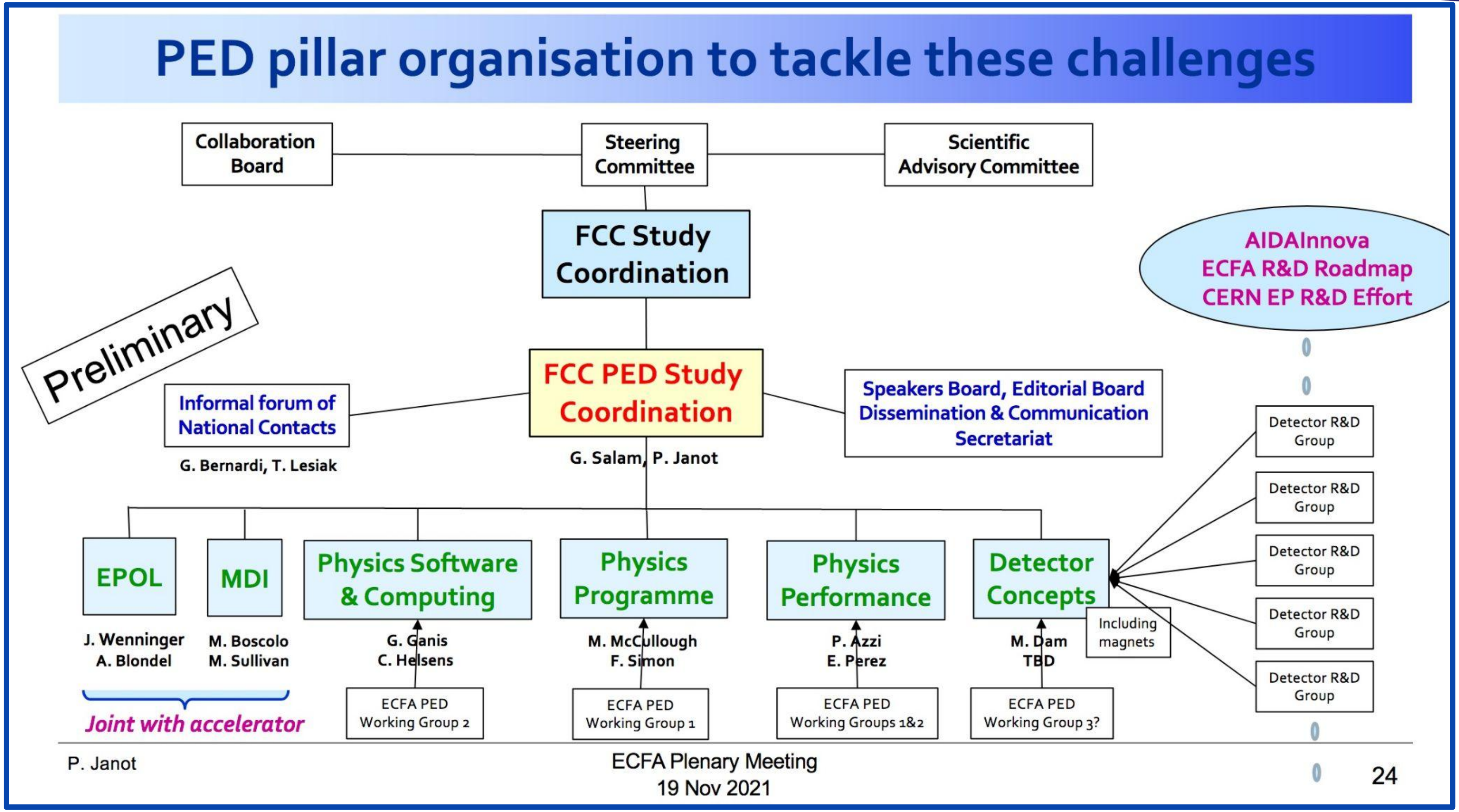
Overview of the Software for FCC

3rd FCC-France workshop

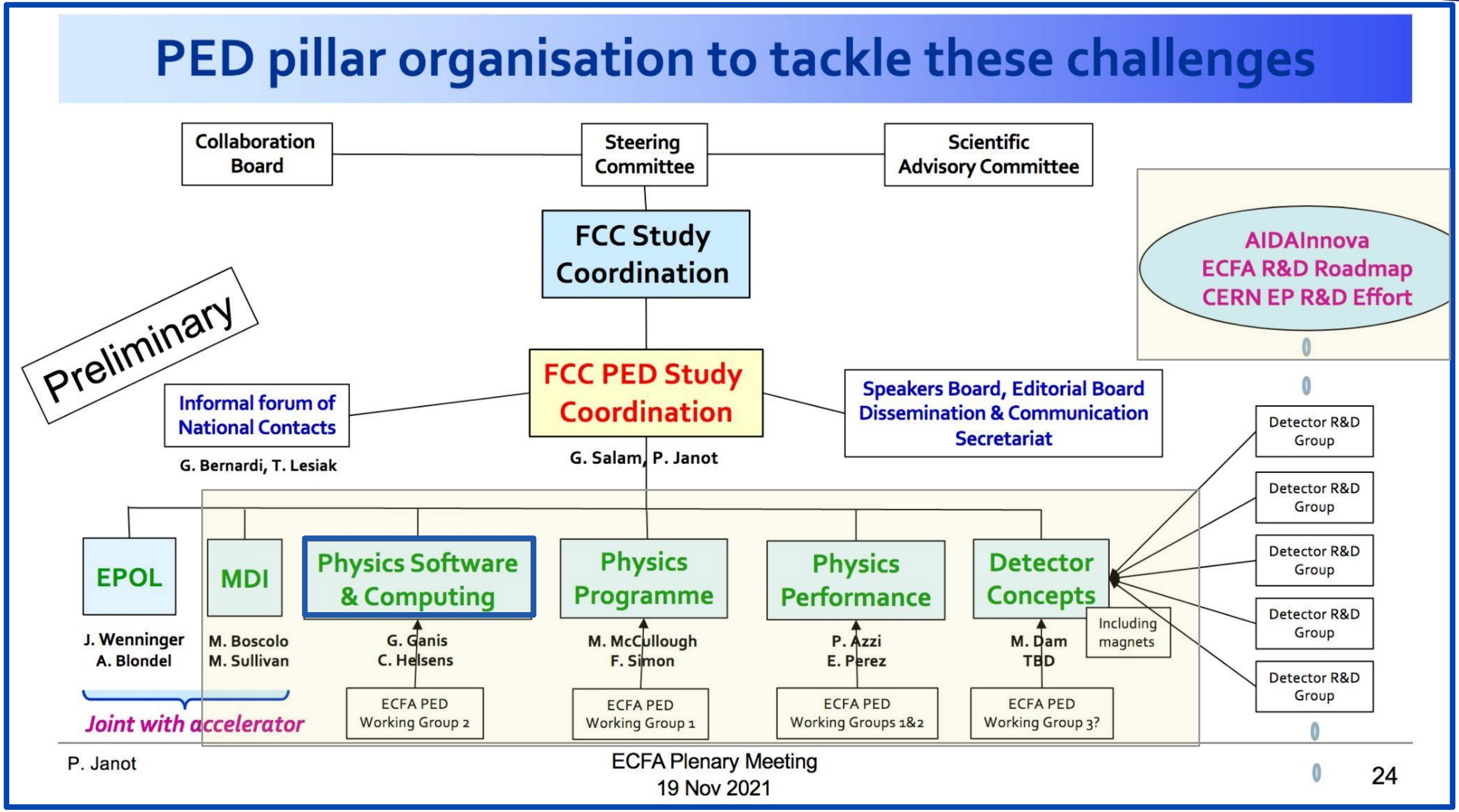
Nov 30, 2021

C Helsens
CERN-EP

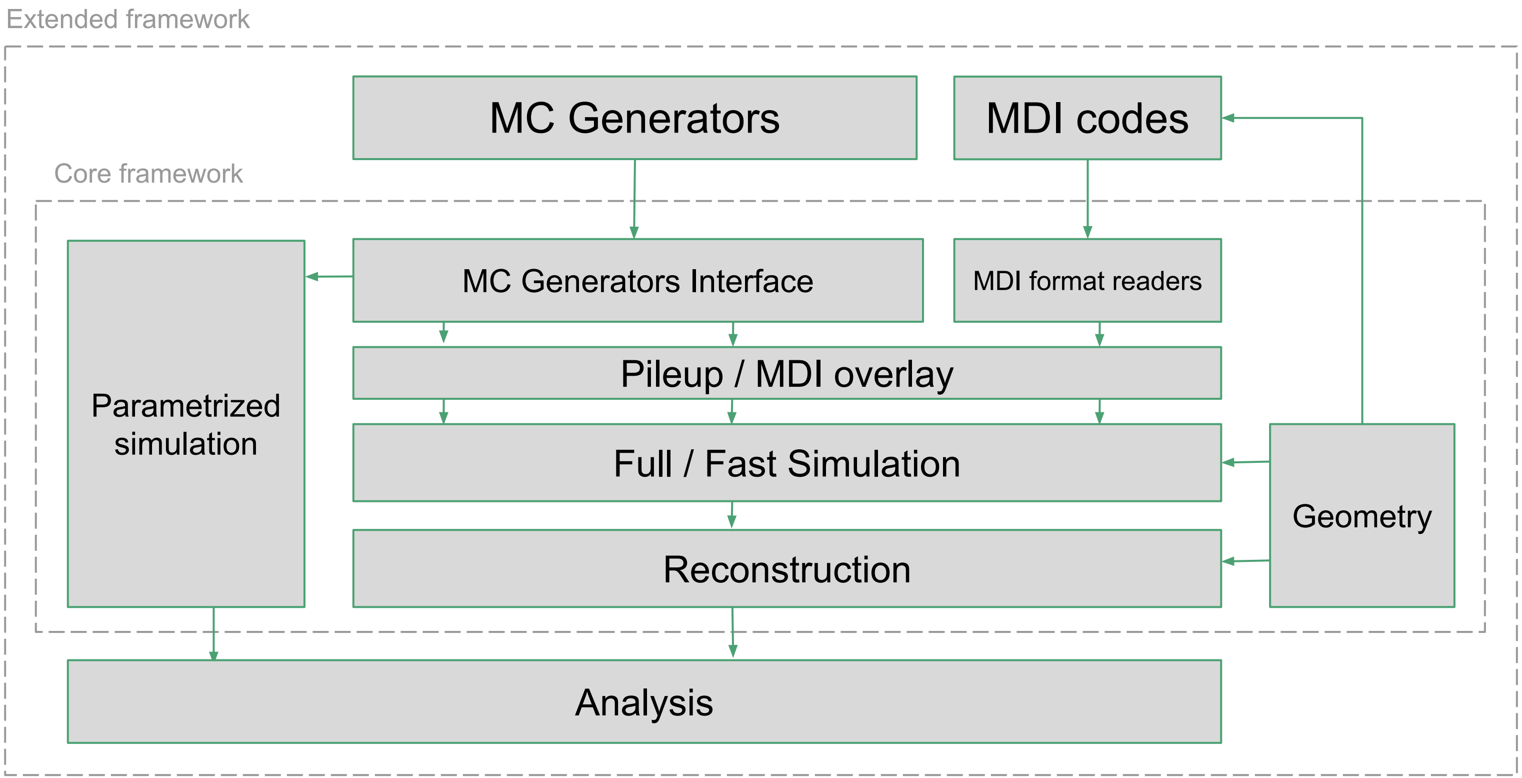
Role and relation with other groups



Role and relation with other groups

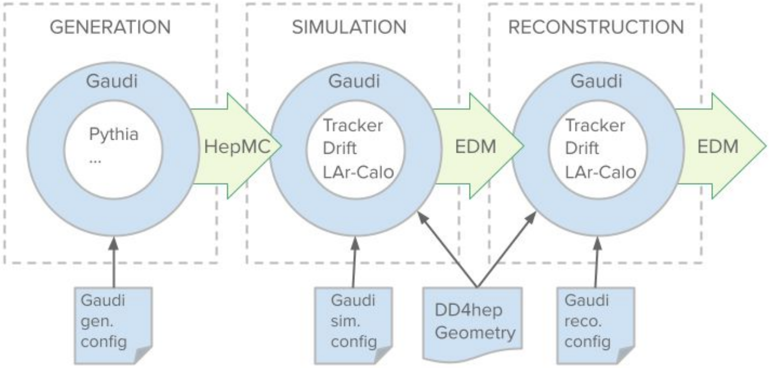


Typical workflows to support



FCCSW approach

- Started in 2014
- Driving considerations
  - One software stack to support all the cases (hh,ee,eh), all the detector concepts
  - Need to support physics and detector studies
    - Parametrised, fast and full simulation (and mixture of the three)
  - Modularity: allow for evolution
    - Component parts can be improved separately
  - Allow multi-paradigm for analysis
    - C++ and Python at the same level
- Adopted Strategy
  - Adapt existing solutions from LHC (Gaudi, …)
  - Look at ongoing common R&D projects (AIDA)
  - Invest in streamlining of event data model
- Focus on FCCee after CDR (2019)



The common software vision

Create a software ecosystem integrating in optimal way various software components to provide a ready-to-use full-fledged data processing solution for HEP experiments

Complete set of tools for

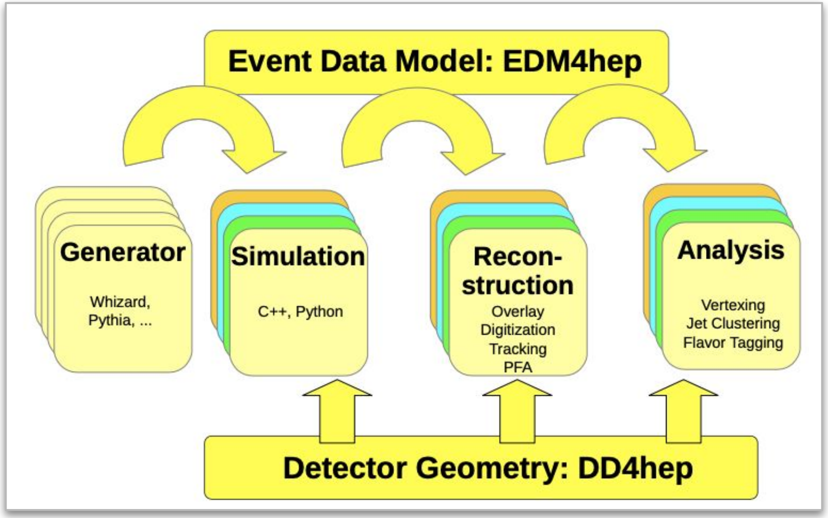
- Generation, simulation, reconstruction, analysis
- Build, package, test, deploy, run

Core Ingredients of current key4hep

- PoDIO for EDM4hep, based on LCIO and FCC-edm
- Gaudi framework, devel/used for (HL-)LHC
- DD4hep for geometry, adopted at LHC
- Spack package manager, lot of interest from LHC

Community project, unifying efforts

- Contributions from CLIC, ILC, FCC, CEPC
- And interest from STCF, muon collider, ...

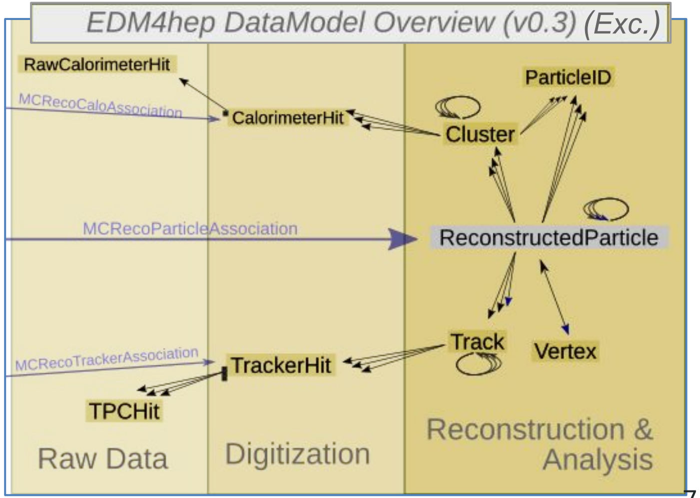


Kick-off meetings in Bologna, Hong Kong

The common event data model: the challenges

EDM provides common language for exchange among framework components

- Challenge 1: efficient support different collision environments (e^+e^-, pp, ...)
  - Positive first experiences with FCC-hh components
- Challenge 2: keep I/O efficient
  - PoDIO: separate definition from implementation, facilitate optimal adaptation to backend
    - POD layer designed for efficient I/O, simple memory layout
    - Flat data support (RNTuple) will provide insight
- Challenge 3: efficient support for schema evolution
  - Requires schema evolution in PoDIO, planned
- Challenge 4: efficient support for detector needs
  - Interaction w/ detector teams from the start
    - Eg. cluster counting for IDEA Drift Chamber



FCCSW @ key4hep adoption process

- Already Gaudi based
  - Move FCC-edm to EDM4hep
  - Re-arrangement and modernization
  - Some components considered for migration to key4hep
    - Generation, simulation, reconstruction, …

- Migration status

old FCCSW (version ≤ 0.16)	Key4hep	new FCCSW	status	migration
FWCore	k4FWCore		done	yes
Sim/SimDelphesInterface	k4SimDelphes		done	yes
Generation		k4Gen	done	under evaluation
Sim		k4SimGeant4	done	under evaluation
Reconstruction/Rec[...]Calorimeter		k4RecCalorimeter	done	under evaluation
Reconstruction/RecDriftChamber		to be determined		
Detector		FCCDetectors	done	no, FCC specific
	to be determined	dual–readout		under evaluation

Monte Carlo Generators

- General purpose generators available
  - Whizard, Pythia8, MadGraph, Sherpa, ...
- Legacy LEP generators still state of the art at the {Z, WW} energies
  - “Archeological” work to recover KKMC, BHLUMI, BabaYaga, ...
- Main software challenges
  - Interfacing with the framework
    - Through common data format, e.g. LHEF, or callable interface (Pythia8)
  - Availability in the shared software stacks
    - Private codes, unversioned tarballs, … version control issues
  - Early interactions with MC groups to ease the interface and facilitate feedback loop
- Other requirements relevant for FCCee
  - Uniform treatment of beam parameters (beam energy spread, crossing angle)

Connections w/
- Physics Programme
- Physics Performance
- ECFA
- Phenomenologists
- Key4hep

See also
Review at the ECFA kick-off (W. Kilian)

ECFA 1st Topical Meeting on Generators (here)

Beam and MDI-related backgrounds

- FCCee interaction region design requires deep level of understanding of the detector backgrounds
  - Only achievable with integration in experiment software

Connections w/
- MDI study group
- Physics Performance
- Detectors

- Several codes for modeling the processes, including
  - GuineaPig, MDISim, SynRad, Sync_Bkg, Pythia, Whizard, ...

  Codes not always in public repositories, outputs in different, non-standard formats

- Target: supercode providing common interface to relevant codes
  - Framework integration to unify/simplify access with controlled configuration and normalization
  - On going effort, example w/ GuineaPig available

- Consistent description of the relevant geometry elements
  - Requires interplay between detector and machine geometry formats (e.g. CAD)

Geometry description: DD4hep

- FCCDetectors contains what is currently available
    - Elements of the interaction region (BeamPipe, Instrumentation, HOMAbs, LumiCal)
    - CLD (CLIC detector for FCC)
    - Simplified Elements for IDEA (no calo, no muon)
    - Simplified Elements for IDEA with FCChh adapted calo (LAr/Tile) (no muon)
    - FCChh baseline (+ some variants)

- Standalone implementations
    - IDEA Dual-Readout calorimeter: dual-readout
    - IDEA Drift Chamber and muon chambers not started

Connections w/
- Detectors
- Physics Performance

Completing the detector palette in DD4hep

- DD4hep allows easily replacement of parts
  - Example is a reduced version of the FCChh LAr ECal + Tile HCal to be evaluated, for example, together with IDEA tracking system
    - First DD4hep description available for testing
  - Plug&Play in place replacement technology to be consolidated and streamlined

- New sub-detector concepts must be integrated in key4hep/FCCSW
  - Part of the FCC Detector Concept mandate:
    - Promote the use of the common FCCSW software platform & tools, including the development of the sub-detector geometrical description, simulation, and local reconstruction;
    - Integrate sub-detectors into detector concepts: a plug-and-play technology is offered by the key4hep software framework;

Parametrized Simulation

- Current studies are mostly based on Delphes
  - Includes a tracking system, embedded into a magnetic field, calorimeters and a muon system
    - TrackCovariance, dEdx, ParticleDensity: enable realistic algorithms for vertexing, b-tagging, …
    - Effect of magnetic field, granularity of calorimeters, sub-detector resolutions
    - Also outputs observables such as isolated leptons, missing energy and collection of jets
  - Interfaced to standard file formats (e.g. LHEF, HepMC)
  - Key4hep provides Delphes interfaces/executables producing EDM4hep output
    - Standalone executables, e.g. DelphesPythia8_EDM4HEP; Framework integration
  - Palette of detector concepts for e^+e^- available

- SVG being also considered in key4hep
  - Potentially more complete and faster
  - Needs some adaptation work (EDM4hep output, …)

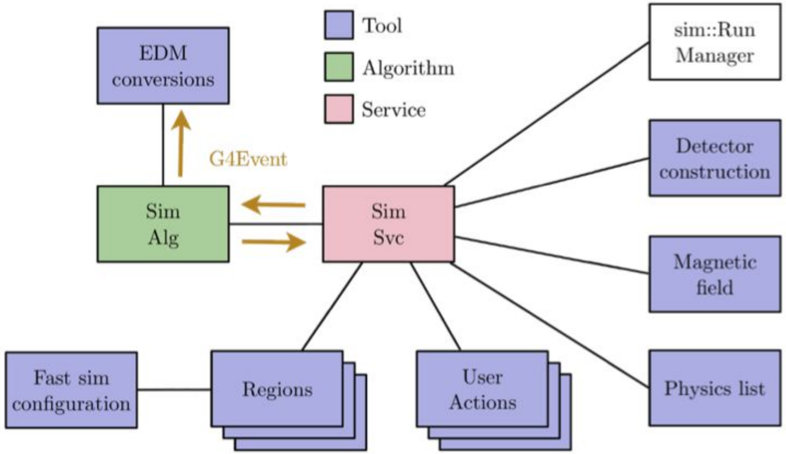
Connections w/
- Physics Performance
- Detectors

Simulation

- 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

- k4SimGeant4
  - Gaudi module ready to be migrated to key4hep
  - Adapt to the final key4hep choice
    - FCC, CEPC, LC approaches being discussed

Connections w/
- Key4hep



Reconstruction

- Little specific to FCC-ee
  - Tracking and calorimetric algorithms for baseline FCC-hh
  - Full sim studies for FCCee not really started
- Lots of algorithms available for iLCSoft
  - Accessible through LCIO to/from EDM4hep on the fly converter
    - Enables initial studies and evaluations
    - Base / reference for native implementation when required
- Need to integrate algorithms attached to a given detector concept
  - E.g. IDEA Drift Chamber or Dual Readout calorimeter
- Framework integration of general purpose tools such as ACTS, PandoraPFA, CLUE/TICL, ...

Connections w/
- Physics Performance
- Detectors
- Key4hep

Analysis: FCCAnalysis

More in Thomas talk's

- Replaces the fully pythonic HEPPY framework used for CDR
  - Good functionality but extremely slow
- Based on RDataFrame, new ROOT paradigm aimed for (HL-)LHC
  - Python framework with C++ backend
  - Bridges the gap with LHC involved people
- Runs on EDM4hep, non FCC specific
  - Prototype of generic analysis framework

Connections w/
- Physics Performance
- Key4hep

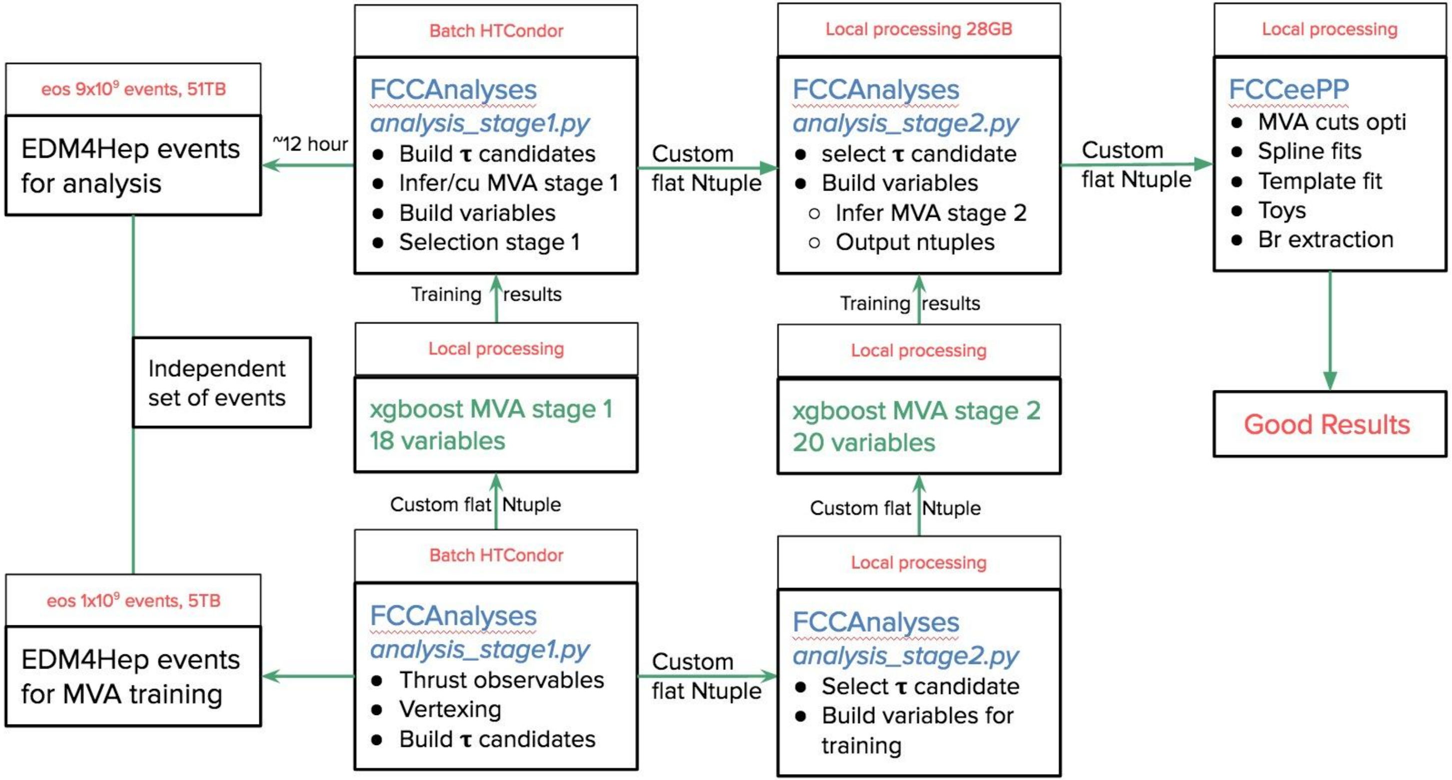
Analysis configuration
4 **python** scripts to configure:

1. Samples to run over
2. Functions/algorithm to call
3. Event selection
4. Plotting configuration

Common utility functions, algorithm, etc…
C++ library

Common interface code
Sample database, RdataFrame, plotting
Python

First published analysis with FCCAnalysis: $B_c \to \tau^+ \nu_\tau$



Involves:
- Distributed processing
- Storage
- ML Training
- Fitting
- Local processing
- ...

Software infrastructure

- Event Producer Workload and Data Management
  - Currently still using home-made solution (HTCondor, EOS, ...) - See Physics' talk
  - Moving towards DIRAC (via iLCDirac)
    - Provides web and command line interfaces, native and advanced support for data management, …
    - Ready for experimental runs at CERN; CINECA site being added

Connections w/
- Key4hep
- Physics Performance

- Build / packaging / testing / deploying
  - Now fully relying on Key4hep
    - Spack for build/packaging, CernVM-FS for deployment; GitLab CI for continuous integration
  - Initial build/deploying infrastructure (extension of LCG builds) being phase-out

A few considerations on the resources needed

- The run at Z peak sets the scale
  - ≈10^{12} evts, 3-6 EByte storage, 10 MHS06 CPU (≈ current ATLAS yearly needs)
- These numbers are similar to the ones expected for (HL-)LHC
  - Do not expect issues for operations in 2040 and beyond
- For the FSR the situation is different
  - Analysis at Delphes level are possible (see $B_c \rightarrow \tau^+ \nu_\tau$)
  - Full simulation of all components require 10^3-10^4 times more
- Techniques of overcome this limitations are required
  - E.g. interplay of full and parametrized simulation (see next)
- Planned community improvements in fast simulation very welcome
  - Possible improvements of the parametrized simulation treatment of critical parts such as calorimetric object could also be envisaged / investigated
    - E.g. based on improvements of fast simulation à la Geant4/GFlash or Machine Learning / GAN

A few considerations on the resources needed

- The run at Z peak sets the scale
  - ≈10^{12} evts, 3-6 EByte storage, 10 MHS06 CPU (≈ current ATLAS yearly needs)
- These numbers are similar to the ones expected for (HL-)LHC

Computing needs were estimated in an essay prepared in the context of a FCC wide EPJ+ contribution on FCC-ee challenges

  - Full simulation of all components require 10^3-10^4 times more
- Techniques of overcome this limitations are required
  - E.g. interplay of full and parametrized simulation (see next)
- Planned community improvements in fast simulation very welcome
  - Possible improvements of the parametrized simulation treatment of critical parts such as calorimetric object could also be envisaged / investigated
    - E.g. based on improvements of fast simulation à la Geant4/GFlash or Machine Learning / GAN

Scale of Monte Carlo productions done

- Spring 2021 production, EDM4hep
  - Delphes events IDEA with *Track Covariance full matrix lower triangle*
  - Total: ~10^{10} events, ~53 TB, mostly at Z peak

- Existing productions for FCC-hh (100 TeV)
  - EDM4hep
    - Total: ~$1.3\text{x}10^8$ events, ~14 TB
  - Old fcc-edm
    - Delphes: ~$2.1\text{x}10^9$ events, ~93 TB
    - Full sim, single particle: ~$2.4 \cdot 10^9$ events, ~160 TB

- Stored at CERN on EOS
  - /eos/experiment/fcc/{ee,hh}/generation/DelphesEvents/

Relation with R&D activities

- CERN EP software R&D
  - All carried-on task/activities are connected to FCC needs
  - Key4hep is crucial, but all the other carried-on activities are connected
    - E.g. (ML-based) fast simulation of calorimeters, RNtuple-related analysis improvements
- AIDA Innova
  - Very similar palette of software R&D activities
  - Could potentially also profit of some person power for specific tasks
- ECFA R&D
  - Indirect connection with WG2 (Physics Analysis Methods)

Priorities, looking forward

- Targets: FSR by end 2025, analysis completed by end 2024 / early 2025
- S&C milestones (as presented at EP management on Nov 2020)
  - End 2021
    - Completed migration to common software EDM4hep / key4hep ✓
    - Minimal set of MC generators available ✓
    - Delphes infrastructure ready and documented ✓≈
    - Minimal set of detectors (geometry, digitization, reconstruction) implemented ✓≈
    - Needed computing resources evaluated ✓
  - Early 2022
    - First version of “supercode” for realistic estimation of MDI related backgrounds
    - Comprehensive documentation of the whole (analysis) chain available
  - End 2022
    - Commissioning of large scale production with the whole setup

Priorities, looking forward

- Consolidate / enable :
  - Analysis workflows with Delphes in particular documentation
  - Full simulation with CLD w/ reconstruction through LCIO converter / iLCSoft algorithms
  - Commission DIRAC-based production infrastructure

- Detector descriptions for full simulation
  - Streamline sub-detector replacements exploiting DD4hep features (PlugnPlay)
  - Missing IDEA components in DD4hep / FCCDetectors
    - Adopt Dual Readout existing implementation
    - Foster implementations for Drift Chamber, Vertex detector and Muon chambers
- Event reconstruction
  - External packages such as ACTS (tracking), CLUE (clustering), Pandora (Pflow)
  - Support custom reconstruction when needed (such as PID in a RICH, pi0, …)

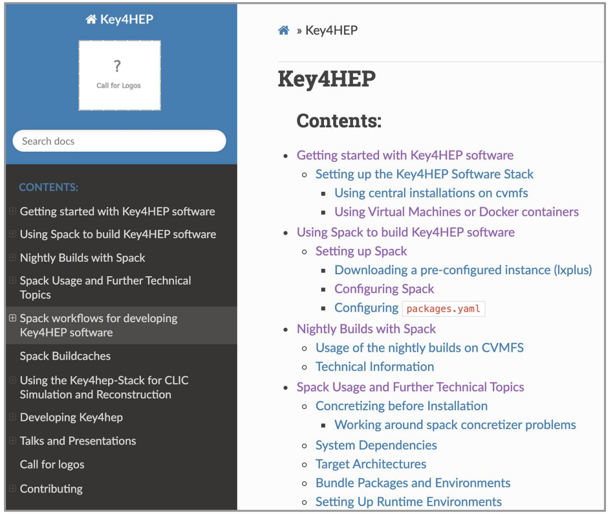
Priorities, looking forward

- Investigate and possibly import improvements in fast realistic simulation of calos
  - In connection with EP R&D, AIDAInnova
- Continue revision and interface of MDI codes
  - Includes finding a solution for CAD to/from DD4hep
- Continue to investigate state of the art analysis solution
  - Adopt them as early as possible
- AoB
  - Event Display
    - Phoenix ? Eve based?
  - ...

Useful pointers

- Project repositories
  - GitHub: https://github.com/HEP-FCC
  - CernVM-FS: /cvmfs/fcc.cern.ch
- Forum
  - https://fccsw-forum.web.cern.ch/
- Existing documentation
  - https://hep-fcc.github.io/fcc-tutorials/index.html

Documentation, tutorials, …



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

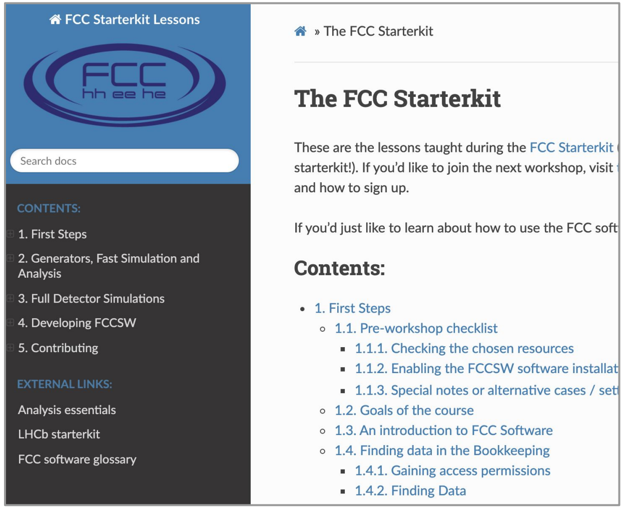
» Key4HEP

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

Key4hep [GitHub Project](#)
[Main documentation page](#)
[Doxygen software documentation](#)



FCC Starterkit Lessons

Search docs

CONTENTS:

1. First Steps
2. Generators, Fast Simulation and Analysis
3. Full Detector Simulations
4. Developing FCCSW
5. Contributing

EXTERNAL LINKS:

- Analysis essentials
- LHCb starterkit
- FCC software glossary

» The FCC Starterkit

The FCC Starterkit

These are the lessons taught during the FCC Starterkit starterkit!). If you'd like to join the next workshop, visit and how to sign up.

If you'd just like to learn about how to use the FCC soft

Contents:

- 1. First Steps
  - 1.1. Pre-workshop checklist
    - 1.1.1. Checking the chosen resources
    - 1.1.2. Enabling the FCCSW software installat
    - 1.1.3. Special notes or alternative cases / set
  - 1.2. Goals of the course
  - 1.3. An introduction to FCC Software
  - 1.4. Finding data in the Bookkeeping
    - 1.4.1. Gaining access permissions
    - 1.4.2. Finding Data

FCCSW [GitHub Project](#)
[Main documentation page](#)