

ANALYSIS AND VISUALIZATION

Juraj Smieško

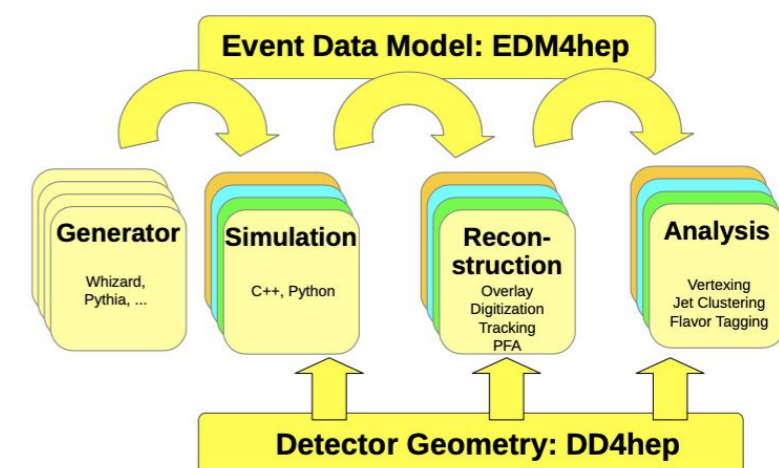
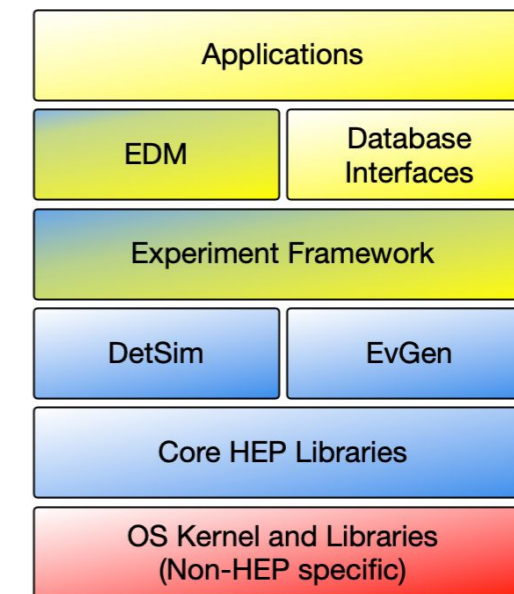
CERN

6th FCC Physics Workshop in Kraków

24 January 2023

KEY4HEP

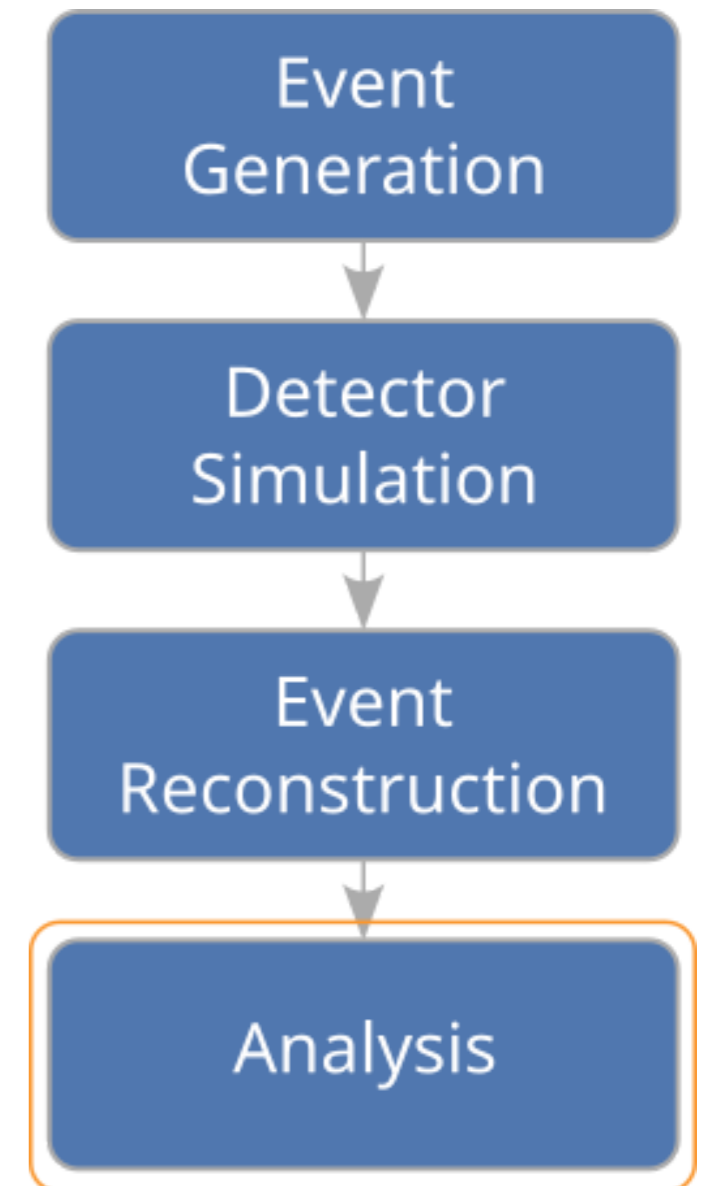
- Set of common software packages, tools, and standards for different Detector concepts
- Common for FCC, CLIC/ILC, CEPC, EIC, ...
- Individual participants can mix and match their stack
- Main ingredients:
 - Data processing framework: [Gaudi](#)
 - Event data model: [EDM4hep](#)
 - Detector description: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



ANALYSIS

ANALYSIS SCOPE

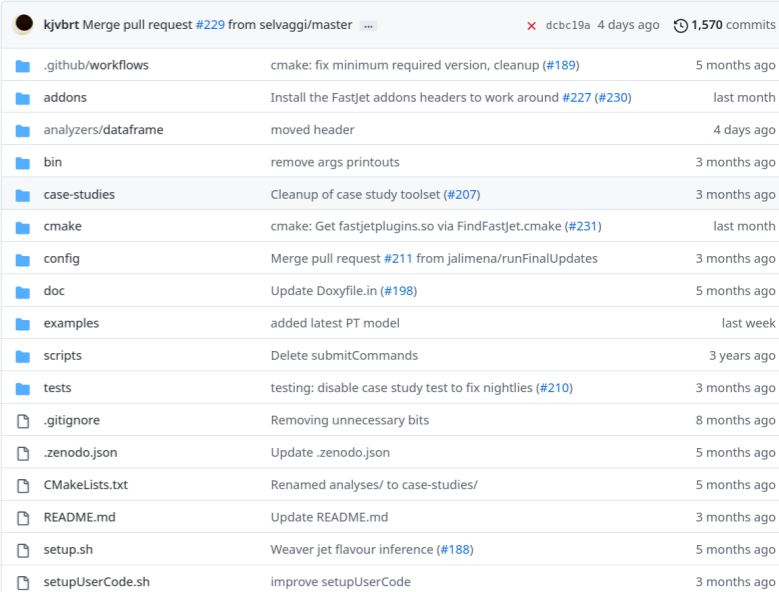
- Takes reconstructed objects and produces physics results
- The objects are described in EDM4hep format
- Input datasets are usually centrally produced
- Access to the detector description
- Needs to accommodate different analysis strategies
- Runs locally and on Batch/Grid



ECOSYSTEM I.

The physics analyses at FCC are spread through two repositories and a storage space:

- **FCCAnalyses**
 - Repository of common tools and algorithms
 - General analysis code in analyzers
 - Steering of the analysis (RDataFrame)
 - Access to the (meta)data
 - Running over large datasets / on batch
 - (Proto)package machinery for case studies
- **FCCeePhysicsPerformance**
 - Main place for the abstracts
 - Contains very specific analysis code
 - Or prototypes of tools of common interest to be eventually moved to FCCAnalysis
 - (Proto)package repository
- Storage space on EOS `/eos/experiment/fcc`



File	Commit Message	Time
.github/workflows	cmake: fix minimum required version, cleanup (#189)	5 months ago
addons	Install the Fastjet addons headers to work around #227 (#230)	last month
analyzers/dataframe	moved header	4 days ago
bin	remove args printouts	3 months ago
case-studies	Cleanup of case study toolset (#207)	3 months ago
cmake	cmake: Get fastjetplugins.so via FindFastjet.cmake (#231)	last month
config	Merge pull request #211 from jalimena/runFinalUpdates	3 months ago
doc	Update Doxyfile.in (#198)	5 months ago
examples	added latest PT model	last week
scripts	Delete submitCommands	3 years ago
tests	testing: disable case study test to fix nightlies (#210)	3 months ago
.gitignore	Removing unnecessary bits	8 months ago
.zenodo.json	Update .zenodo.json	5 months ago
CMakeLists.txt	Renamed analyses/ to case-studies/	5 months ago
README.md	Update README.md	3 months ago
setup.sh	Weaver jet flavour inference (#188)	5 months ago
setupUserCode.sh	improve setupUserCode	3 months ago

Case studies (evolving list)

1. Electroweak physics at the Z peak
2. Tau Physics
3. Flavour physics
4. WW threshold
5. QCD measurements
6. Higgs physics
7. Top physics
8. Direct searches for new physics

ECOSYSTEM II.

Supporting repositories:

- [WebTools](#)
 - Code for the website hosting dataset info
- [FCC-config](#)
 - Stores configurations for the datasets
- [EventProducer](#)
 - Responsible for production of input datasets
- [fcc-tutorials](#)

NO	NAME	NEVENTS	NWEIGHTS	NFILES	NBAD	NEOS	SIZE (GB)
1	wzp6_ee_nunuH_Hbb_ecm240	1,200,000	0	12	0	12	10.72
2	wzp6_ee_nunuH_Hcc_ecm240	1,100,000	0	11	0	11	8.89
3	wzp6_ee_nunuH_Htautau_ecm240	1,200,000	0	12	0	12	1.91
4	wzp6_ee_nunuH_Hss_ecm240	1,008,052	0	11	0	11	7.49
5	p8_ee_WW_ecm240	58,228,784	0	584	0	584	558.10

FCC Tutorials

FUTURE CIRCULAR COLLIDER

Search docs

CONTENTS:

- 1. First Steps
- 2. Generators, Fast Simulation and Analysis
 - 2.1. FCC: Getting started with event generation
 - 2.2. FCC: Getting started with simulating events in Delphes
 - 2.3. FCC: Getting started with analysing simulated events
 - 2.4. FCC: tracking and vertexing example using specific flavour decays
 - 2.4.1. Installation of FCCAnalyses
 - 2.4.2. Building a custom sub-package in FCCAnalyses
 - 2.4.3. Reconstruction of the primary vertex and of primary tracks
 - 2.4.4. Reconstruction of displaced vertices in an exclusive decay chain: starting example
 - 2.4.5. Exercise: analysis of $\tau \rightarrow 3\mu$
- 3. Full Detector Simulations

» 2. Generators, Fast Simulation and Analysis »

2.4. FCC: tracking and vertexing example using specific flavour decays [Edit on GitHub](#)

2.4. FCC: tracking and vertexing example using specific flavour decays

Learning Objectives

This tutorial will teach you how to:

- fit some tracks to a common vertex in **FCCAnalyses**, reconstruct the primary vertex and the primary tracks
- retrieve the tracks corresponding to a specific flavour decay in **FCCAnalyses**
- produce **flat ntuples** with observables of interest with **FCCAnalyses**
- build your own algorithm inside **FCCAnalyses**

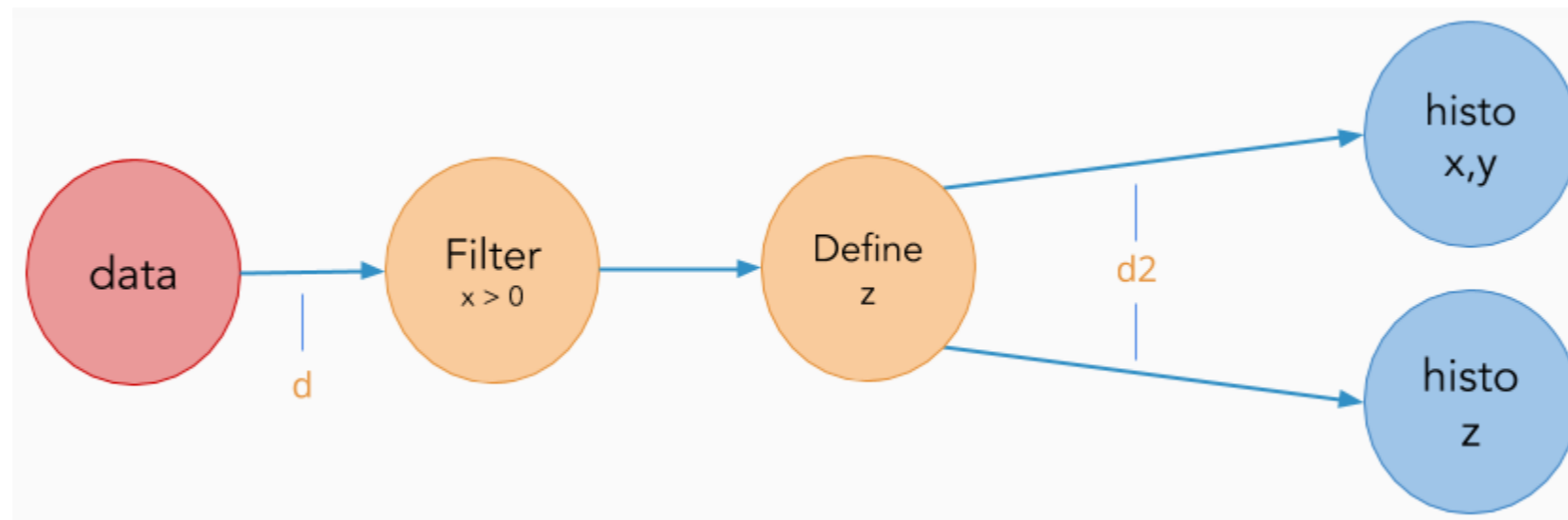
For the vertex fitter, we make use of the code developed by Franco Bedeschi, [see this talk](#). The [subsequent updates presented in July 2022](#) offer possibilities for complex reconstructions, but they are not yet ready to use in the public FCCAnalyses version (coming soon).

To reconstruct the primary vertex and the primary tracks, we follow the LCFI+ algorithm (T. Suehara, T. Tanabe), described in [arXiv:1506.08371](#).

2.4.1. Installation of FCCAnalyses

For this tutorial we will need to develop some code inside FCCAnalyses, thus we need to install it locally. If not already done, you need to clone and install it. Go inside the area that you have setup for the tutorials and get the FCCAnalyses code:

RDATAFRAME



- Describes processing of data as actions on table columns
- The actions are instantly or lazily evaluated
- Multi threading is available out of the box
- Optimized for bulk processing

ARCHITECTURE

- The analysis is build around [ROOT RDataFrame](#) with rich "standard library"
- Over the years this library (analyzers) have been written
- Analyzers are usually structs which operate on EDM4hep objects
- Optional dependencies for analyzers can be FastJet, DD4hep, ACTS and ONNX
- Dataset metadata are loaded from remote location --- AFS/HTTP server
 - Number of events generated, cross-section, ...
- Python used for steering, but not necessary
 - One can write analysis in pure C++

```
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     };
```


FCCANALYSES LIBRARY

- Vertexing
- ACTS vertex finder
- Event variables
- Calorimeter hit/cluster variables
- Reconstructed/MC particle operations
- Flavour tagging
- Jet clustering/constituents

Case studies (evolving list)

1. [Electroweak physics at the Z peak](#)
2. [Tau Physics](#)
3. [Flavour physics](#)
4. [WW threshold](#)
5. [QCD measurements](#)
6. [Higgs physics](#)
7. [Top physics](#)
8. [Direct searches for new physics](#)

WORKFLOW

- The analysis is divided into three stages:
 - `analysis_stage1.py`, ... — pre-selection stages, analysis dependent, usually runs on batch
 - `analysis_final.py` — final selection, produces final variables
 - `analysis_plots.py` — produces plots from histograms/TTrees
 - The stages files contain objects which are loaded into "main" function with the help of `getattr()`
 - The first stage reads the data in EDM4hep format
 - Running on batch is done by running on-the-fly generated shell script in subprocess

PROTO PACKAGES

`case-studies` machinery allows to create (semi)independent analysis

Example analysis is split into several locations:

- Analysis stages are in `examples` in FCCAnalyses
- Abstract and Results in `case-studies` in FCCeePhysicsPerformance
- Benchmarks are in `tests` in FCCAnalyses
- Analysis specific code in `case-studies` in FCCeePhysicsPerformance

AREAS TO IMPROVE I.

Needs to be addressed for the FSR:

- Debugging tools
- Individual event investigation + visualization
- Integration with the production system
- - Migration to Dirac
 - Automatic deployment of new samples
- Rigidity of the FCCAnalysis framework
 - Restrictive predefined stages
 - Weaver analysis example:
 - Analysis "properly" implements only first stage
 - Requires custom stage for training/testing
 - Place to store common objects/variables between stages
 - All python machinery crammed into one module

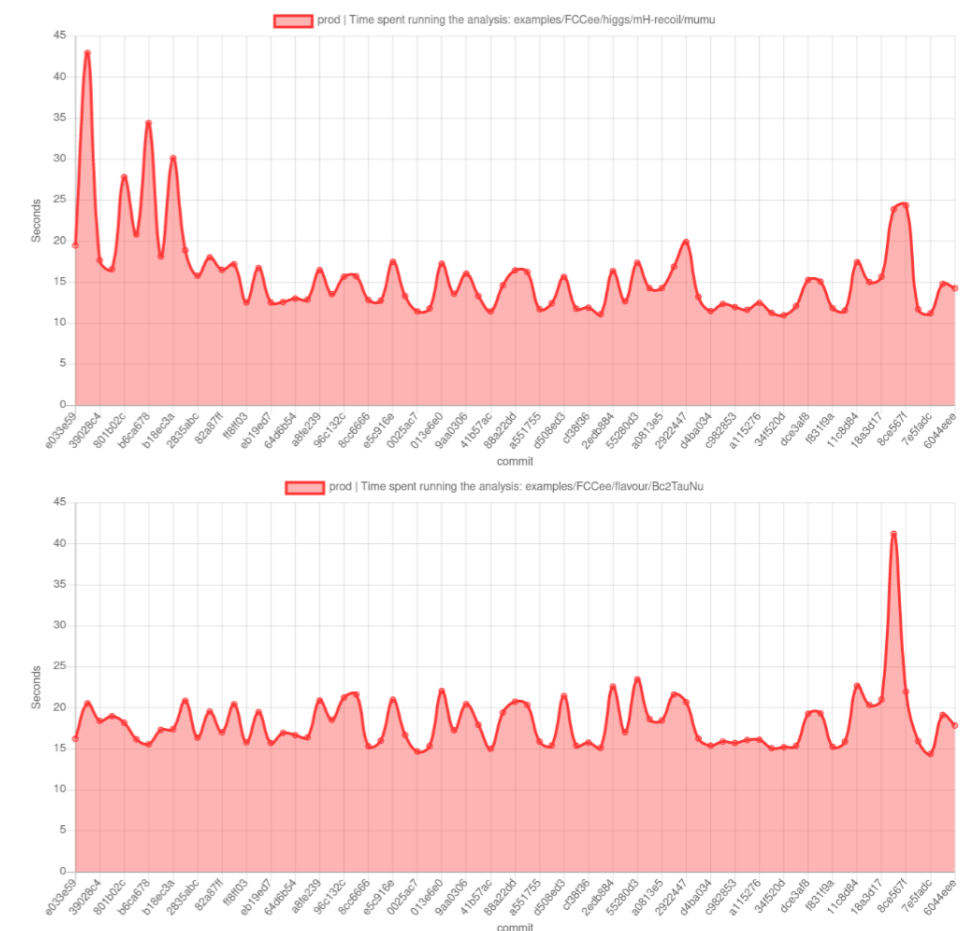


AREAS TO IMPROVE II.

Nice to have:

- Error signaling
- Testing and reorganization of the analyzers
- Long term reproducibility
- Benchmarking covering all analyzers and analyses

Benchmark



VISUALIZATION

MAIN USE CASES

Detector

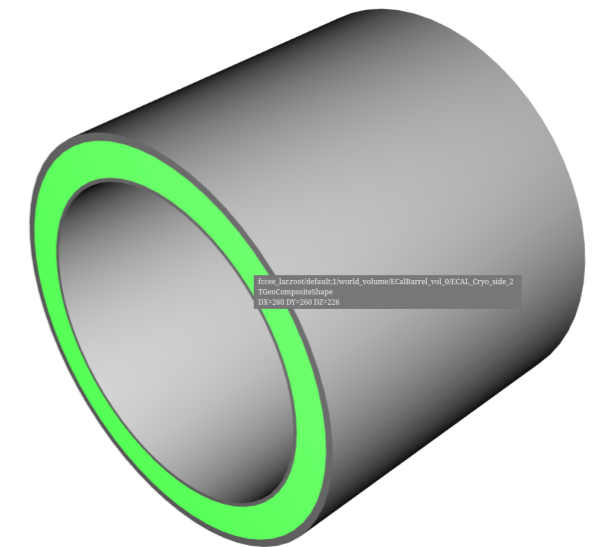
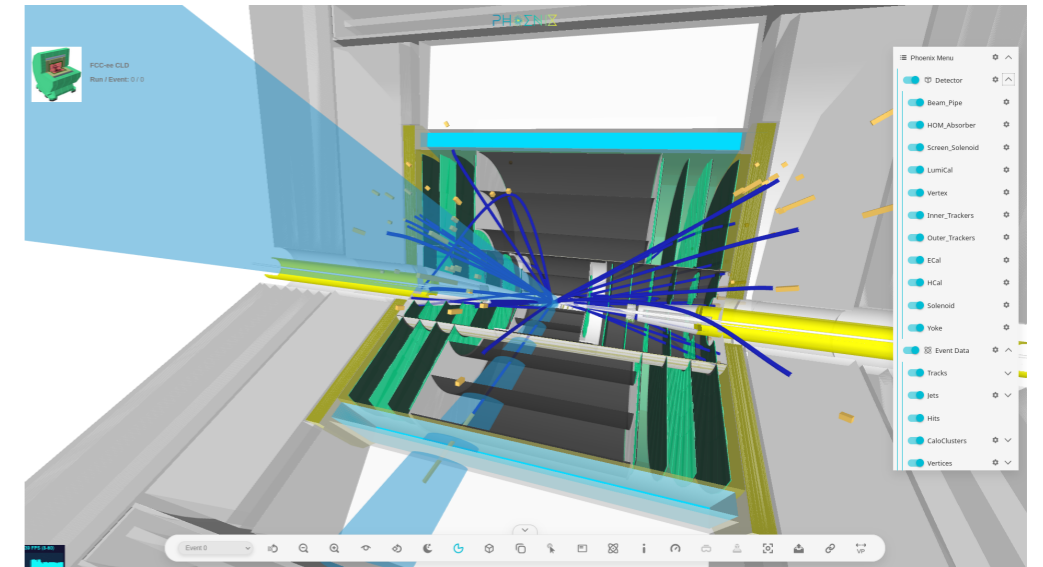
- Detector description in DD4hep format
 - Combination of C++ and XML
- Conversion available to:
 - ROOT, GDML, glTF
- Can be viewed in:
 - geoDisplay, Geant4, JSROOT, Phoenix

Events

- Events could come from different sources in EDM4hep format
 - Full simulation: Pythia
 - Fast simulation: Delphes
 - Simple particle gun
- Storage formats:
 - ROOT, JSON
- Viewers:
 - Phoenix, CED

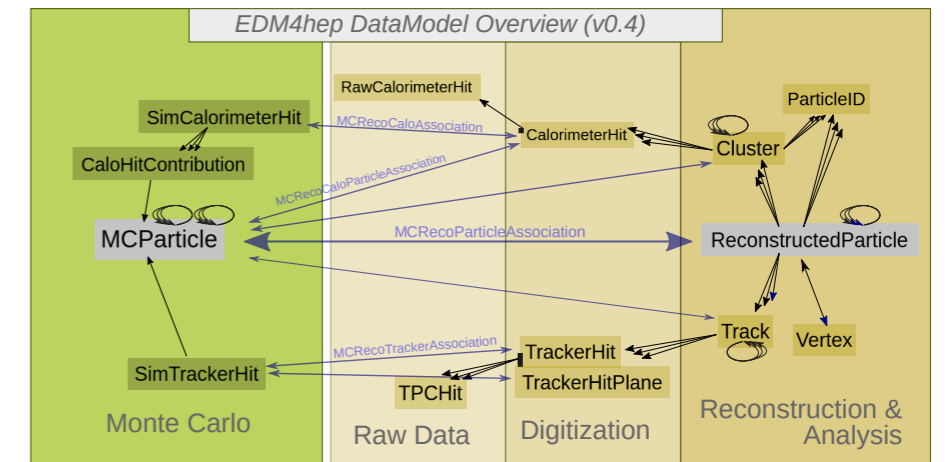
WEB BASED VISUALIZATION

- Visualize events and detectors
- Using web based tools
 - Independent of OS graphics
- **Phoenix**: Detector independent event display
 - Developed under HSF
 - Written in TypeScript
 - Static application
- **JSROOT**
 - Part of the ROOT project
 - Offers possibility to work with ROOT files on the web



PHOENIX WORKFLOW

- Separate event data and detector description
- Events
 - Described in [EDM4hep](#) event data format
 - Convert ROOT files into JSON files
 - EDM4hep data structure is kept
- Detector
 - Detector is described in [DD4hep](#) compact files
 - Convert XML into ROOT for JSROOT
 - Convert ROOT into glTF for Phoenix



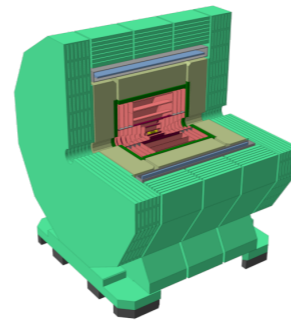
Visualizing FCC event data in the browser.



Playground

Get started with the different Phoenix features.

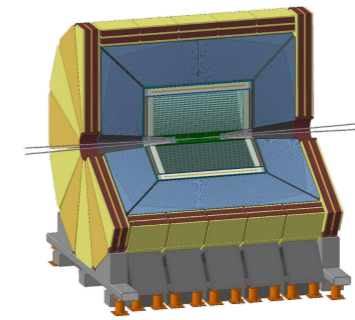
Show



FCC-ee CLD

Show the FCC-ee CLD detector.
One simple event.

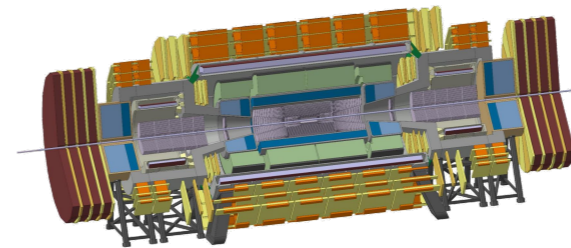
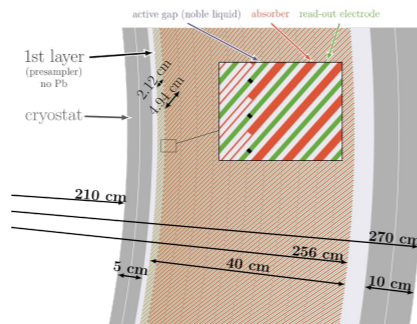
Show



FCC-ee IDEA

Show the FCC-ee IDEA detector.
One simple event.

Show



List of FCC detectors available.

CONCLUSIONS & OUTLOOK

- Choice of ROOT RDataFrame in combination with reading EDM4hep format suits most of the analyses
 - Continually growing list of analyses
- Large "standard library" is being build up
- Rigidity of the framework starts to limit more complex analyses
- Several ways of visualizing event data and detector geometry readily available
- Necessity of bridging the gap between the FCC stack and the graphic subsystem of the OS

More heads
are welcome!



