ANALYSIS@FCC

FCCAnalyses framework

Juraj Smieško

CERN

PyHEP.dev 2023

Princeton, 27 July 2023

INTRODUCTION

CERN

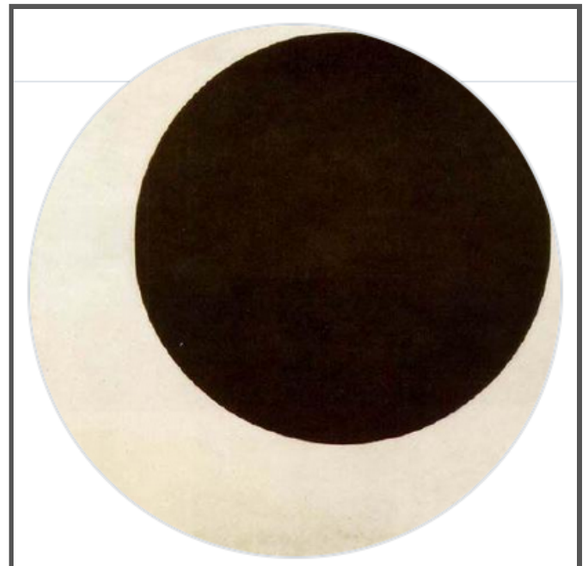
- FCCSW
- FCCAnalyses
- Phoenix/Web tools

CHARLES UNI.

- LAr Calorimeter for FCC-ee
- TileCal operations
- TileCal offline DQ tools
- Pileup mitigation

COMENIUS UNI.

- Photon + c-jet / Intrinsic Charm
- TileCal offline DQ tools



Juraj Smiesko
kjvbrt

Pronunciation: `You-rye`

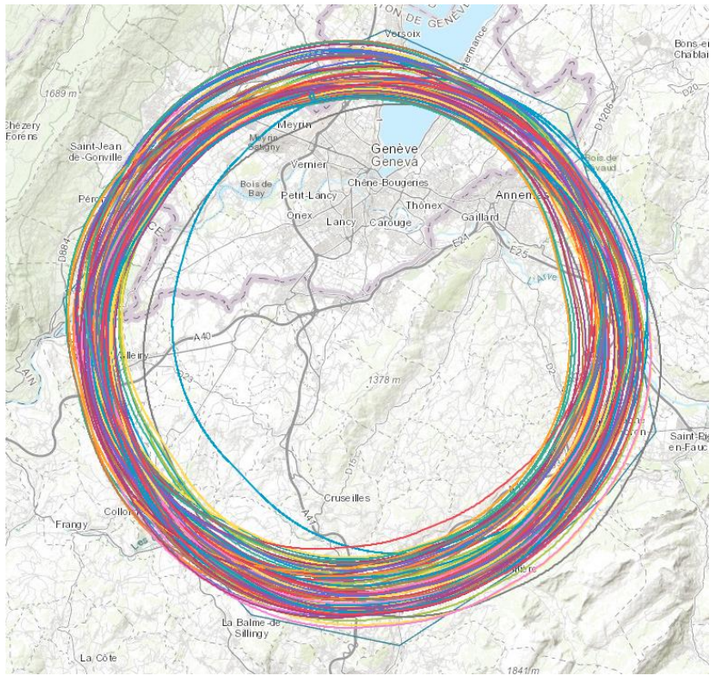
Institute: CERN

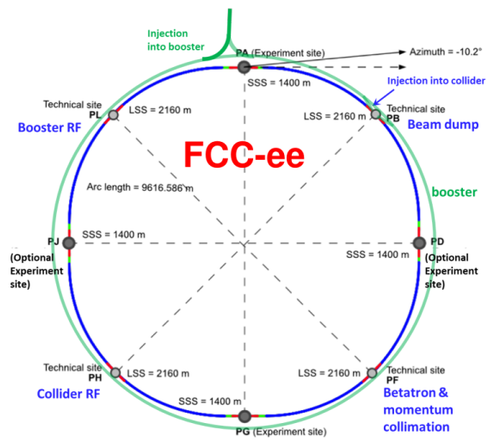
Alma Mater: Comenius University, SK

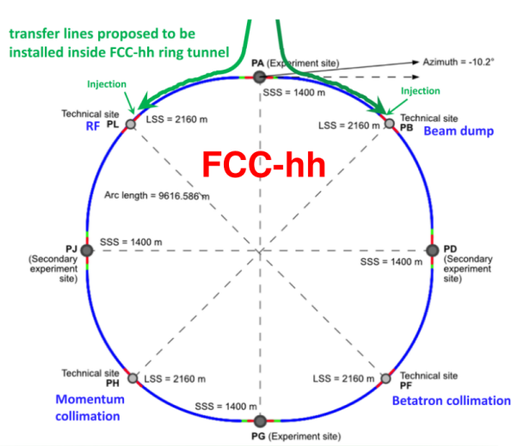
FUTURE CIRCULAR COLLIDER

Energy and luminosity upgrade in integrated program

- FCC-ee (Z, W, H, tt):
  Highest luminosities at Z, W, ZH among proposed Higgs and EW factories with indirect discovery potential up to ~ 70 TeV
- FCC-hh (~100 TeV):
  Direct exploration of next energy frontier (~ x10 LHC) and unparalleled measurements
- Feasibility Status Report in 2025





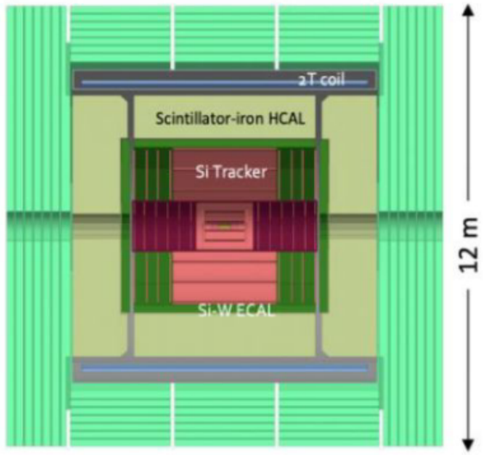


2045 - 2063 | 2070 - 2095

See latest FCC Week in London

FCC DETECTORS

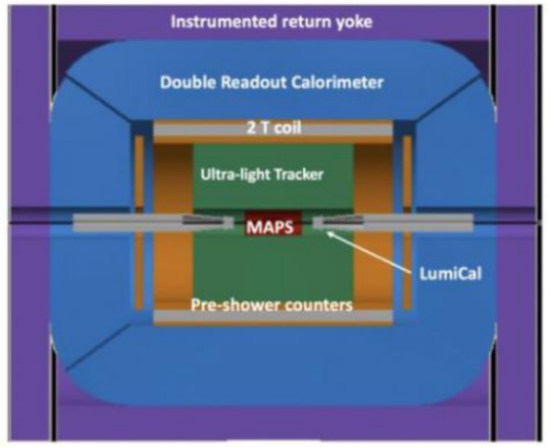
CLD



- Well established design
  - ILC -> CLIC detector -> CLD
- Full Si vtx + tracker;
- CALICE-like calorimetry;
- Large coil, muon system
- Engineering still needed for operation with continuous beam (no power pulsing)
  - Cooling of Si-sensors & calorimeters
- Possible detector optimizations
  - σ_p/p, σ_E/E
  - PID ($\mathcal{O}$(10 ps) timing and/or RICH)?

CDR

IDEA



- A bit less established design
  - But still ~15y history
- Si vtx detector; ultra light drift chamber w powerful PID; compact, light coil;
- Monolithic dual readout calorimeter;
  - Possibly augmented by crystal ECAL
- Muon system
- Very active community
  - Prototype designs, test beam campaigns, …

Noble Liquid ECAL based

new

- A design in its infancy
- Si vtx det., ultra light drift chamber (or Si)
- High granularity Noble Liquid ECAL as core
  - Pb/W+LAr (or denser W+LKr)
- CALICE-like or TileCal-like HCAL;
- Coil inside same cryostat as LAr, outside ECAL
- Muon system.
- Very active Noble Liquid R&D team
  - Readout electrodes, feed-throughs, electronics, light cryostat, …
  - Software & performance studies

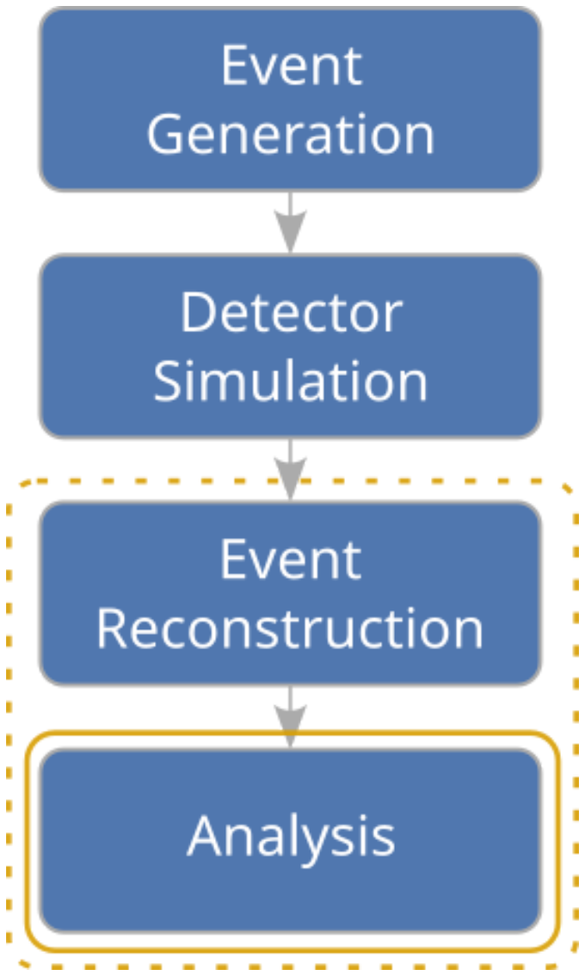
See: M. Salvaggi's presentation

FCCANALYSES SCOPE

Goal of the framework is to aid the users in obtaining the desired results not only from the reconstructed physics objects

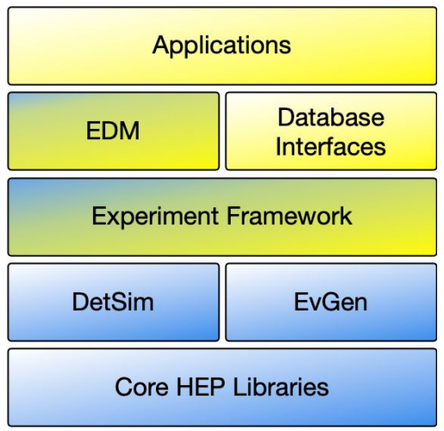
Requirements:

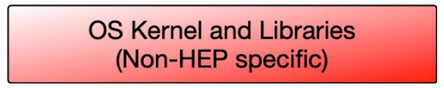
- Efficiency — Make quick turn-around possible
- Flexibility — Allow heavy customization
- Ease of use — Should not be hard to start using
- Scalable — Seamlessly handle from small to large datasets

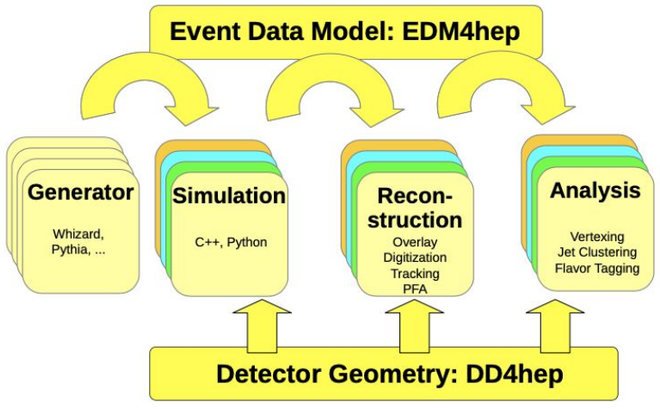


KEY4HEP

- Set of common software packages, tools, and standards for different Detector concepts
- Common for FCC, CLIC/ILC, CEPC, EIC, …
- Individual participants can mix and match their stack
- Main ingredients:
  - Data processing framework: Gaudi
  - Event data model: EDM4hep
  - Detector description: DD4hep
  - Software distribution: Spack



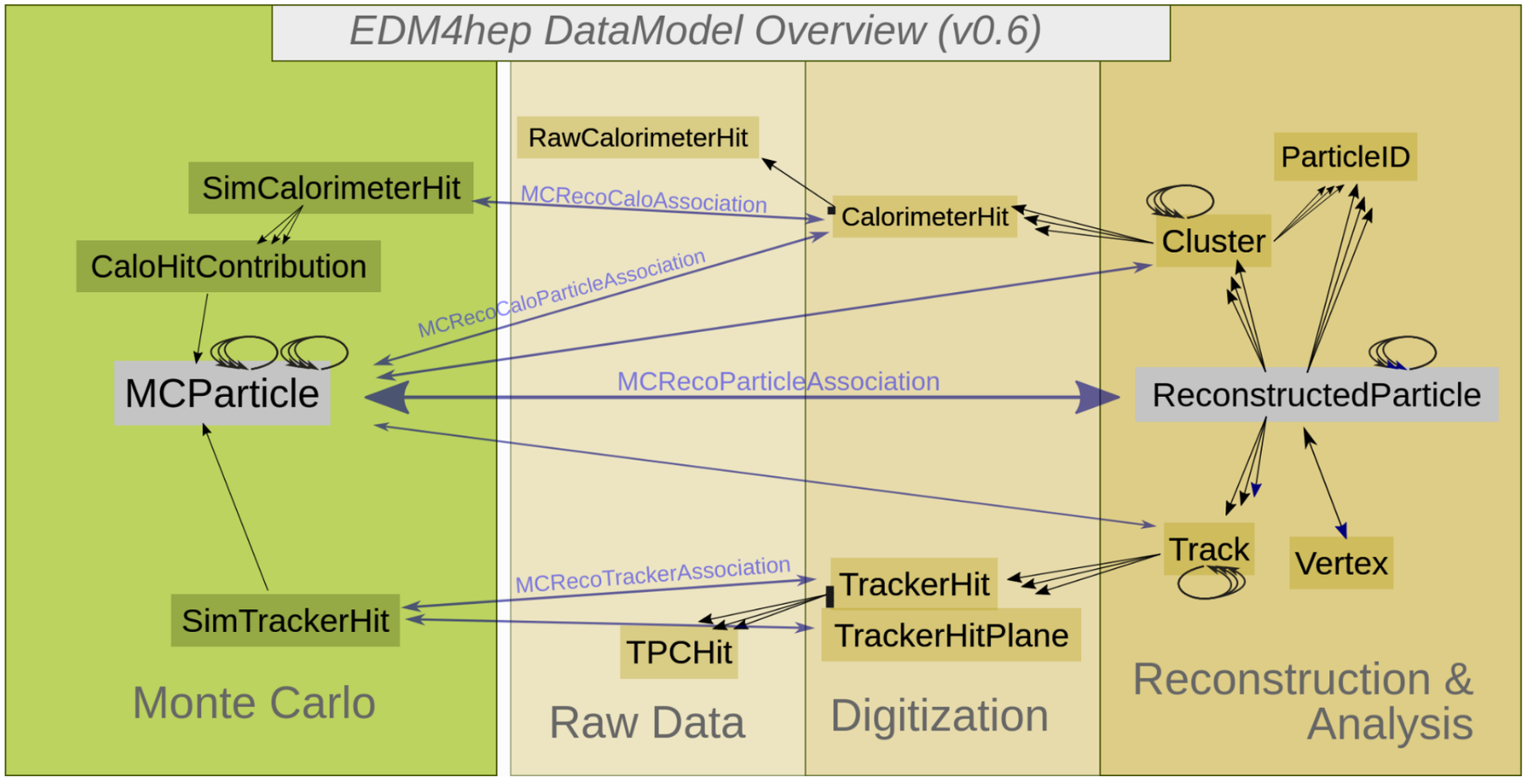




EDM4HEP I.

Describes event data with the set of standard objects.

- Specification in a single YAML file
- Strives to be minimal
- Generated with the help of Podio



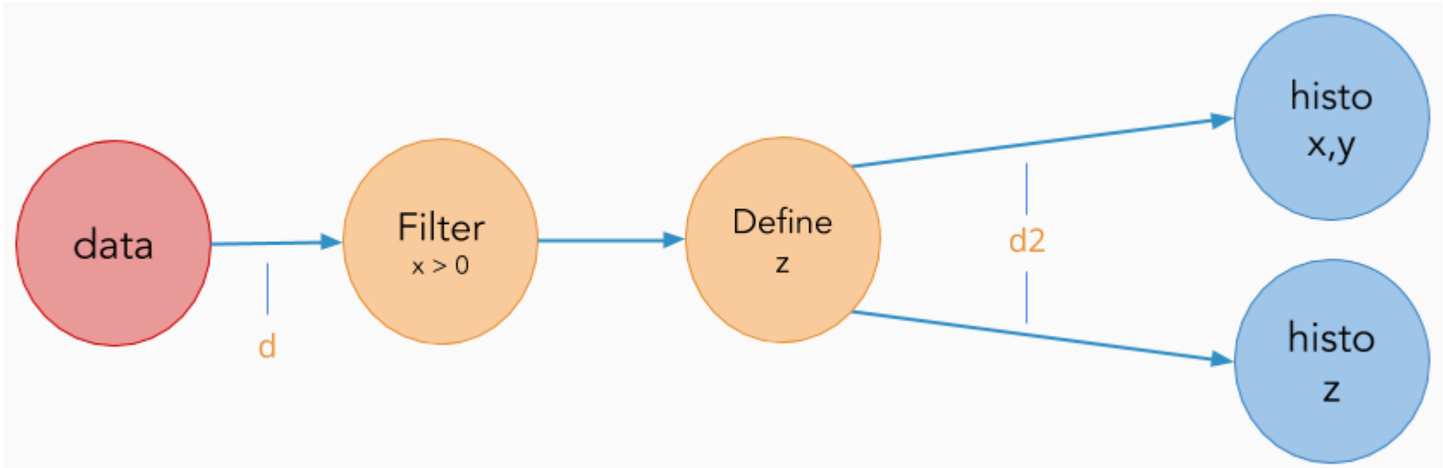
EDM4HEP II.

Example object:

```
#-------------  CalorimeterHit
edm4hep::CalorimeterHit:
  Description: "Calorimeter hit"
  Author : "F.Gaede, DESY"
  Members:
    - uint64_t cellID             //detector specific (geometrical) cell id.
    - float energy                //energy of the hit in [GeV].
    - float energyError           //error of the hit energy in [GeV].
    - float time                  //time of the hit in [ns].
    - edm4hep::Vector3f position //position of the hit in world coordinates in [mm].
    - int32_t type                //type of hit. Mapping of integer types to names via coll
```

- Current version: `v0.8.0`
- Objects can be extended / new created
- Bi-weekly discussion: Indico

ROOT RDATAFRAME



- Describes processing of data as actions on table columns
  - Defines of new columns
  - Filter rules
  - Result definitions (histogram, graph)
- The actions are lazily evaluated
- Multi threading is available out of the box
- Optimized for bulk processing

INTEGRATION WITH EXISTING TOOLS

- Boundary between reconstruction and analysis blurred
  - Especially for full-sim
  - Plan: Develop algorithm on analysis side, then move to reconstruction
- Many C++ tools/libraries created over the years
  - Most are integrated into the Key4hep stack
  - At the moment we have:
    - ROOT — together with RDataFrame
    - ACTS — track reconstruction tools
    - ONNX — neural network exchange format
    - FastJet — jet finding package
    - DD4hep — detector description
    - Delphes — fast simulations

DISTRIBUTION

FCCAnalyses latest release `v0.7.0` can be found:

- As a package in the stable Key4hep stack
  - Allows to quickly put together small analysis
  - Limited options for customization

Latest/development version of the FCCAnalyses can be found:

- As a package in the nightlies Key4hep stack
  - Might easily break
- By checking out `master` branch
  - Allows greater customization
  - Requires discipline

Platforms: CentOS 7, AlmaLinux 9, Ubuntu 22.04

ANALYSIS ARCHITECTURE

One can write and run an analysis in several ways:

- Managed mode: `fccanalysis run my_ana.py`
  - The RDataFrame frame is managed by the framework
  - Analysis script has to contain compulsory attributes
  - Libraries are loaded automatically
  - Dataset metadata are loaded from remote location — CVMFS/HTTP server
  - Batch submission on HTCondor
  - Customization: Possible at the level of analyzer functions
  - Intend for: Quick analysis, no advanced analyzer functions
- Standalone mode: `python my_ana.py`
  - The RDataFrame frame is managed by the user
  - Can leverage the FCCAnalyses library of analyzer functions
- Ntupleizer style

WRITING AN ANALYZER FUNCTION

- Typically an analyzer is a `struct` which operates on an EDM4hep object
- ROOT RDataFrame needs to be aware of the analyzer function
  - Provided as a string
  - A file loaded and JITed by the `ROOT.gInterpreter`
  - Compiled in the library

```
128    /// Get the invariant mass in a given hemisphere (defined by it's angle wrt to axis).
129    struct getAxisMass {
130    public:
131      getAxisMass(bool arg_pos=0);
132      float operator() (const ROOT::VecOps::RVec<float> & angle,
133                        const ROOT::VecOps::RVec<float> & energy,
134                        const ROOT::VecOps::RVec<float> & px,
135                        const ROOT::VecOps::RVec<float> & py,
136                        const ROOT::VecOps::RVec<float> & pz);
137    private:
138      bool _pos; /// Which hemisphere to select, false/0=cosTheta<0 true/1=cosTheta>0. Default=0
139    };
```

DOCUMENTATION

Several documentation types

- FCC Tutorials: https://hep-fcc.github.io/fcc-tutorials/
  - Focused on providing a tutorial on a specific topic
- Code reference: https://hep-fcc.github.io/FCCAnalyses/doc/latest/index.html
  - Provides details about implementation of individual analyzers
- Manual pages:
  - Info about commands directly in the terminal: `man fccanalysis`
- FCCAnalyses website, FCCSW website

QUESTIONS

- Interaction of the C++ analyzer functions with the Python in the context of RDataFrame
- Efficient work with the Podio and EDM4hep data format
- Ways to distribute of analyzer functions among the users
- Large scale management of the pre-generated input samples
- Integration of facilities needed to support full simulation studies
- Non CLI based modes of interaction with the analysis code