
  - Development ongoing
- Aim to have an easy to use library which can be adapted for all types of ML architectures in DD4hep
- **Essential** step to be able to study performance of model with **full physics benchmarks**

Necessary update to Geant4 version 11.1!

- **Trigger**
  - Fast Sim trigger
    - e.g. particle type, energy, geometry
- **Model**
  - Model-specific implementation of ML architecture
    - e.g. BIB-AE, Flow, Diffusion mode
- **Inference**
  - Concrete inference in C++
    - ONNX, LibTorch etc...
- **Geometry**
  - Concrete placement in detector geometry
    - Endcap\_barrel etc

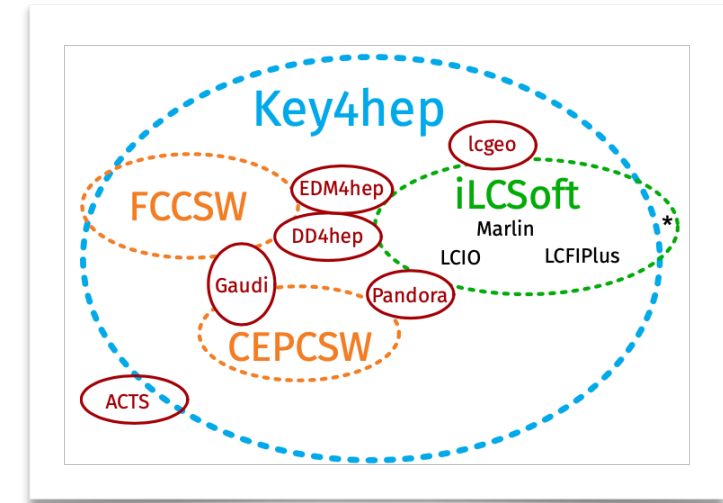
20 GeV photon in ILD generated with a BIB-AE

P.McKeown: integration of generative ML based fast simulation in dddmim for ILD



# Summary and Outlook

- **Key4hep** started as a new future collider community wide effort in 2020 to put together a modern turnkey software stack
  - contributors: CEPC, CLIC, FCC, EIC, ILC, LUXE, Muon Collider ...
- **iLCSoft and ILD software integral part of Key4hep from the start**
- **battle proven ILD standard reconstruction** can be run in Key4hep w/ **MarlinWrapper** as before - or w/ EDM4hep output
- many new developments in HLR tools - often ML/AI based
  - need to validate and integrate in ILD standard reconstruction

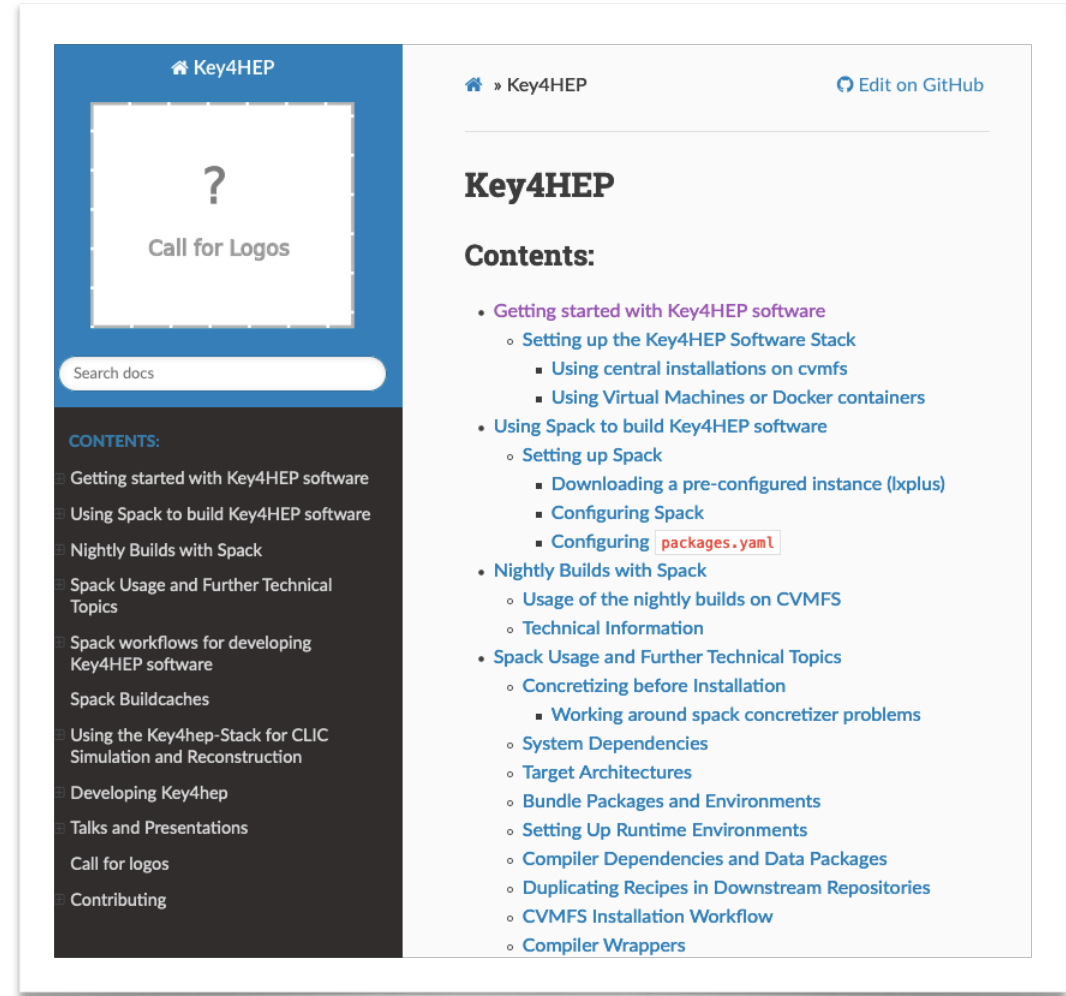


- Key4hep offers great opportunity to **modernise ILD software stack AND collaborate w/ other Higgs factories** - when studying ILD for FCC
- should make an attempt for next larger production/study in ILD to move to use more of the new tools in **Key4hep: Gaudi, EDM4hep**
  - have brief discussion on Wednesday
- limiting factor for all software developments: **manpower** ...

# 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>



The screenshot shows the Key4HEP documentation website. The top navigation bar includes a home icon, the text 'Key4HEP', and a link to 'Edit on GitHub'. The main content area features a large blue box with a white question mark and the text 'Call for Logos'. Below this is a search bar labeled 'Search docs'. A dark sidebar on the left contains a 'CONTENTS:' section with a list of topics. The main content area on the right has a 'Contents:' section with a detailed list of topics and sub-topics.

**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 (lplus)
      - 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