Status of FCC Software

1st Workshop FCC France

May 14, 2020

G Ganis, C Helsens
CERN-EP

FCC Software (FCCSW) today

- FCCSW is still largely what was used for the CDR
- Good modular structure based on Gaudi (LHCb, ATLAS)
  - Base for Key4hep
- Provide support for all the required functionality
  - Event Data Model (EDM), Generators, Detector geometry, Fast/Full simulation, Reconstruction, ...
- Current main limitations are in the implemented functionality
  - Available generators, in particular for FCC-ee
  - Palette of detector concepts with parametrized description
    - Quality of the description
  - Palette of detectors with detailed geometry description
    - Digitisation of their signal
  - Reconstruction algorithms

FCC Software tomorrow

Common software for future experiments

- Bologna workshop, June 2019
  - Present: LHC, ILC, CLIC, FCC, CEPC, SCTF, HSF
  - Agreed to investigate the possibility to have a common event data model (EDM4hep) and contribute to the development of a Common Turnkey Software Stack (Key4hep)
    - One framework (Gaudi best candidate), DD4hep, EDM4hep, Geant4, ROOT, …

- Follow-up in Hong Kong, 17 January 2020
  - Present: ILC, CLIC, FCC, CEPC
  - Agreed to set-up regular weekly meetings, a GitHub repository, documentation, deployment area on CVMFS, …
  - Get quickly first version of EDM4hep and Key4hep available

Today status: {EDM4hep v0.1, Key4hep v0.1} available since april 2020; ready to start testing

Mandate for the after-CDR FCC software

- Support for more detailed studies, in particular for e+e-, focusing on
  - Completeness
    - State-of-Art generators, MDI support, reconstruction / analysis algorithms, …
  - Flexible detector description
    - Easy switch/ replace sub-detectors, change dimensions / layout, …
  - Easy-of-use
    - Low usability thresholds and fast / easy learning curve
  - Adequate computing support and CPU / storage resources
  - Extensive documentation and regular training
- Ensure that SW is part-and-parcels of the Turnkey Software Stack
- Foster development and use in
  - Physics studies, Detector optimization, Machine-Detector Interface
- Foster / support substantial participation for FCC institutes worldwide

Experimental challenges for FCC-ee (on software)

- Ref: A Blondel @ FCC Physics on March 30th and case studies
- Requirements on detector understanding O(1-2) better than LEP
  - Need to simulate a lot of (reliable) data
- Priority is to have **as soon as possible**
  - Flexible **full** simulation and reconstruction for case and detector design studies
    - b-tagging, reconstruction and vertex geometry, tracking, PID etc.
  - Flexible **fast** simulation to support case studies
    - And also generator-level studies, or brain activity
- Independent of Snowmass
  - But Snowmass may be instrumental to foster activities and provide synergies
- Possible computing efficiency issue (in particular @ Z)
  - Interplay fast / full simulation may be required to mitigate

What can we do for FCC-ee with what we have today?

Monte Carlo Generators and FCCSW

- Generators repository: GenSer @ LCG software stacks
  - Generator Service hosted by EP-SFT @ CERN

    Collaboration with the authors and with the LHC experiments to prepare validated code for communities at the LHC
  - Actively used by ATLAS, LHCb, SWAN and some SME experiments
  - Deployed via CernVM-FS
- MC generators are typically standalone codes
  - Noticeable exception is Pythia8, which provides a callable interface
- FCCSW interoperates MC generators mostly through common data formats
  - HepMC, LHEF
- Pythia8 used to read HepMC, LHEF files

MC Generators: status and areas of work

- GenSer generators palette biased towards LHC
  - Good for FCC-hh, incomplete for FCC-ee
- General purpose generators such as Pythia8, Whizard, MadGraph5 available
  - But we need to get experience on how to use them effectively for FCC-ee
- Old LEP generators (KKMC, BHLUMI, MCSANC, BabaYaga, …)
  - Not available yet, but (often) still State-Of-Art
  - Wrappers to produce HepMC and/or LHEF output required

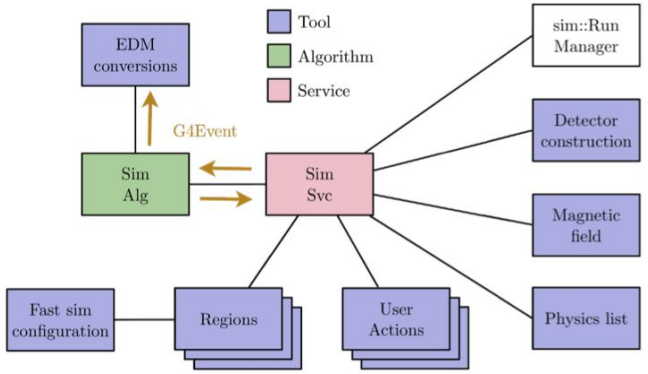
Contributions welcome/required on interfacing and testing

Experience needed for interfacing: FORTRAN, FORTRAN&C++ interplay

For testing: ability/willingness to understand settings of a given generator

Simulation

- Delphes (parametrized)
  - Gaudi interface
    - FCC EDM output
- Geant4 (fast / full)
  - Gaudi components exists to create
    - User Actions
    - Regions
    - Sensitive detectors
    - Selective output options
  - Mixing fast and full G4 simulation possible
    - SimG4Full / SimG4Fast



FCC detector concept palette for Delphes

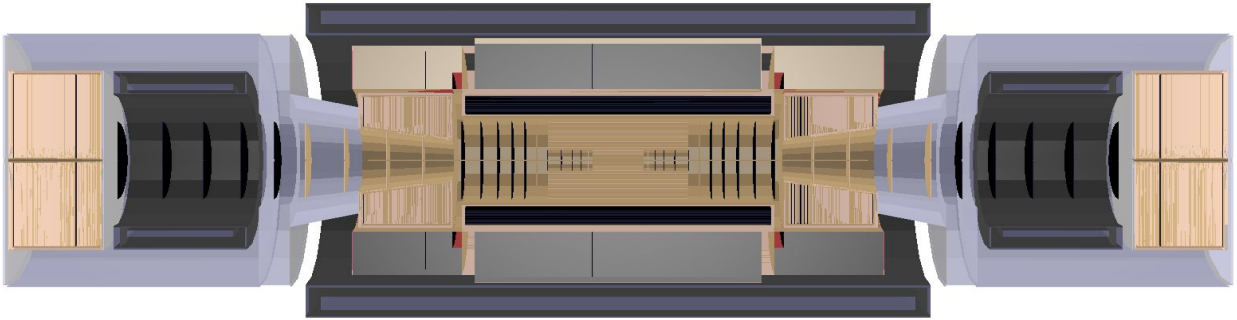
- Validated and used for CDR
  - FCC-hh baseline, HL-HELHC baseline
- IDEA, CLICDet and others available for FCC-ee
  - Not extensively used, need validation
- Latest version of Delphes includes TrackCovariance, dEdx, ParticleDensity
  - Enable simulation vertexing, b-tagging, …
  - Help developing/understanding algorithms

Possible contributions: testing, validation, fine tuning of existing cards; scripts or tool to easy variate relevant dimensions

Experience needed: familiarity with Delphes, Gaudi, simulation

FCC detector palette in DD4hep: FCC-hh

FCC-hh CDR baseline



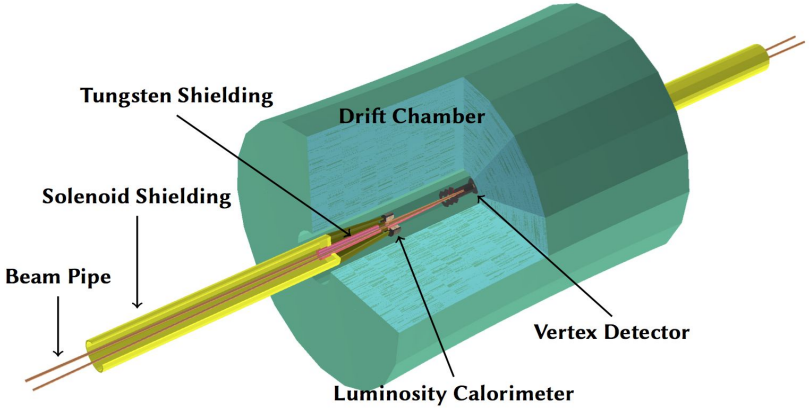
- Barrel, Endcap, Forward
- Beam Pipe, Shielding, Magnet solenoid
- Silicon Tracker
- LAr ECal, Tile HCal
- Muon System

FCC detector palette in DD4hep: FCC-ee

FCC-ee IDEA

CLD

- Beam Pipe, Beam instrumentation
- Lumical, HOM Absorber
- Vertex detector
- Drift Chamber
- Dual Readout Calorimeter
- Muon System



DR calo full simulation available in “standalone”. Integration in FCCSW/Key4hep requires:

- Translation of geometry in DD4hep format
  - Requires support for optical properties, available in DD4hep since 11/2019
- Integration of digitisation

FCC detector palette in DD4hep: FCC-ee

Possible alternatives for FCC-ee

- "IDEA" tracker with reduced version of LAr ECal + Tile HCal
  - First DD4hep description available for testing
- CLD
  - Geometry description in DD4hep exists: https://github.com/iLCSoft/lcgeo
  - Requires integration in FCCSW (digitisation modules exists in iLCSoft)

Contributions welcome/required on:
- IDEA: cross-check/complete existing stuff or provide (DR calo, muon) DD4hep descriptions and digitization
- Enabling of CLD in FCCSW: digitisation, ...

Experience needed: familiarity/willingness to learn: DD4hep, detector geometry, Geant4

Reconstruction

- Challenges: algorithm detector concept independent
  - Full flexibility, avoid duplication
- Tracking
  - Track seeding (Silicon tracker, FCC-hh), Hough Transform (drift chambers, FCC-ee)
  - Under development / investigation: ACTS integration, Conformal tracking
- Calorimeters
  - Sliding window (rectangular/ellipse), Topo-clustering
  - Under development / investigation: ML techniques

Possible contributions: tracking, vertexing, ACTS, ML, particle ID

Experience needed: familiarity with reconstruction algorithms, Gaudi, C++

What can we do for FCC-ee in Full Simulation?

- Tracking for IDEA
  - Basic digitisation producing space points
  - No integrated reconstruction algorithm available
    - Proof-of-concept of Hough Transform algorithm available as standalone script

- LAr calorimetry
  - Description + digitization available
    - Exercise in tutorials
  - Not yet calibrated

Considerations for Physics Analyses

- HEPPY: High Energy Physics with PYthon
  - Modular python framework for the analysis of collision events
  - Developed and still used for CMS

- In FCCSW HEPPY is used to
  - Process EDM events, apply-preselection, produce a flat and light ROOT ntuple
  - Analyse the ROOT ntuple
    - Not the only code used for this purpose

- Flexible but slow
  - O(100) evts/s

Considerations for Physics Analyses: RDF

- Work ongoing on a ROOT RDataFrame-based framework
- Improve efficiency going directly from EDM to final plots
  - Framework based on 4 modules to be defined by users
  - See example repository on GitHub
- Foster use of common tools
  - Web-based event shared database
  - Enables sharing of analysis modules
- Can process up to 15k evts/s w/o special optimization

Possible contributions: consolidate existing replacement prototype based on RDataFrame

Experience needed: familiarity with ROOT and RDataFrame; advanced Python

Areas of work summary

- MC generators
  - Interfacing, testing
- MDI
  - Shared formats
  - GuineaPig++ integration
  - Overlay of MDI/signal events
- Detector concepts
  - IDEA DR Calo full simulation
  - IDEA Muon system full sim
  - Validation of LAr Ecal for FCC-ee
  - Enabling of CLD in FCCSW/k4h
- Validation/testing of Delphes cards
- Reconstruction
  - Tracking algorithms
  - Vertex reconstruction
  - ACTS integration
  - ML for calo reconstruction
- Identification
  - e, mu, tau, c, b tagging / ID
- Analysis tools
  - RDataFrame based analysis
- AoB
  - Porting to other OSs
  - ...

FCC-France interest

- About 20 potential users
  - Very few developers (~10%), but, likely, with experience
- Main interest in (charged) Higgs and flavour-physics
- Start working with Delphes, while waiting for full simulation
- Mostly interested in
  - Calorimeters and vertex detectors
  - Calo objects and PF algorithms, vertexing b-tag and tau reco
  - Performance of ML, reconstruction

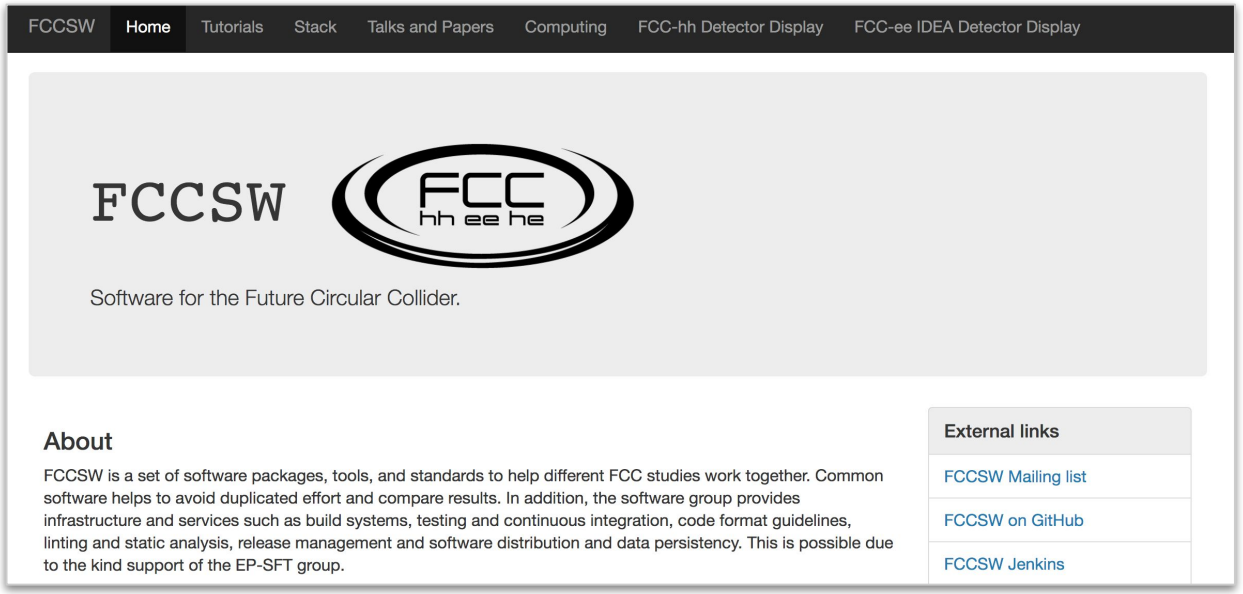
- Most urgent/important for software:
  - Flexibility, support full simulation, full palette of generators

Summary

- Software is essential during this phase of the project
  - No CDR+/TDR without a robust software
  - Should be carefully designed for long term usage
  - The current software stack, assembled using as much as possible existing components, served well the purposes of the CDRs
- New phase very challenging
  - Unprecedented level of precision expected at FCC-ee
  - Potentially orders of magnitude more Monte-Carlo than LHC
- Try to get as much as possible from the community
  - Following closely, participate-to, collaborate-w/ common activities {Key4hep, EDM4hep}
- Everyone should feel concerned
  - Immediate areas of work identified

Thank you!

- Web site https://cern.ch/fccsw

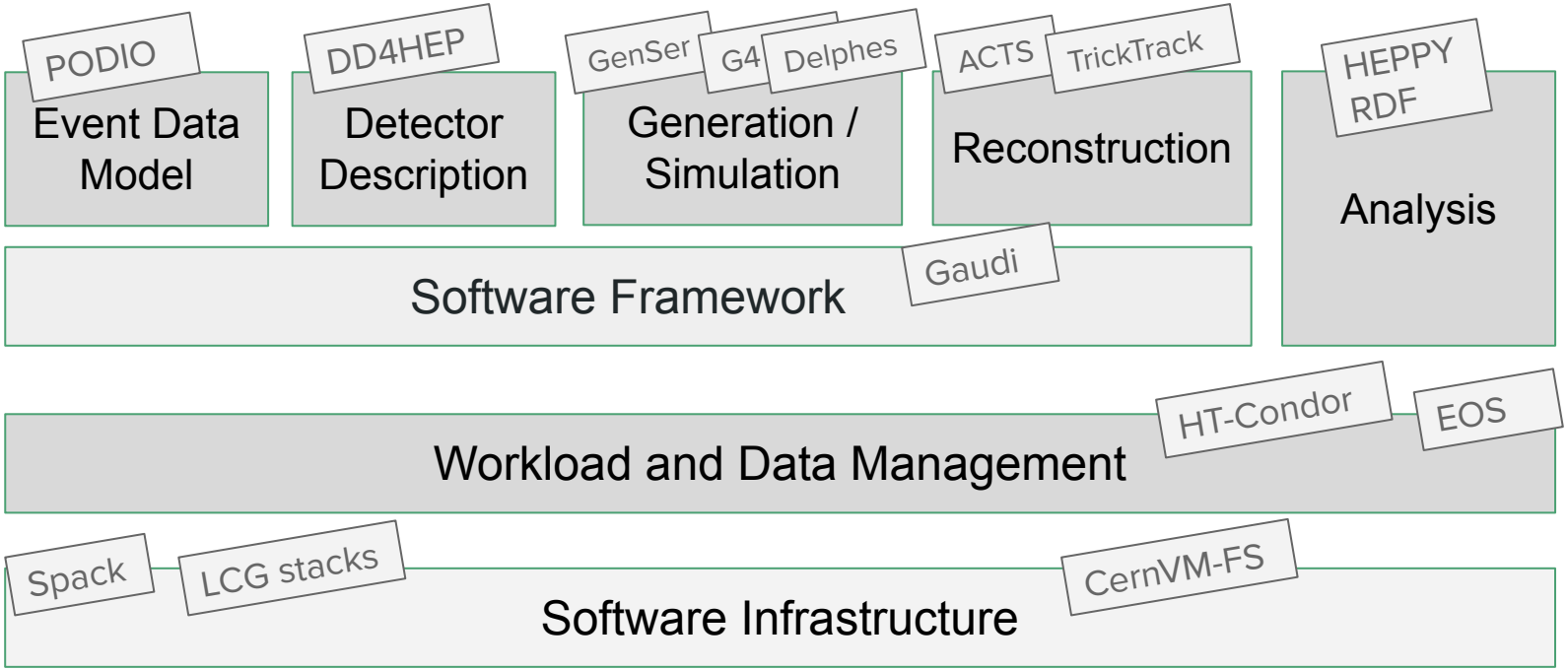


Backup

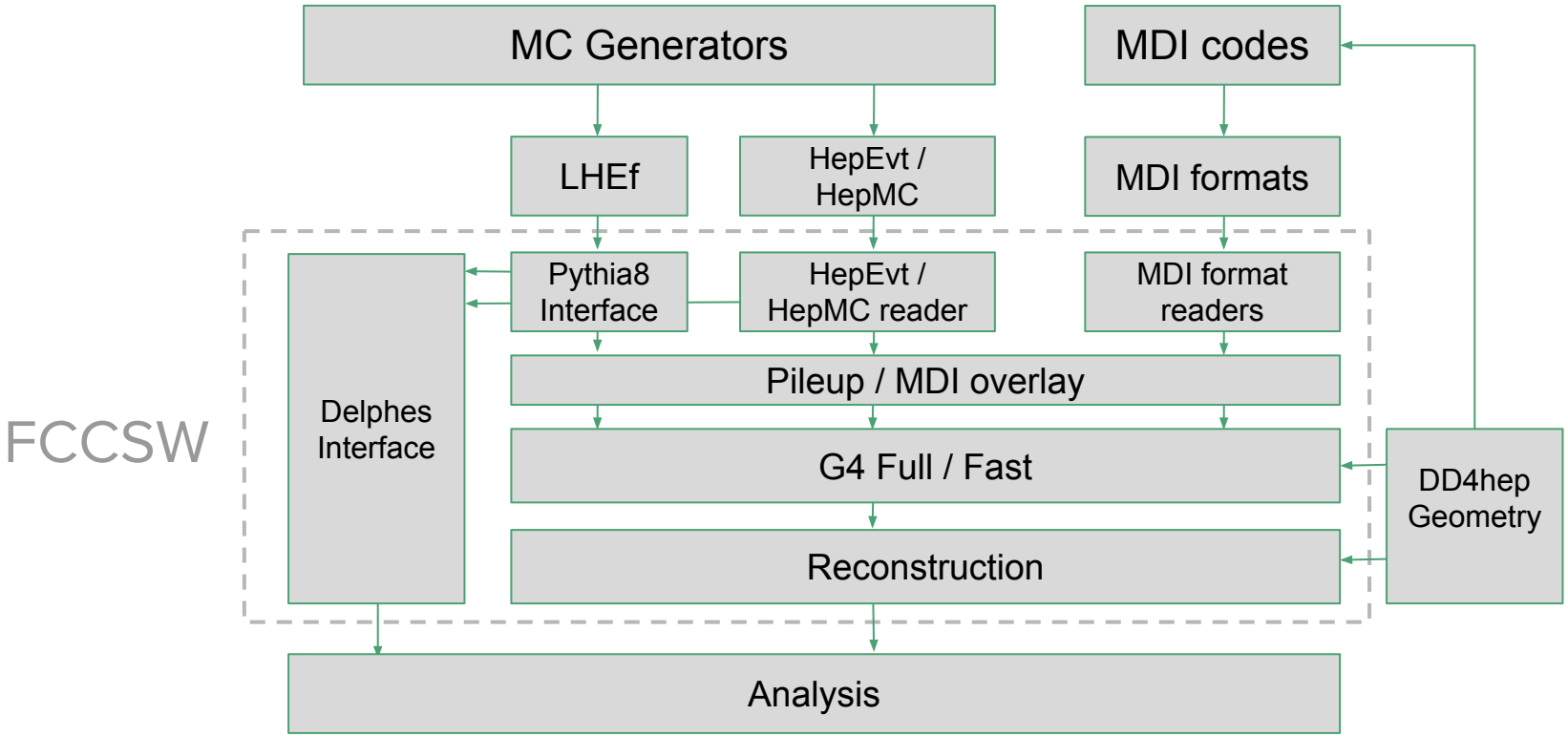
Key4hep/EDM4hep and Delphes

- EDM4hep v0.1 available
  - Should be OK for FCC-ee
- Key4hep v0.1 includes K4FWCore steering component
  - Equivalent, and derived from, to FWCore in FCCSW
- Prioritization of the DelphesInterface module addition under discussion
  - Derived from FCCSW SimDelphesInterface
- Ideally this could be used for Snowmass Delphes studies

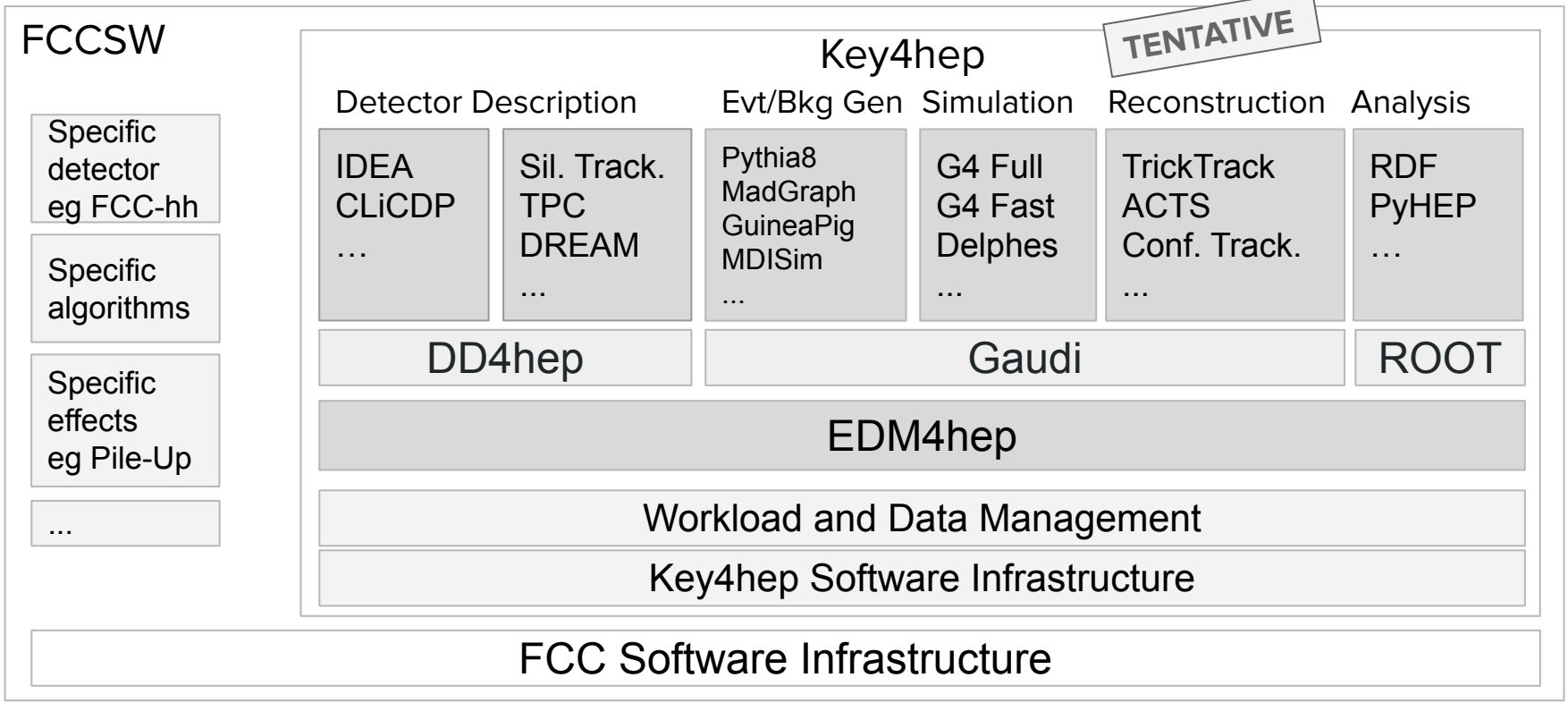
Available components



Typical workflows



Connection with Key4HEP

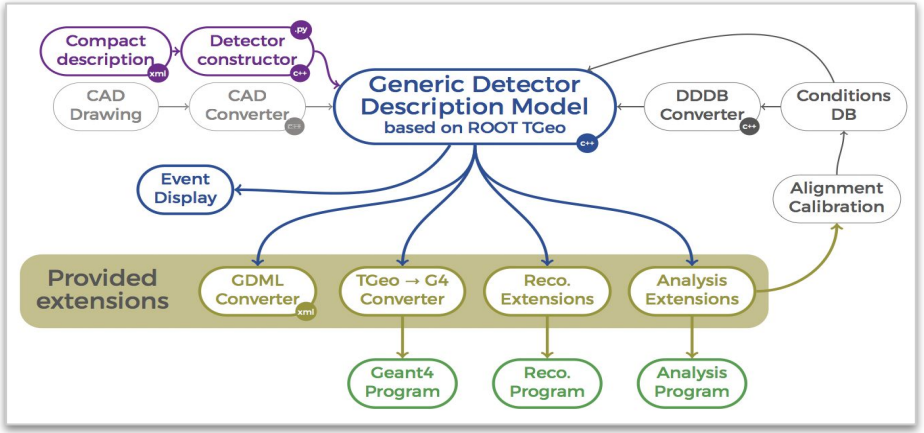


Event Data Model

- Current FCC-EDM
  - Event/Run: EventInfo
  - MC truth: MCParticle, GenVertex, GenJet
  - Tracker: Track (PositionedTrackHit, TrackCluster, TrackState)
  - Calorimeter: CaloCluster (PositionedCaloHit)
  - Associations: ParticleMCParticleAssociation, DigiTrackHitAssociation, CaloHitAssociation, CaloHitMCParticleAssociation
  - High-Level objects: TaggedParticle, Vertex (WeightedTrack), TaggedJet, ResolvedJet, MET
- Tuned on the needs of FCC-hh
  - High-level objects of LHC inspiration
- TaggedParticle contains cross association between tracks and calo objects

Detector Description: DD4hep

- Generic detector view appropriate to support
  - Simulation, reconstruction, analysis, …
- Design goals
  - Complete detector description
  - Single source of information
  - Support all stages of the experiment
  - Easy of use

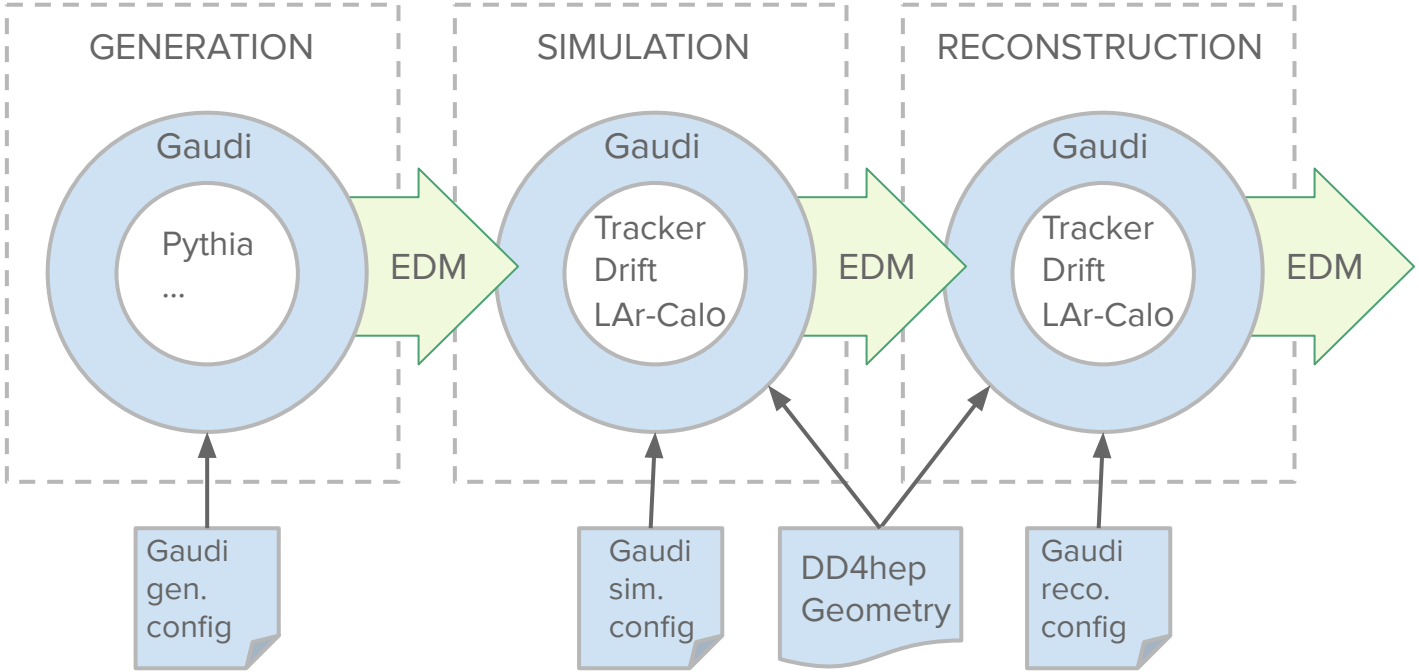


- Part of AIDA2020
- Used by CLIC, ILC, FCC, LHCb, CMS, SCT

Software Framework: Gaudi-based

- Framework toolkit to provide required interfaces and services to build HEP experiment frameworks
  - Opensource project and experiment independent
- Data processing framework designed to manage experiment workflows
  - Separate data and algorithms; well defined interfaces
  - User's code encapsulated in Algorithm's, Tool's / Interface's, Service's
  - Different persistent and transient views of data
  - C++, with Python configuration
- Originating from LHCb, Gaudi is adopted also by ATLAS
  - Actively developed to face LHC Run 3 and Run 4 challenges (high PU)
- Using the latest Gaudi version (v32r2).

Gaudi and FCCSW



Gaudi and FCCSW

- Python scripts to configure tools and algorithms and define their processing order

```
$ fccrun \                                                  # Generic Gaudi app
      ./Examples/options/geant_fullsim_fccee_pgun.py \      # job definition
      --energyMin=10  --energyMax=10 --particleName="mu-" \ # parameters
      --n=1000 --filename=fccee_idea_mu.root
```

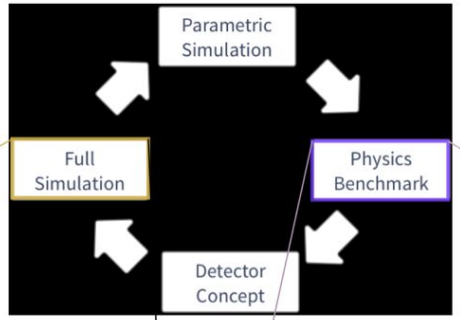
- Each algorithm defines its parameters which can be overwritten of the command line

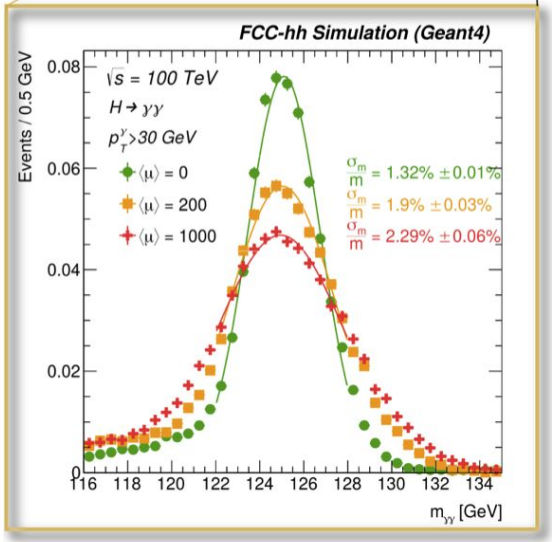
Fast / Full Simulation Interplay

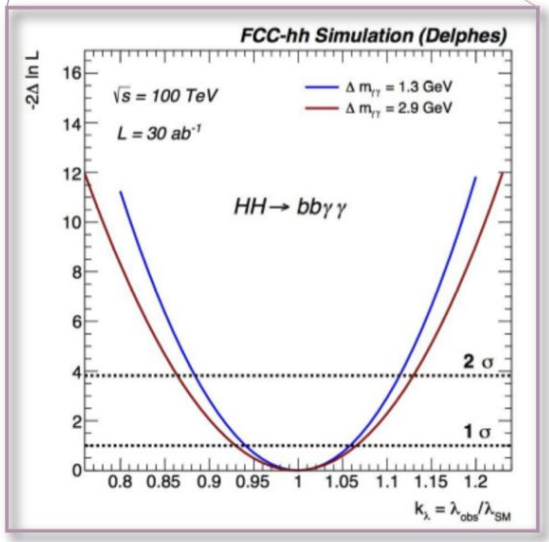
Example:

Higgs self-coupling

@ FCC-hh







MDI integration: status and areas of work

- Agreement to define shared documented data formats
  - Work not started yet
- GuineaPig++ interfaced to iLCSoft
  - May be used as source of inspiration
- Event overlay in FCCSW
  - Same technology used for pileUP
  - Need to be tested and validated

Contributions required to all the above items
Experience required: file formats, FORTRAN, C++

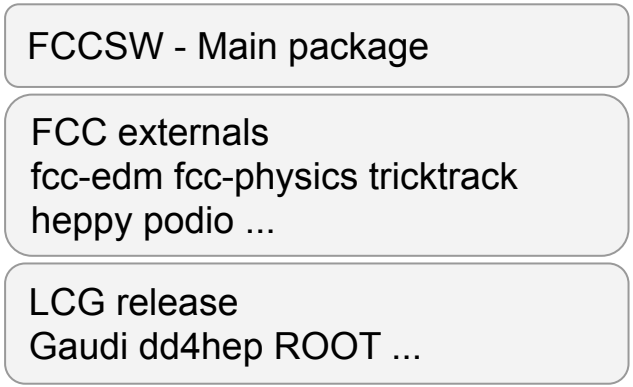
Software Infrastructure

- Typical HEP development workflow

- Deliverables
  - FCCSW
  - Externals: FCCSW specific dependencies
  - Based on LCG releases provided by EP-SFT

- Builds (nightlies, releases) managed by Spack package manager
  - Good feedback to HSF packaging WG

- Deployment on dedicated CernVM-FS repositories
  - /cvmfs/fcc.cern.ch/, /cvmfs/fcc-nightlies.cern.ch/



CERN resources and access policy

- CERN resources are available to member of institutes having signed the Memorandum of Understanding and its addendum
- EOS areas for data or large files: /eos/experiment/fcc
  - Current quota: 400 TB
  - E-group membership: fcc-eos-access (and alike)
  - Dedicated areas for ee, hh, eh, helhc, users
    - Plan to deprecate ‘users’: each CERN user has 1 TB at /eos/user/*u*/*username*
    - Needs to be enabled on Account Management page
- EOS areas for shared files: /eos/project/f/fccsw-web/www
  - Also accessible also via web
- Dedicated queue on LXBATCH
  - AccountingGroup = "group_u_FCC.local_gen" (on HTCondor)
  - E-group membership: fcc-experiments-comp