Simulation framework in the CEPCSW prototype

Ziyan Deng, Wenxing Fang, Chengdong Fu, Xingtao Huang, Gang Li, Weidong Li, Tao Lin, Manqi Ruan, Shengsen Sun, Jiaheng Zou

lintao@ihep.ac.cn

(on behalf of CEPCSW group)

CEPC workshop

IHEP

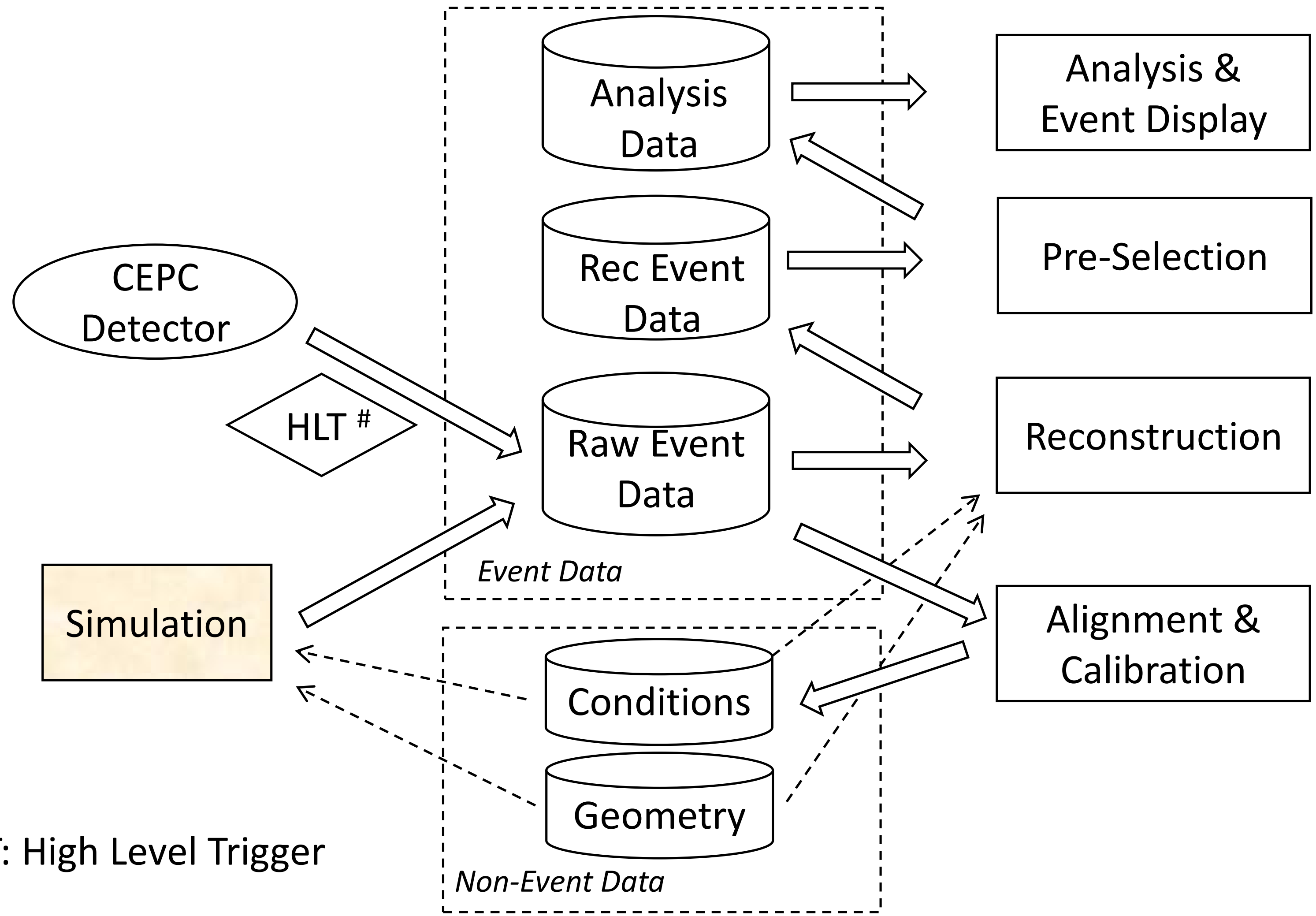
18-20 Nov. 2019

Outline

During R&D stage, CEPC seeks a lightweight & unified data processing system. A simulation framework is developed based on Gaudi and DD4hep.

- Motivation
- Design & implementation
  - Event Data Model
  - Geometry management
  - Physics generator interface
  - Detector simulation
- Software performance
- Summary and plans

Data processing chain



HLT: High Level Trigger

Motivation

CEPC data volume and computing challenges

		Estimated Data Volume
Short term	R&D: detector design	~PB/year
Long term	Higgs/W factory	1.5~3 PB/year
	Z factory	0.5~5 EB/year

Requirements on simulation:

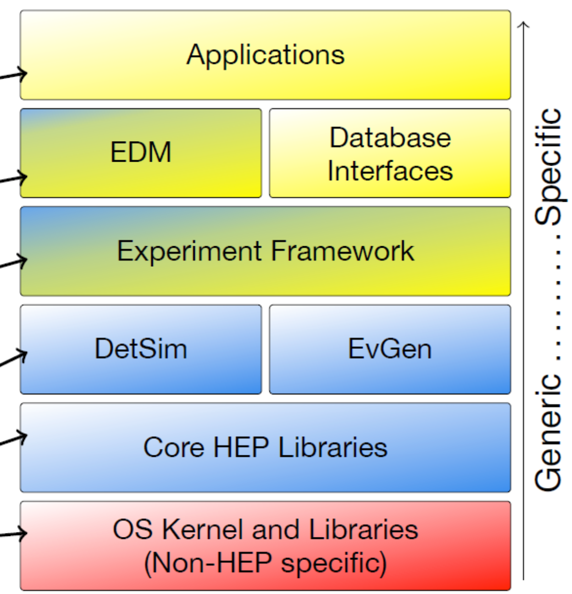
- R&D stage: fast development and iteration.
  - Detector performance studies.
  - geometry management & geometry update
- Operation: extensible, efficient and sustainable.
  - Modern software engineering, parallelism

KEY4HEP: Common software stack

A typical HEP Software Stack

Applications usually rely on large number of libraries, where some depend on others

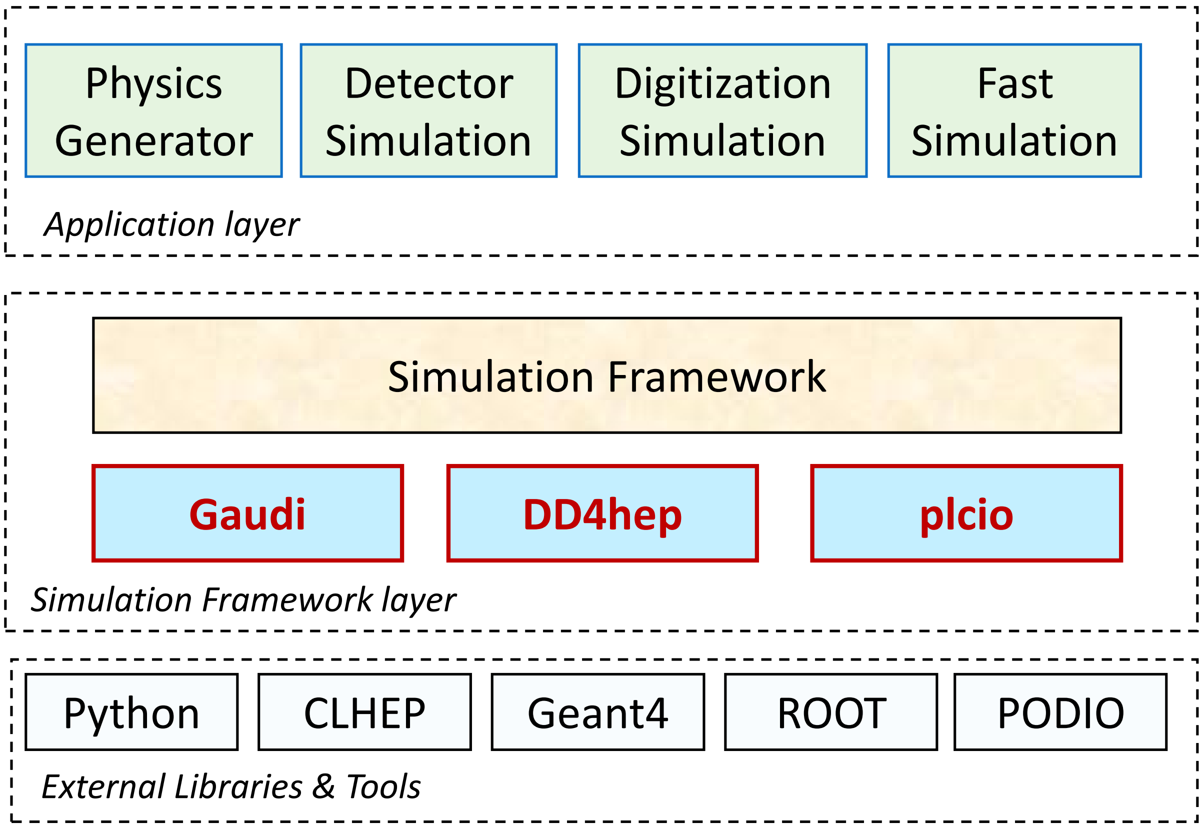
- Interfaces to tracking and reconstruction libraries (PandoraPFA, ACTS)
- (More or less) experiment specific event data model libraries
- Experiment core orchestration layer, which controls everything else: Marlin, Gaudi, CMSSW, AliRoot
- Packages used by many experiments: DD4hep, Pythia, . . .
- Usual core libraries (ROOT, Geant4, CLHEP, . . .)
- Non-HEP libraries: boost, python, cmake . . .



CHEP'19, Nov. 5, 2019 A. Sailer et al.: Towards a Turnkey Software Stack for HEP Experiments 3/16

Andre Sailer, CHEP2019, https://indico.cern.ch/event/773049/contributions/3474763/

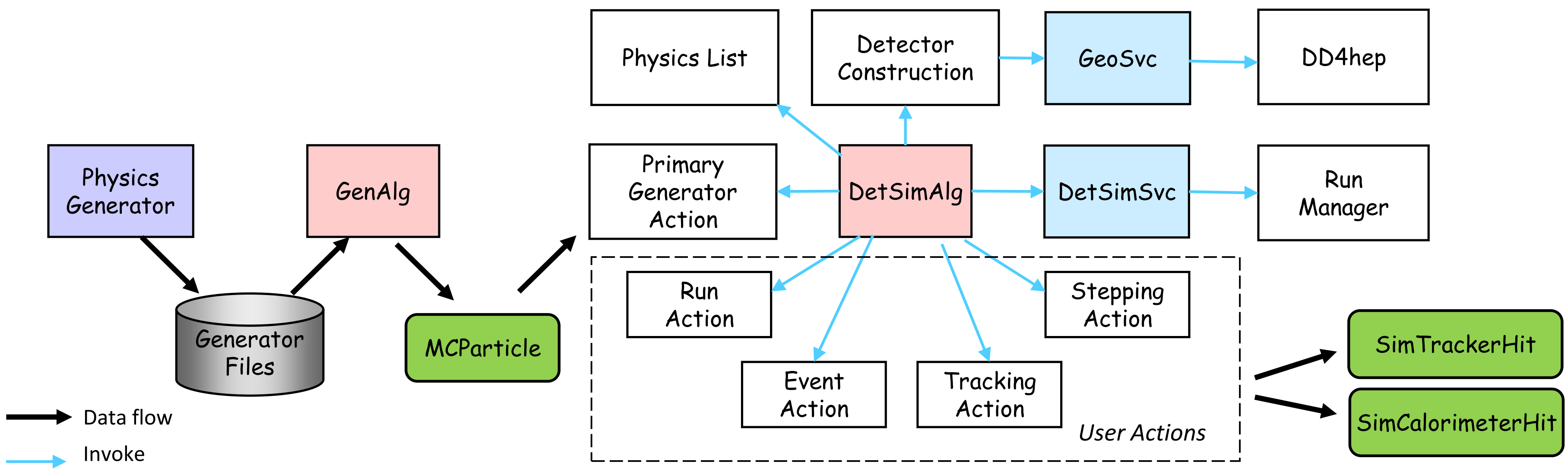
Simulation Software Stack



Simulation framework provides flexible integration of the applications (physics generator, detector simulation, digitization).

Design of simulation framework

- Based on Gaudi and DD4hep
- In the current prototype, “Tracker” is setup.



Core Packages: Generator, Simulation/DetSimInterface, Simulation/DetSimCore
Other packages: Detector, Simulation/DetSimGeom, Simulation/DetSimAna

Schematic view of Tracker

From CDR

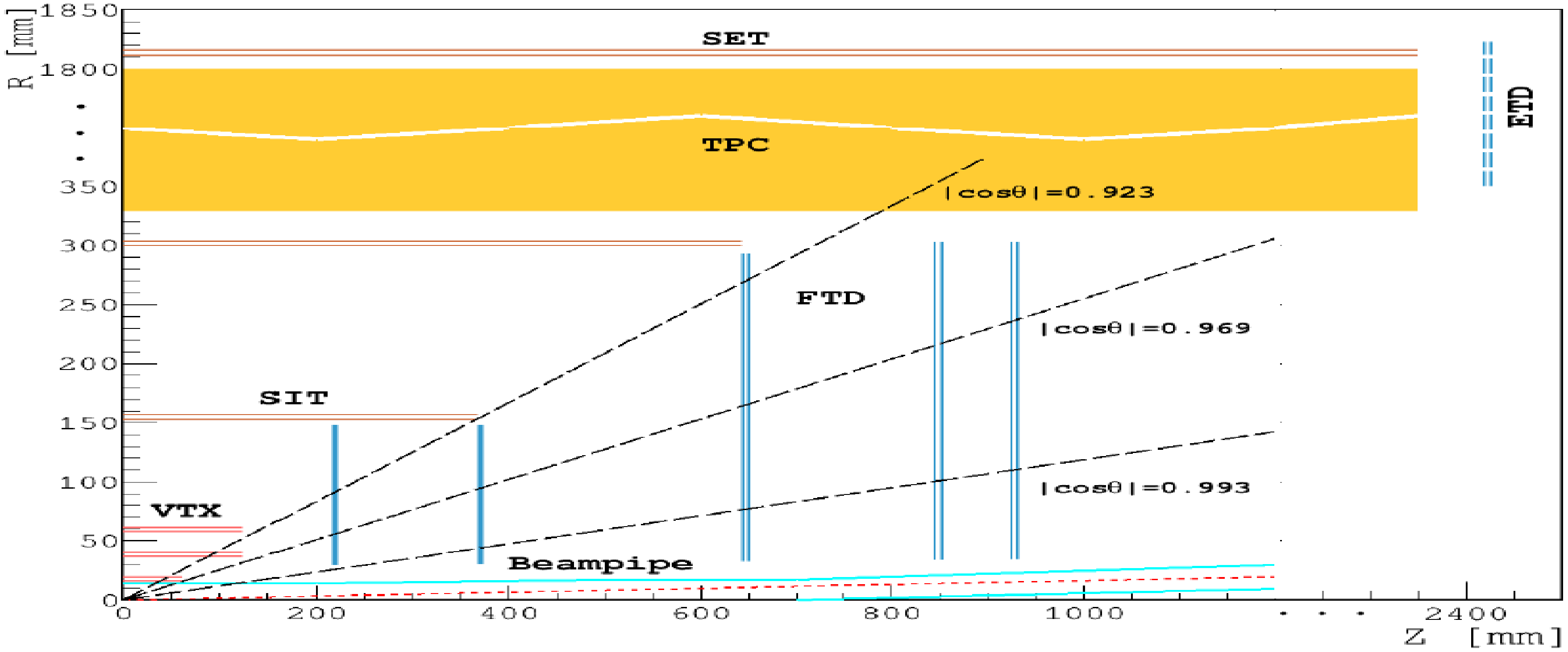
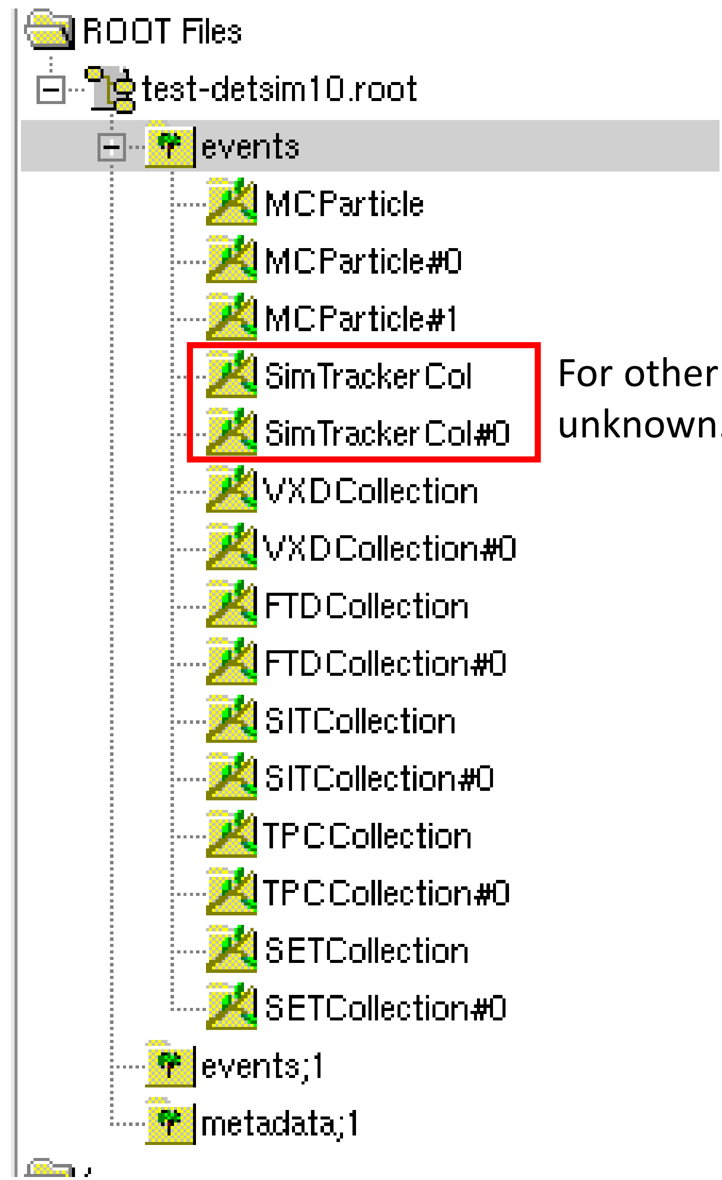


Figure 4.1: Preliminary layout of the tracking system of the CEPC baseline detector concept. The Time Projection Chamber (TPC) is embedded in a Silicon Tracker. Colored lines represent the positions of the silicon detector layers: red lines for the Vertex Detector (VTX) layers; orange lines for the Silicon Inner Tracker (SIT) and Silicon External Tracker (SET) components of the silicon tracker; gray-blue lines for the Forward Tracking Detector (FTD) and Endcap Tracking Detector (ETD) components of the silicon tracker. The cyan lines represent the beam pipe, and the dashed red line shows the beam line position with the beam crossing angle of 16.5 mrad. The ETD line is a dashed line because it is not currently in the full simulation. The radial dimension scale is broken above 350 mm for display convenience.

Event data model

- Based on plcio.
  - MCParticle
  - SimTrackerHit
  - SimCalorimeterHit
- Even though the data type is same, the collections are separate for different detectors.
- Collection names in output are kept as same as in Mokka:
  - MCParticle
  - VXDCollection
  - SITCollection
  - TPCCollection
  - SETCollection

} SimTrackerHit



For other unknown.

Geometry

- The DD4hep is used to construct the detector.
  - Based on Chengdong's version (CEPC workshop, Oxford, 2019).
- Package: Detector/DetCEPCv4
  - Geometry with VXD only: CepC_v4-onlyVXD.xml (Default)
  - Geometry with Tracker: CepC_v4-onlyTracker.xml
  - VXD: compact/vxd07.xml + src/tracker/VXD04_geo.cpp
- GeoSvc provides the interface to access the DD4hep.
  - `dd4hep::Detector* lcdd();`
- In detector simulation, a DetElemTool is used to convert the DD4hep geometry to Geant4 geometry.
  - Package: Simulation/DetSimGeom/
  - This design allows the traditional Geant4 geometry construction.

Identifier

- ID for each detector component.
- Keep compatible between Mokka and DD4hep.

```
<readouts>
  <readout name="VXDCollection">
    <id>system:5,side:-2,layer:9,module:8,sensor:8</id>
  </readout>
</readouts>
```

- CellID0 and CellID1: 32bit for each
- The bitmap starts from low to high

1110 000000101 11 00001 high <- low
layer(9) system(5)
Module(8) side(-2: sign value)

```
const unsigned ILDCellID0::subdet = 0 ;
const unsigned ILDCellID0::side   = 1 ;
const unsigned ILDCellID0::layer  = 2 ;
const unsigned ILDCellID0::module = 3 ;
const unsigned ILDCellID0::sensor = 4 ;

const int ILDDetID::NOTUSED    =  0 ;
const int ILDDetID::VXD        =  1 ;
const int ILDDetID::SIT        =  2 ;
const int ILDDetID::FTD        =  3 ;
const int ILDDetID::TPC        =  4 ;
const int ILDDetID::SET        =  5 ;
const int ILDDetID::ETD        =  6 ;
const int ILDDetID::ECAL       = 20 ;
const int ILDDetID::ECAL_PLUG  = 21 ;
const int ILDDetID::HCAL       = 22 ;
const int ILDDetID::HCAL_RING  = 23 ;
const int ILDDetID::LCAL       = 24 ;
const int ILDDetID::BCAL       = 25 ;
const int ILDDetID::LHCAL      = 26 ;
const int ILDDetID::YOKE       = 27 ;
const int ILDDetID::COIL        = 28 ;
const int ILDDetID::ECAL_ENDCAP= 29 ;
const int ILDDetID::HCAL_ENDCAP= 30 ;
const int ILDDetID::YOKE_ENDCAP= 31 ;
```

LCIO/src/cpp/src/UTIL/ILDConf.cc

Physics generator interface

- Implemented as a major algorithm with multiple small GenTools.
  - The major algorithm creates the event object and passes it to GenTools.
  - A GenTool then adds or modifies the MC particle in the event.
- Several GenTools are ready for use.
  - GtGunTool: a particle gun, supporting multiple particles.
  - StdHepRdr: reads StdHep format data.
  - SLCIORdr: reads LCIO format data.
- Configure in the job option file.
  - GtGunTool: Particle name/PDGcode, momentum, direction
  - Readers: Input

Physics list

- Default physics list: “QGSP_BERT”
  - Same as Mokka (Control::PhysicsListName = "QGSP_BERT")
- Can be changed easily by setting
  - detsimalg. PhysicsList = “QGSP_BERT_HP”;
- Using Geant4’s G4PhysListFactory to instance the real physics list.

```
nlists_hadr = 23;
G4String ss[23] = {
  "FTFP_BERT","FTFP_BERT_TRV","FTFP_BERT_ATL","FTFP_BERT_HP","FTFQGSP_BERT",
  "FTFP_INCLXX","FTFP_INCLXX_HP","FTF_BIC", "LBE","QBBC",
  "QGSP_BERT","QGSP_BERT_HP","QGSP_BIC","QGSP_BIC_HP","QGSP_BIC_AllHP",
  "QGSP_FTFP_BERT","QGSP_INCLXX","QGSP_INCLXX_HP","QGS_BIC",
  "Shielding","ShieldingLEND","ShieldingM","NuBeam"};
```

```
nlists_em = 12;
G4String s1[12] = {"","_EMV","_EMX","_EMY","_EMZ","_LIV","_PEN",
                   "__GS","__SS","_EM0","_WVI","__LE"};
```

Sensitive detector

- The DDG4's sensitive detectors are registered automatically.
  - Done in the IDetElemTool::ConstructSDandField.
  - See: Simulation/DetSimGeom/src/AnExampleDetElemTool.cpp
- The volumes marked as sensitive are associated with SD using the "type" name in the compact file.
  - Supporting two types: tracker and calorimeter.

Compact:

```
<detector name="VXD" type="VXD04" vis="VXDVis" id="ILDDetID_VXD"
limits="Tracker_limits" readout="VXDCollection" insideTrackingVolume="true">
```

Logs:

```
Compact        INFO  ++ Converted subdetector:VXD of type VXD04 [tracker]
```

```
ToolSvc.AnExamp...   INFO Type/Name: tracker/VXD
ToolSvc.AnExamp...   INFO  -> Adding SiActiveLayer_00
ToolSvc.AnExamp...   INFO  -> Adding SiActiveLayer_01
ToolSvc.AnExamp...   INFO  -> Adding SiActiveLayer_02
ToolSvc.AnExamp...   INFO  -> Adding SiActiveLayer_03
ToolSvc.AnExamp...   INFO  -> Adding SiActiveLayer_04
ToolSvc.AnExamp...   INFO  -> Adding SiActiveLayer_05
```

Associate tracker SD and SiActiveLayer_XXX.

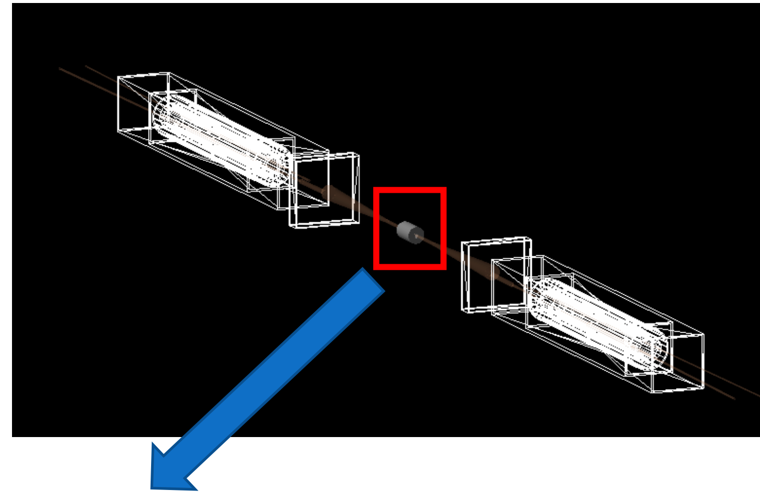
User actions

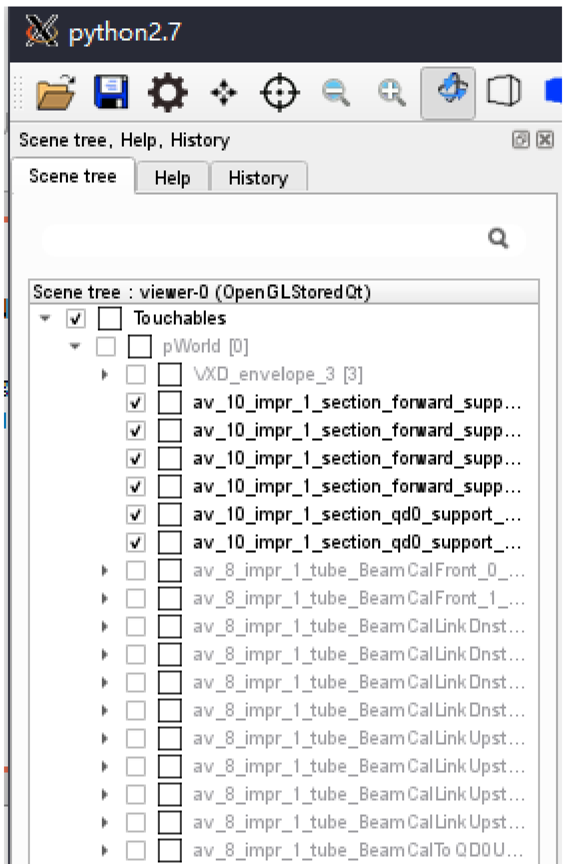
- Implemented as IAnaElemTool
  - Path: Simulation/DetSimAna
  - The registered tools will be invoked by the Geant4 user actions automatically (Run, Event, Tracking, Stepping Actions).
- ExampleAnaElemTool
  - An example to convert the Geant4's hit objects to plcio's hit objects.
  - All the necessary collections are defined here.

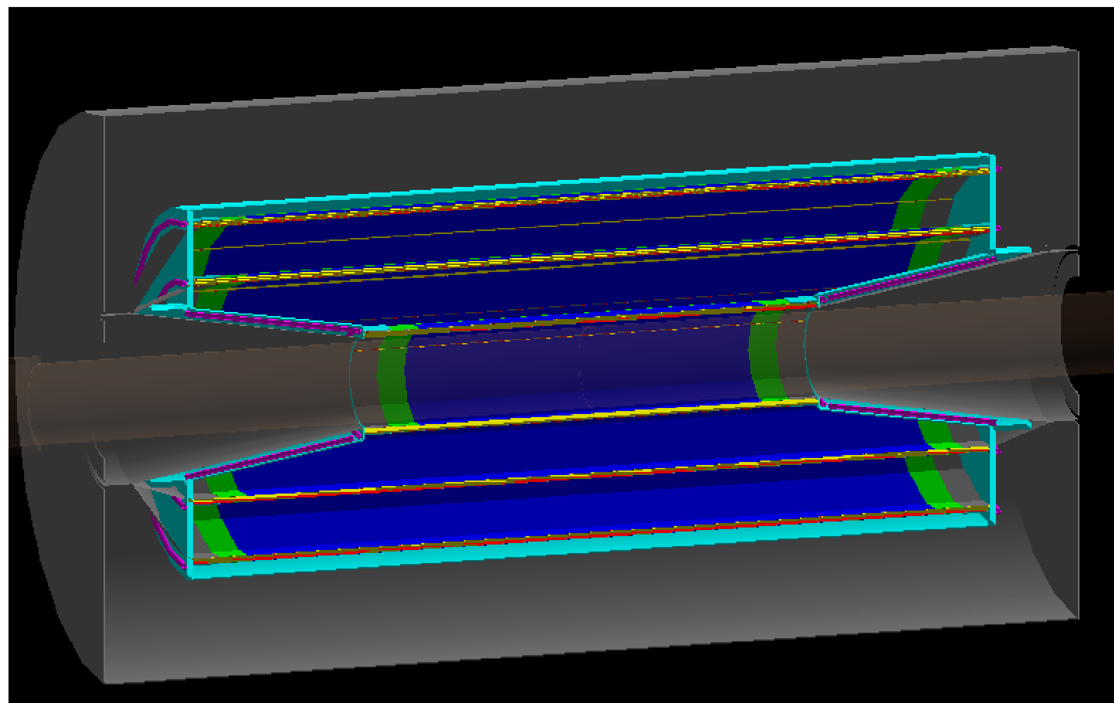
```
DataHandle<plcio::SimTrackerHitCollection> m_trackerCol{"SimTrackerCol",
        Gaudi::DataHandle::Writer, this};
DataHandle<plcio::SimTrackerHitCollection> m_VXDCol{"VXDCollection",
        Gaudi::DataHandle::Writer, this};
DataHandle<plcio::SimTrackerHitCollection> m_FTDCol{"FTDCollection",
        Gaudi::DataHandle::Writer, this};
DataHandle<plcio::SimTrackerHitCollection> m_SITCol{"SITCollection",
        Gaudi::DataHandle::Writer, this};
DataHandle<plcio::SimTrackerHitCollection> m_TPCCol{"TPCCollection",
        Gaudi::DataHandle::Writer, this};
DataHandle<plcio::SimTrackerHitCollection> m_SETCol{"SETCollection",
        Gaudi::DataHandle::Writer, this};
```

Visualization

- Using Geant4’s Qt-based UI.
- Need to enable vis in DetSimAlg.







Job option

- Job option is a python script.
- Users need to copy the standard one.
- Use gaudirun.py to run it. (It can be improved in the future.)

```
./run gaudirun.py '$EXAMPLESROOT/options/tut_detsim.py'
```

```
from Gaudi.Configuration import *

##############################################################################
# Random Number Svc
##############################################################################
from Configurables import RndmGenSvc, HepRndm__Engine_CLHEP__RanluxEngine_

# rndmengine = HepRndm__Engine_CLHEP__RanluxEngine_() # The default engine in Gaudi
rndmengine = HepRndm__Engine_CLHEP__HepJamesRandom_() # The default engine in Geant4
rndmengine.SetSingleton = True
rndmengine.Seeds = [42]   <= Random number seed

# rndmgensvc = RndmGenSvc("RndmGenSvc")
# rndmgensvc.Engine = rndmengine.name()
```

Job option (Event Data & GeoSvc)

```
##############################################################################
# Event Data Svc
##############################################################################
from Configurables import CEPCDataSvc
dsvc = CEPCDataSvc("EventDataSvc")


##############################################################################
# Geometry Svc
##############################################################################

# geometry_option = "CepC_v4-onlyTracker.xml"
geometry_option = "CepC_v4-onlyVXD.xml"

if not os.getenv("DETCEPCV4ROOT"):
    print("Can't find the geometry. Please setup envvar DETCEPCV4ROOT." )
    sys.exit(-1)

geometry_path = os.path.join(os.getenv("DETCEPCV4ROOT"), "compact", geometry_option)
if not os.path.exists(geometry_path):
    print("Can't find the compact geometry file: %s"%geometry_path)
    sys.exit(-1)

from Configurables import GeoSvc
geosvc = GeoSvc("GeoSvc")
geosvc.compact = geometry_path
```

<= DD4hep XML file

The geometry file under DetCEPCv4 is loaded.

Job option (Physics Generator)

```
##############################################################################
# Physics Generator
##############################################################################
from Configurables import GenAlgo
from Configurables import GtGunTool
from Configurables import StdHepRdr
from Configurables import SLCIORdr
from Configurables import HepMCRdr
from Configurables import GenPrinter

# gun = GtGunTool("GtGunTool")
# gun.Particles = ["pi+"]
# gun.Energies = [100.] # GeV

# gun.ThetaMins = [] # rad; 45deg
# gun.ThetaMaxs = [] # rad; 45deg

# gun.PhiMins = [] # rad; 0deg
# gun.PhiMaxs = [] # rad; 360deg

stdheprdr = StdHepRdr("StdHepRdr")
stdheprdr.Input = "/cefs/data/stdhep/CEPC250/2fermions/E250.Pbhabha.e0.p0.whizar

# lciordr = SLCIORdr("SLCIORdr")
# lciordr.Input = "/cefs/data/stdhep/lcio250/signal/Higgs/E250.Pbbh.whizard195/E

# hepmcrdr = HepMCRdr("HepMCRdr")
# hepmcrdr.Input = "example_UsingIterators.txt"

genprinter = GenPrinter("GenPrinter")

genalg = GenAlgo("GenAlgo")
# genalg.GenTools = ["GtGunTool"]
genalg.GenTools = ["StdHepRdr"]
# genalg.GenTools = ["StdHepRdr", "GenPrinter"]
# genalg.GenTools = ["SLCIORdr", "GenPrinter"]
# genalg.GenTools = ["HepMCRdr", "GenPrinter"]
```

<= Particle Gun

<= StdHep Reader

<= LCIO Reader

<= Enable the GenTools

Job option (Detector simulation)

```
##############################################################################
# Detector Simulation
##############################################################################
from Configurables import DetSimSvc

detsimsvc = DetSimSvc("DetSimSvc")

# from Configurables import ExampleAnaElemTool
# example_anatool = ExampleAnaElemTool("ExampleAnaElemTool")

from Configurables import DetSimAlg

detsimalg = DetSimAlg("DetSimAlg")

# detsimalg.VisMacs = ["vis.mac"]                      <= If need visualization, uncomment it

detsimalg.RunCmds = [                                  <= Part of Geant4 macros are support
#    "/tracking/verbose 1",
]
detsimalg.AnaElems = [
    # example_anatool.name()                           <= User actions
    "ExampleAnaElemTool"
]
detsimalg.RootDetElem = "WorldDetElemTool"
                                                       <= Geometry
from Configurables import AnExampleDetElemTool
example_dettool = AnExampleDetElemTool("AnExampleDetElemTool")
```

Job option (I/O & AppMgr)

```
################################################################################
# POD I/O
################################################################################
from Configurables import PodioOutput
out = PodioOutput("outputalg")
out.filename = "test-detsim10.root"
out.outputCommands = ["keep *"]

################################################################################
# ApplicationMgr
################################################################################

from Configurables import ApplicationMgr
ApplicationMgr( TopAlg = [genalg, detsimalg, out],
                EvtSel = 'NONE',
                EvtMax = 10,
                ExtSvc = [rndmengine, dsvc, geosvc],
)
```

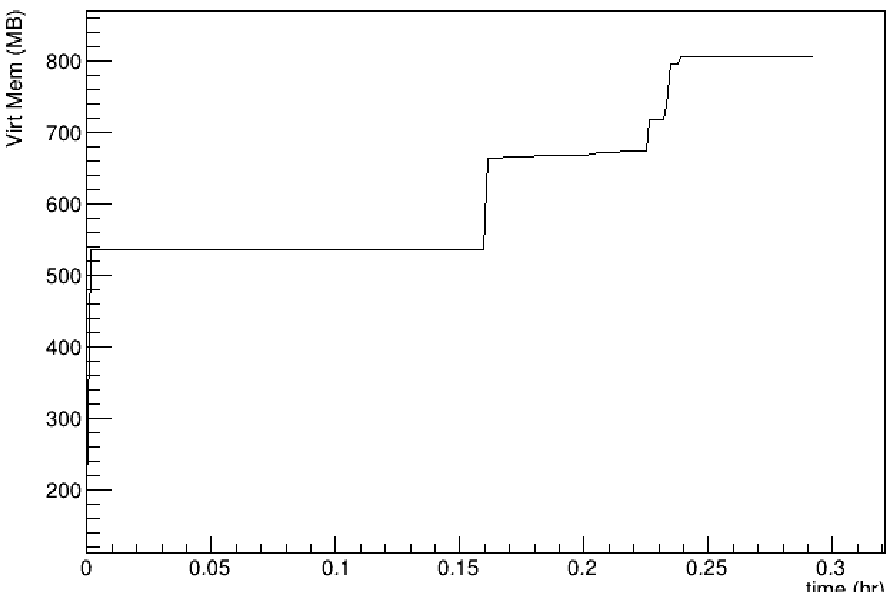
<= Output file name

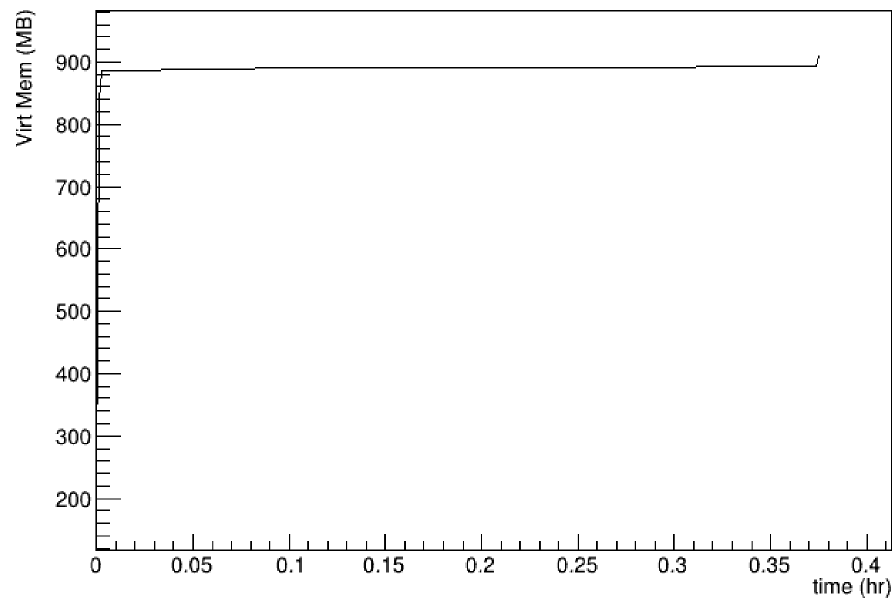
<= Total events

Software performance

- Measure two cases
  - Physics generator only
  - Physics generator + detector simulation

CPU: Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz
Memory: 128 GB





Input file:
CEPC250/2fermions/E250.Pbhabha.e0.p0.whizard195/bhabha.e0.p0.00001.stdhep
Total events: 200,000

Geometry： VXD only
Total events: 1000

Summary and plans

- A simulation framework prototype is developed.
  - Integration: based on Gaudi and DD4hep.
  - EDM&ROOTIO: PODIO and plcio
  - Geometry: DD4hep. “Tracker” is ready in the simulation framework.
  - Physics generator: GenTool based. Particle gun and StdHep reader.
  - Transportation: Geant4. User actions are implemented as tools.

Sub-components	Plans
Generators	Generators on the fly
Event Data Model	MCTruth correlation
Geometries & fields	TPC, calo, magnetic field, different options
Fast simulation	Integration
Validation & Production	stress testing, performance testing, MC data challenges etc.
Parallelism	Gaudi+Geant4 10

Git Repo: http://cepcgit.ihep.ac.cn/cepc-prototype/CEPCSW

Thank you!