News from the IR workshop — LHeC detector simulation

ESPP white paper preparation meeting – November 15, 2024

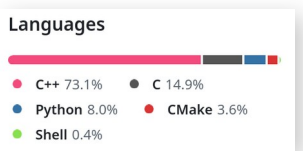
L. Forthomme (AGH University of Kraków), on behalf of the LHeC IP SR study working group

Introduction – needs for a software stack

- **Main goal**: develop an offline software framework for future LHeC/FCC-eh developments
  - To be used for **simulation** (interfacing various e-p event generator + generic input formats: HepMC/LHEF/HEPEVT/…)
  - To be used for **data handling** (I/O, conditions storage+transient event model definition)
  - To be used as **reconstruction tool** (interfacing various algorithms implementations)
- **Software requirements**:
  - Use of commonly, or soon-to-be widely used libraries and tools (for geometry, simulation, clustering/tracking/vertexing/… reconstruction) → **do not "reinvent the wheel"**
  - Handle **multiple geometry schemes** studied in the past (O(20) handled scenarii at the time being)
  - Aim for modularity: platform-independent, **"plugin-based"** development of independent components, adapt to the SW availability from "any" stack present on the build system
  - Ability to handle multiple **I/O formats** for data handling (ROOT TTrees/RNtuples, RAW files, hdf5?, …)
- Several data handling (I/O, transformation modules, …) framework studied
  - To quote a few, considered both "CMSSW-like" options (*art* framework/"stripped" CMSSW version), or ATLAS/FCC options (*Gaudi*-based)

State of the art – a bit of technicalities…

- **Interfacing to Key4hep** of P. Koska's fork of DD4hep (P. Kostka) including LHeC/FCC-eh detectors geometry
  - **Gaudi-based** offline SW framework, C++ modules and Python scripts steering
  - Fully compatible with FCC-ee/FCC-hh/CEPC/… simulation/reconstruction tools, better handling of **multi-threading** than standalone DD4hep
  - **Geometry definition** & tool sets (surfaces definitions, utilities…) still handled by **DD4hep**
  - Reusing major **data formats definition** from Key4hep/EDM4hep (SimHits/(Rec)Hits/DIGIs/…) inheriting from podio ; object derivatives can be defined for LHeC detector-specific usages
- Gaudi-translation of the few "producer" modules already mentioned for "single-config" event generation:
  - "trivial" particle gun, BDSIM scorer planes output, Pythia 8, CepGen (Pythia 6 + Sherpa interfaces in preparation)
  - Geant4 propagation into all sensitive volumes, currently relies on "stock" tracker/calorimeter hits collection production (flexible ; can be customised to detectors-specific collections storing additional Geant4 attributes for e.g. DIGI/RecHits conversion)
- Current repository location: https://github.com/forthommel/lhecsw
  - To be migrated e.g. under the LHeC Github organisation, or CERN's GitLab



Snapshot as of Sep. 2024

Example steering file – Pythia 8 configuration (→ RecHits)

- **Python steering** of simulation/reconstruction jobs
  - Combination of standard Gaudi Configurables options and LHeC framework-specific includes (can be used to define a common/shared set of algorithms + parameterisation for future studies)
  - Attempt to automate **internal conversion** between transient data formats (e.g. Pythia 8 → HepMC3 → Gaudi)
  - **Output data model** definition from "standard" podio library, with ROOT (TTree/TNtuple) and SIO (SLAC) formats I/O management
- Implementation of first Geant4 SimHits → (Rec)Hits **conversion algorithms**
  - Currently handling a few algos w/ Geometry-sentient spatial/temporal resolution smearing for vertex/pixel trackers (SimAlgos/Tracker), and energy-smearing for calorimeters (SimAlgos/Calorimetry) ; more to follow
  - All collections of interest can be saved and reused for later stages of processing ; standard "producer/consumer" I/O structure

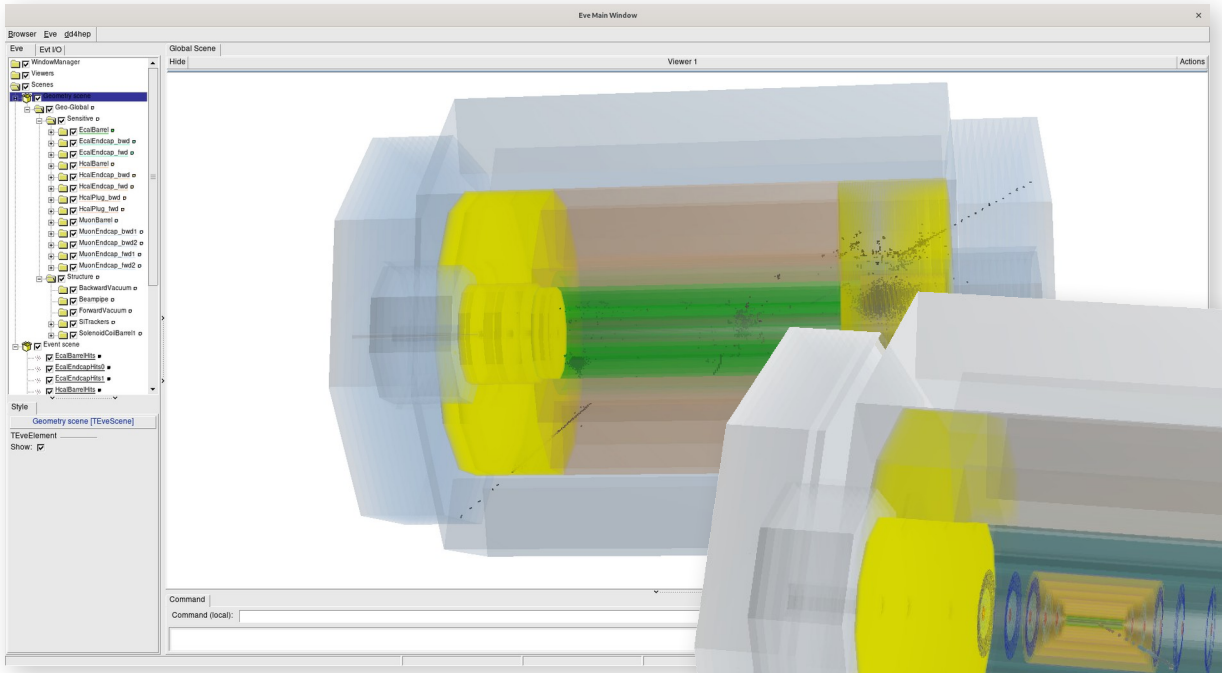
```
from Gaudi.Configuration import *
from Configurables import GenAlg
from Configurables import PodioOutput, FCCDataSvc
from Configurables import ApplicationMgr

from Generator.pythia8Interface_cff import *
from Geometry.geoservice_cfi import geoservice
from SimG4.sim_cff import geantservice, geantsim
from SimAlgos.digi_cff import digis

pythia8.preInitCommands = [
    'Beams:idA = 2212',     # beam 1 = proton
    'Beams:idB = 11',       # beam 2 = electron
    'Beams:frameType = 2',  # beams are back-to-back, but with different energies
    'Beams:eA = 7000.',     # proton energy (GeV)
    'Beams:eB = 50.',       # electron energy (GeV)
    'PDF:lepton2gamma = on',
    'PhotonCollision:gmgm2mumu = on',
]
genalg = GenAlg("Pythia8", SignalProvider = pythia8)
genalg.hepmc.Path = "hepmc"

geantsim.eventProvider = pythia8Particles

out = PodioOutput("out",  # PODIO output algorithm
    outputCommands = ["keep *"],
    filename = "output.root",
    OutputLevel = DEBUG,
)
podioevent = FCCDataSvc("EventDataSvc")

ApplicationMgr(  # wrap everything together
    TopAlg = [
        genalg,
        pythia8HepMCConverter,
        Geantsim,
        *digis,
        out
    ],
    EvtSel = 'NONE',
    EvtMax   = 100,
    ExtSvc = [podioevent, geoservice, geantservice]
)
```

Key4hep/Gaudi-specific

LHeC-specific (conditions/algos)

$\gamma\gamma \rightarrow \mu^+\mu^-$

Example – Pythia 8 event builder + visualisation tool



Symmetric geometry

Collections:

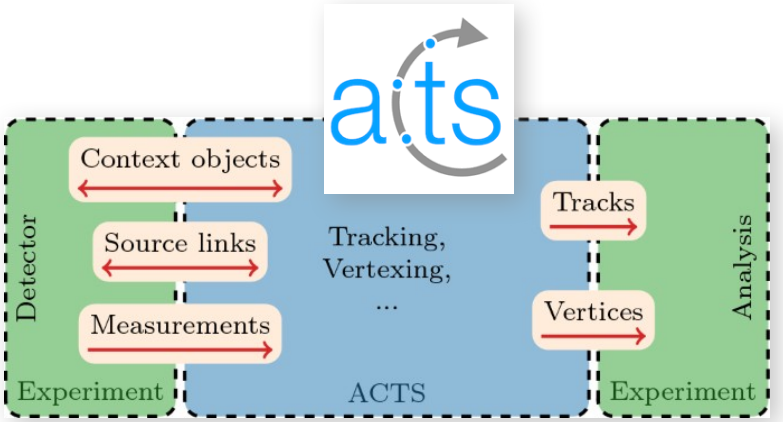
Name	ValueType	Size	ID
ecalBarrel1HitsDigitiser	edm4hep::CalorimeterHit	4	b5e7b7e8
ecalBarrel1HitsDigitiserHitsAssociation	edm4hep::MCRecoCaloAssociation	4	6722ac97
ecalBarrel1SimHits	edm4hep::SimCalorimeterHit	4	afaaab69
ecalBarrel2HitsDigitiser	edm4hep::CalorimeterHit	0	69687184
ecalBarrel2HitsDigitiserHitsAssociation	edm4hep::MCRecoCaloAssociation	0	37c43a01
ecalBarrel2SimHits	edm4hep::SimCalorimeterHit	0	feae9ddd
ecalBarrel3HitsDigitiser	edm4hep::CalorimeterHit	0	33a60594
ecalBarrel3HitsDigitiserHitsAssociation	edm4hep::MCRecoCaloAssociation	0	ac55eb44
ecalBarrel3SimHits	edm4hep::SimCalorimeterHit	0	960420a6
ecalPlug0SimHits	edm4hep::SimCalorimeterHit	0	a201fec3
ecalPlug1SimHits	edm4hep::SimCalorimeterHit	0	7f5ea194
GenParticles	edm4hep::MCParticle	1	0bcf5f90
hcalBarrelHitsDigitiser	edm4hep::CalorimeterHit	0	7ad12628
hcalBarrelHitsDigitiserHitsAssociation	edm4hep::MCRecoCaloAssociation	0	027344c1
hcalBarrelSimHits	edm4hep::SimCalorimeterHit	0	3de736f1
hcalEndcap0SimHits	edm4hep::SimCalorimeterHit	0	45180350
hcalEndcap1SimHits	edm4hep::SimCalorimeterHit	0	4b8cc742
hcalPlug0SimHits	edm4hep::SimCalorimeterHit	0	6a6792c0
hcalPlug1SimHits	edm4hep::SimCalorimeterHit	0	607891e0
muonBarrelSimHits	edm4hep::SimCalorimeterHit	0	186fc833
muonEndcap0iSimHits	edm4hep::SimCalorimeterHit	0	33564cba
muonEndcap02SimHits	edm4hep::SimCalorimeterHit	0	477b46b8
muonEndcap11SimHits	edm4hep::SimCalorimeterHit	0	c79ceeb4
muonEndcap12SimHits	edm4hep::SimCalorimeterHit	0	99f7c0ce
trackerBackwardSimHits	edm4hep::SimTrackerHit	0	5d0d13e5
trackerBarrelHits	edm4hep::TrackerHit	70	d48c7c18
trackerBarrelHitsHitsAssociation	edm4hep::MCRecoTrackerAssociation	70	776a5ab0
trackerBarrelSimHits	edm4hep::SimTrackerHit	70	7497b207
trackerForwardSimHits	edm4hep::SimTrackerHit	0	b8420509
trackerOuterBarrelHits	edm4hep::TrackerHit	3	6befe28e
trackerOuterBarrelHitsHitsAssociation	edm4hep::MCRecoTrackerAssociation	3	1c4fabd6
trackerOuterBarrelSimHits	edm4hep::SimTrackerHit	3	074e5ddb
vertexBarrelHits	edm4hep::TrackerHit	1	774b7463
vertexBarrelHitsHitsAssociation	edm4hep::MCRecoTrackerAssociation	1	54dee89c
vertexBarrelSimHits	edm4hep::SimTrackerHit	1	f99bb351
vertexOuterBarrelHits	edm4hep::TrackerHit	1	6ef691b2
vertexOuterBarrelHitsHitsAssociation	edm4hep::MCRecoTrackerAssociation	1	676c7039
vertexOuterBarrelSimHits	edm4hep::SimTrackerHit	1	eb52d74a

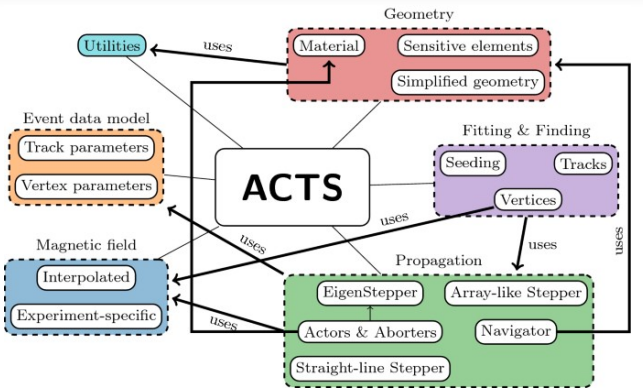
podio collections

Asymmetric geometry

Tracking & vertexing

- Investigation of **tracking/vertexing algorithms**, isolated one possible candidate: ATLAS' ACTS library
  - High-level, C++ ; natively supports DD4hep geometry definitions
  - Large collection of seeding/tracking/vertexing algorithms implementations, easy switch between various models/parameterisations
- Porting to Key4hep environment: key4hep/k4ActsTracking
  - Repository currently stalled, only DD4hep → ACTS geometry conversion recipe w/o reconstruction algorithms interfacing
  - ACTS linking efforts under the Key4hep/k4ActsTracking hood is underway ; manpower is (as usual) critical!
  - Currently working in parallel on private development branch: interfacing to track finding done, currently at track fitting+cleaning interfacing stage
  - ACTS workshop next week (https://indico.cern.ch/event/1397634/)





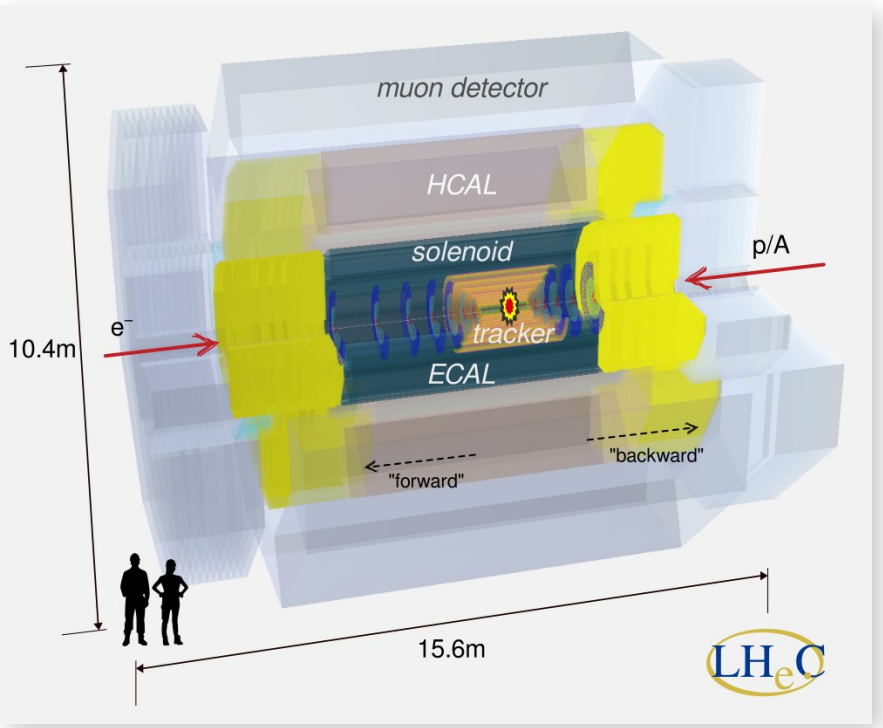
Software stack availability

- LHeCSW stack currently developed in a “private” environment on CERN/LXPLUS:
  - Relatively simple to build on any system including the standard **Key4hep** bundle: DD4hep, Geant4, Gaudi, EDM4hep
  - Natively provides some sourcing scripts to handle on e.g. LXPLUS7/8
  - Can still be used on a standard laptop (e.g. using a Docker container), although dependencies chain not so trivial to build on host machine

- **Versioning** of the current LHeC SW stack being considered, e.g. in the LHeC **CVMFS** area
  - `/cvmfs/lhec.cern.ch` to host all versions for eased maintainability/versioning of the full SW stack
  - Can rely on e.g. EP-SFT/LCG stacks for most of the dependencies (e.g. Geant4, Pythia 8, Sherpa, CepGen, …), with additional e-p-specific code maintained separately
  - Potentially w/ automation of library maintenance through continuous integration: Jenkins/GitLab CI (work in progress @ CERN GitLab: lforthom/lhecsw)

Outline

- LHeC offline software stack being prepared for e-p collider studies
  - One of the major requirements for the future physics case studies development
  - Heavily relies on Key4hep environment widely used in e.g. FCC-ee/CEPC case studies, allows to reuse some analysis code through a limited output collections names change (e.g. k4DataSource, FCCAnalyses, …)
  - Allows from basic I/O: from standard event generators, to podio ROOT output
  - Allows for modular (~cross-independent) development of various sub-detectors simulation/reconstruction tools



- More work to be done on many fronts:
  - MDI studies using BDSIM interfacing
  - more MC generators to be supported
  - more RawToDigi/DigiToRaw conversion modules to be handled
- Additional manpower is crucial for future parallel developments to be foreseen for physics case studies

Thanks for your attention!

Spares

BDSIM interfacing

BDSIM: Geant4 wrapper proposing a standardised definition of **common beamline elements** and their effect on beam dynamics (see tomorrow's talk on SR studies)

- LHeCSW provides a BDSIM interfacing tool ("TTree-reader") delivering a derivative of a trivial Geant4 particle gun

- Allows for visualisation/propagation of radiation synchrotron into DD4hep/Geant4 model
  - Particles transverse position at a given sampler translated into particles 4-momenta in the central detector, trivial event history (although can be recovered from Geant4-BDSIM history)
  - Effect of radiation can be studied on each individual subcomponent/material building up the detector geometry

- More developments can be done to directly interface BDSIM, might be overly complex for the current usage
  - Development branch currently being worked on: forthommel:ext-bdsim_direct_interface
  - Requiring minor adaptation from BDSIM output objects definition (avoiding in-between ROOT buffering stage through a collections/storage object with accessors)

Potential tasks/future developments

- Define a few **"standard candle"** resolution/efficiency **distributions** extracted in earlier attempts (e.g. CDR I/II)
  - In a first stage, can help validating the various approaches developed so far for each sub-detector
  - Can live in a (CI-oriented) test/relvals infrastructure, using a few plots of interest helping future developers in all forthcoming algos/data formats/conditions implementations
  - May lead to potential new interfacing of MCs/simulation toolboxes
  - Can possibly live inside the Key4hep environment… See e.g. the Key4hep validation website

- For fast simulation purposes, introduce some "conditions translators" between DD4hep-based geometry and **Delphes** detector simulation tool
  - E.g. a few resolution/acceptance extractors (→ Delphes TCL input) given a change of conditions/geometry scenario
  - Can possibly live inside the Key4hep environment too…