DDSim a simulation package based on DD4hep

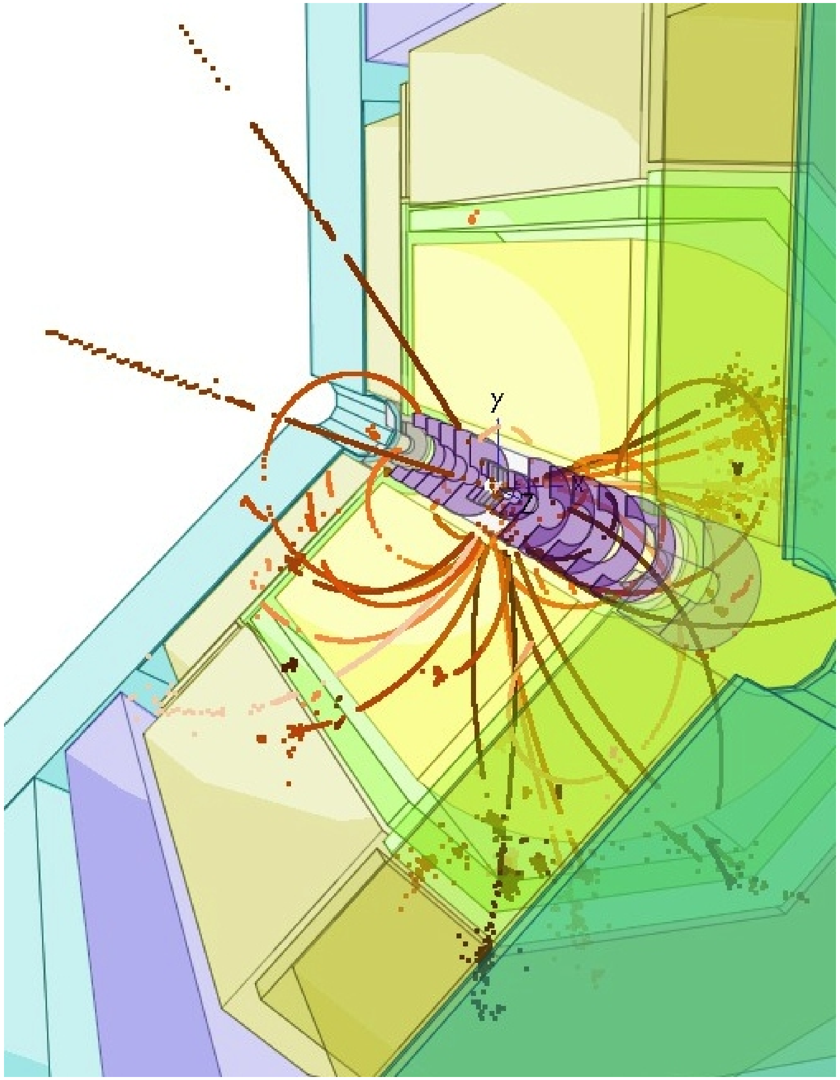
Frank Gaede, CERN/DESY

LCWS 2014, Belgrade, Serbia
5-10 October 2014

Outline

- Introduction
- DD4hep
- DDG4
- DDRec
- DDSim
- ILD and CLIC simulation models
- Summary/Outlook

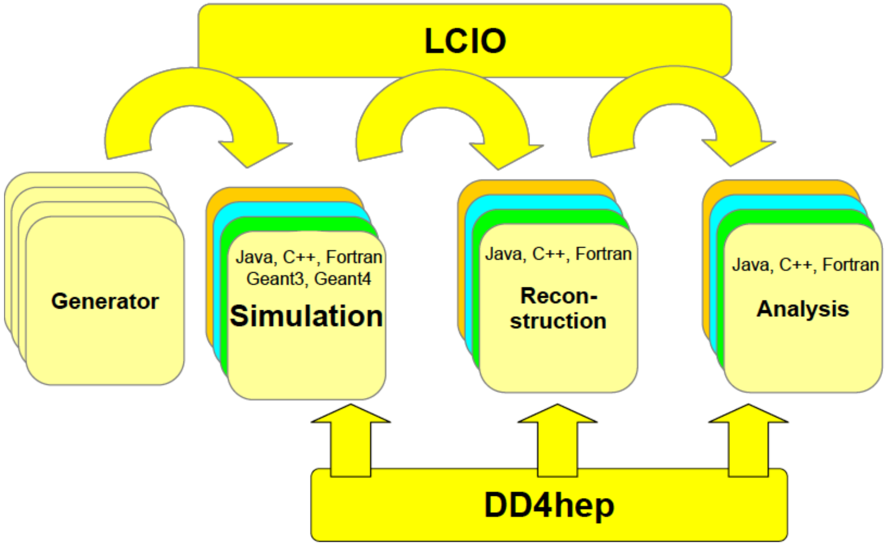
work (and material) by:
M.Frank, C.Grefe, A.Sailer, N.Nikiforou, S.Lu, M.Petric, F.G.



Frank Gaede, LCWS14 Belgrade, Oct 5-10, 2014

Introduction

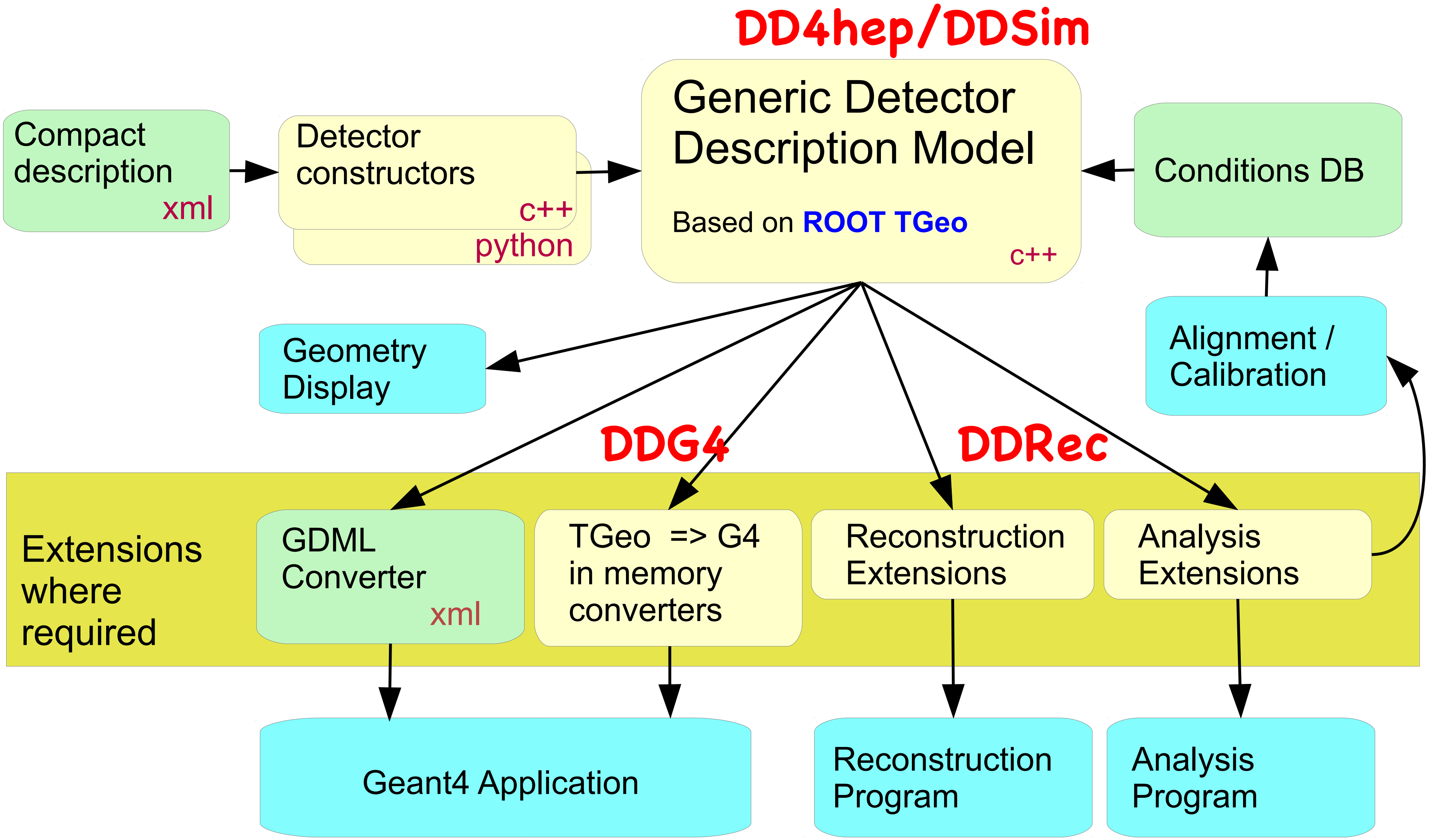
- in Linear Collider Software Meetings 2012/2013 decided to use new detector geometry description **DD4hep** as basis for a new common LC simulation package



- **DD4hep:** common detector geometry description
  - developed in AIDA WP2 (CERN, DESY)
- will be used by CLICdp, FCC and ILD,...

- defining a common geometry API is the second step - after the common EDM: LCIO - that is needed to have an open and modular software framework

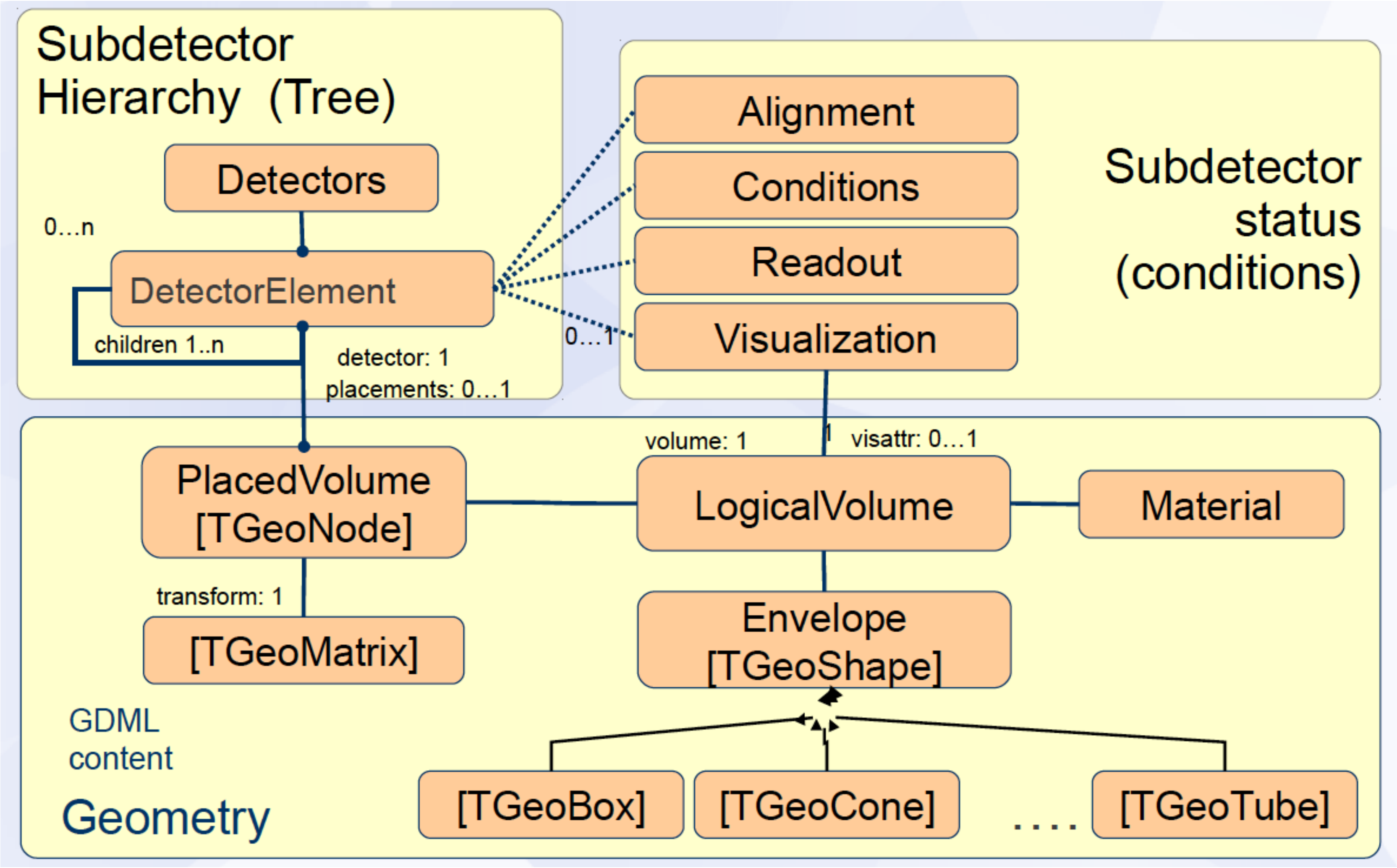
DD4hep: schematic overview



one source of geometry to feed into all HEP applications

Frank Gaede, LCWS14 Belgrade, Oct 5-10, 2014

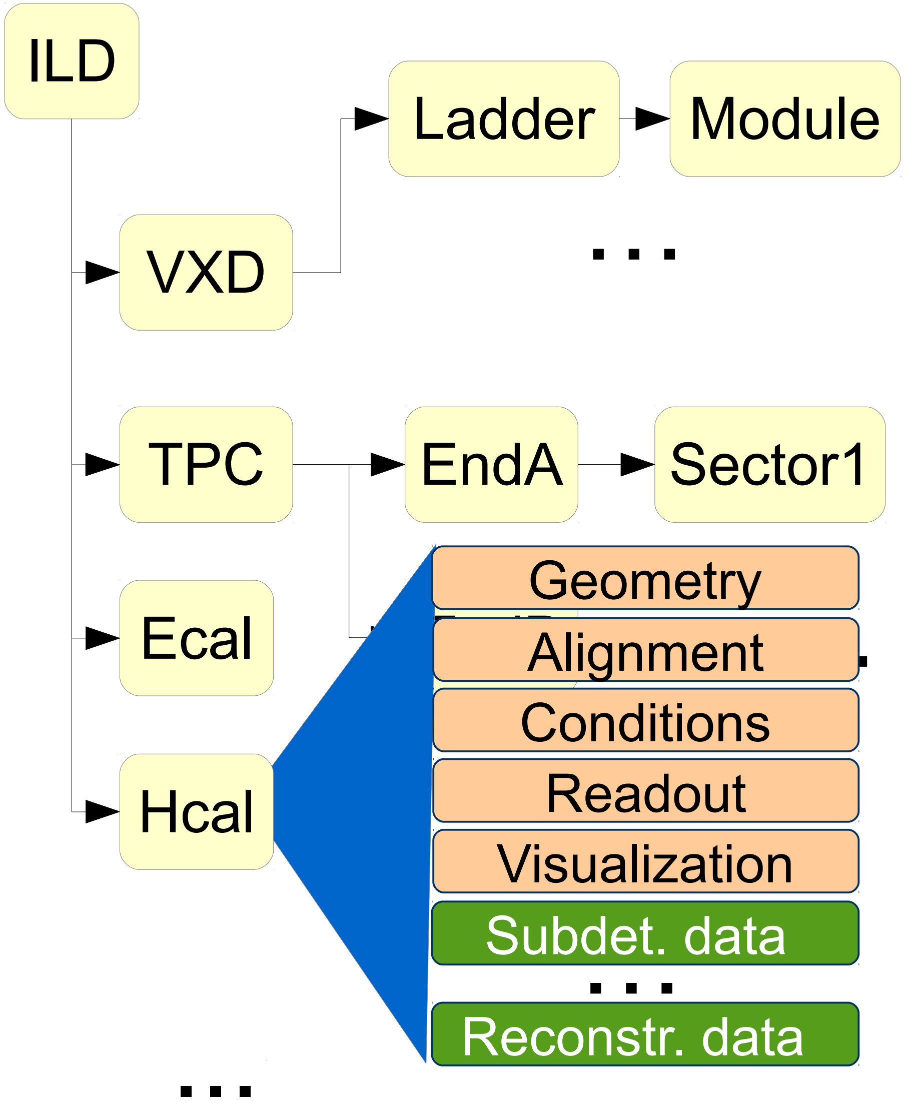
DD4hep geometry implementation



- DD4hep uses the TGeo geometry classes to instantiate geometry tree -> can use all TGeo features directly
- additional (user) code added to detector element class

DD4hep: detector elements

- detector is described in a tree-like hierarchy of DetectorElements:
  - sub detectors or parts thereof:
    - modules, sensors, …
  - DetectorElement describes:
    - Geometry
      - Shape, material, …
    - detector properties:
      - readout, alignment, conditions
    - visualization
- allows additional user/ experiment specific data

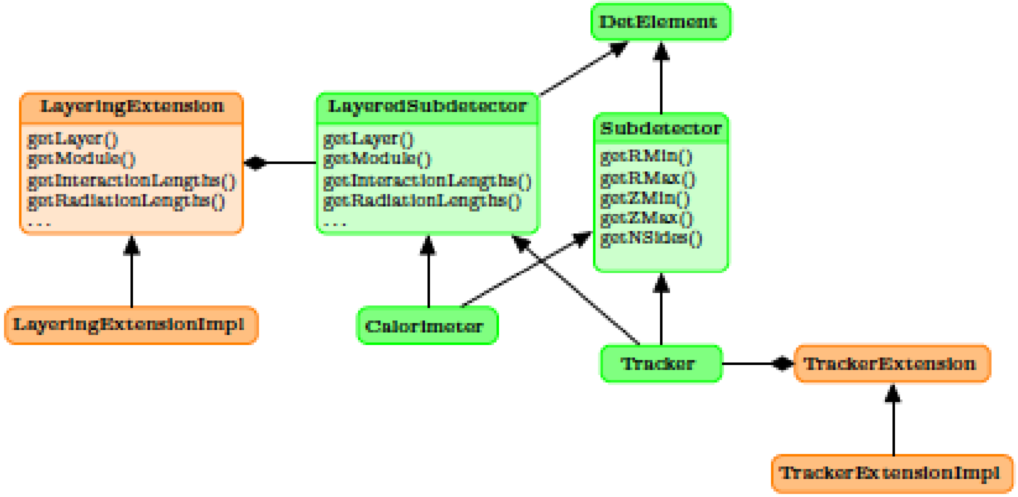


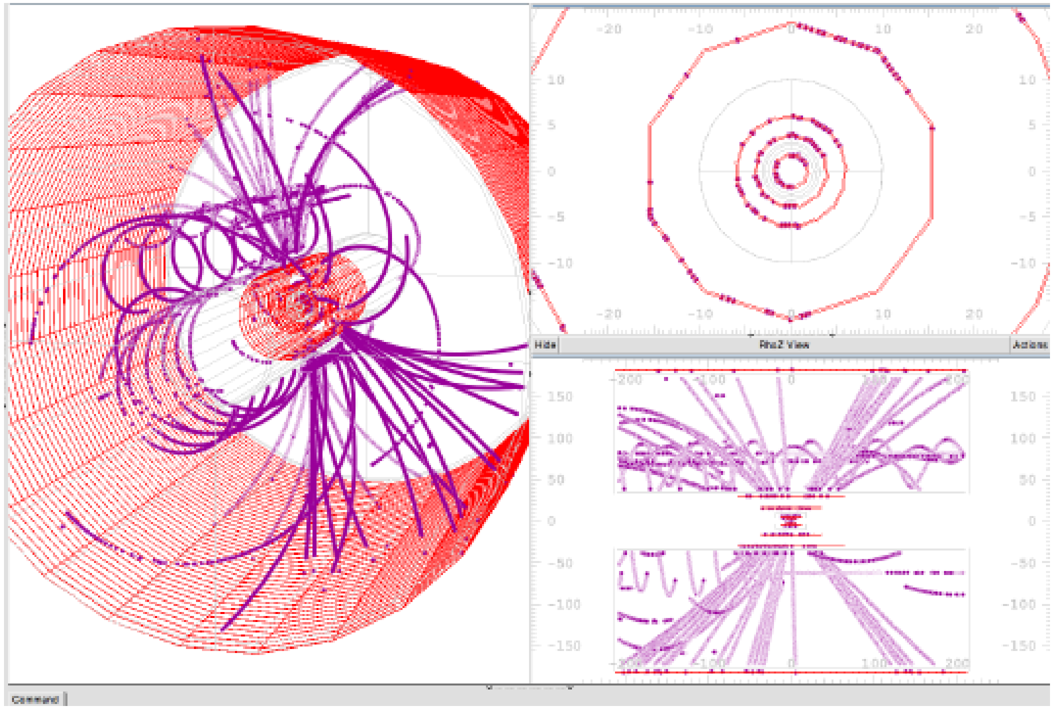
DDG4 – built in Geant4 gateway

- in memory conversion of TGeo geometry to Geant4 geometry
- modular design using plugin mechanism for
  - sensitive detectors, Geant4 user actions : stepping, tracking,…
  - input (generator files) and output (LCIO,…)
- configure mechanism with xml, python or CINT:
  - physics lists, limits, fields,…
  - define sequences for
    - input, sensitive detectors, user actions, output,…
- features :
  - full flexibility in sensitive detectors
  - can use extension code in simulation and reconstruction
  - supported by CERN for FCC and CLICdp

DDRec - interface to reconstruction

- extension mechanism is used to define interface for reconstruction
- calorimeters and trackers defined as LayeredSubdetectors
  - use to eventually replace GEAR
  - work in progress ...
- for tracking additional Surfaces provide:
  - u,v,normal and origin
  - inner and outer (averaged) material incl. thickness
- -> planes and cylinders allow for simple navigation in detector geometry for the tracking





DDRec versus GEAR

- need to be able to run existing reconstruction code MarlinReco, MarlinTrk, PandaraPFA and LCFIPlus with new simulation model
- completely based on GEAR
- -> need for (intermediate) interface from DDRec to GEAR

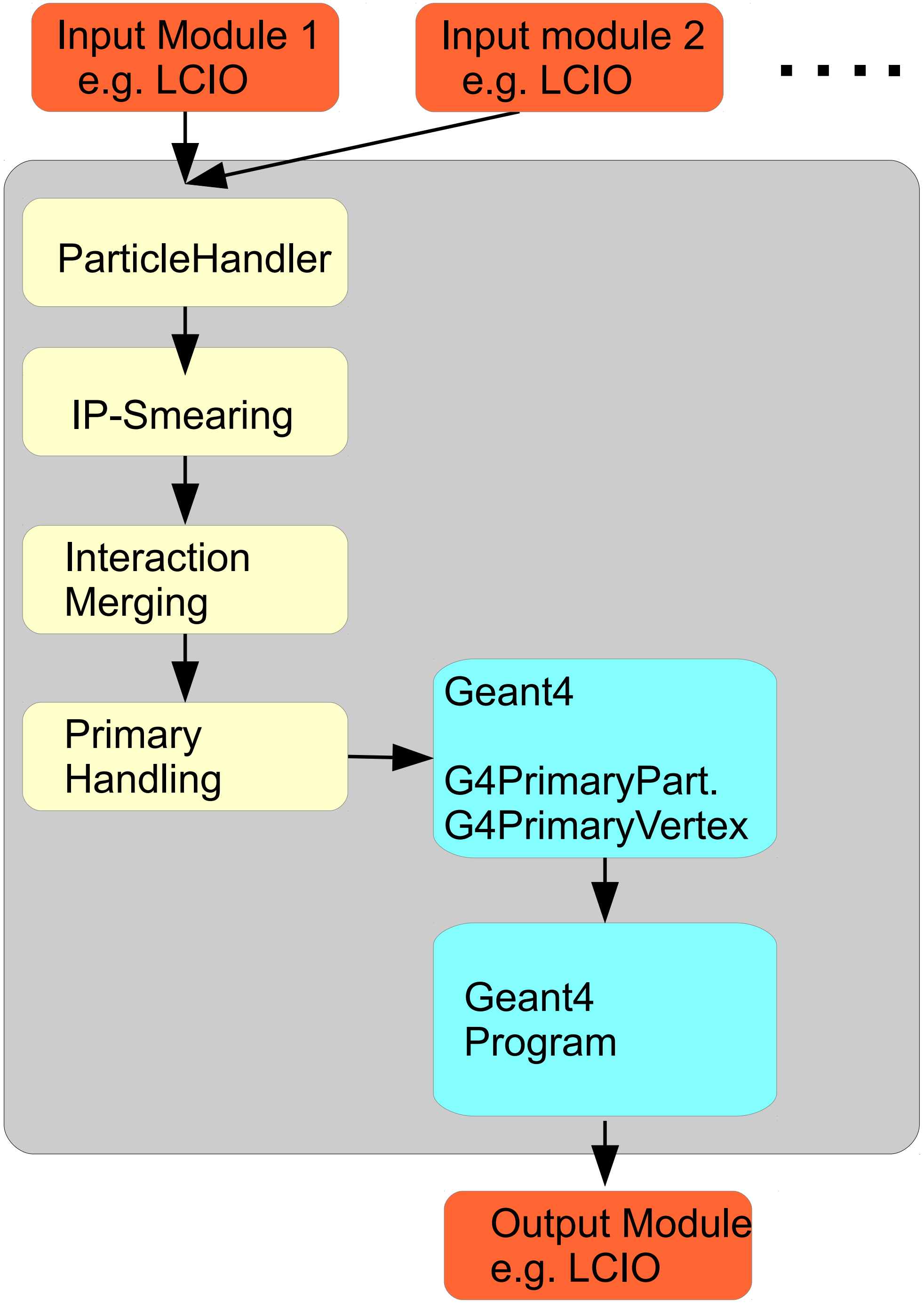
- current thinking:
- introduce light weight classes that hold relevant data and fill them on detector construction
- write 'quick and dirty' throw away code to generate GEAR objects from DD4hep detector model
- test and run existing reconstruction
- further develop new DDRec classes and eventually
- replace GEAR w/ DDRec in Marlin based reconstruction

recent developments in DD4hep

- implemented data classes (EDM) for MC truth and sim tracker/calorimeter hits analogue to LCIO classes:
  - MCParticle, SimTrackerHit, SimCalorimeterHit
- provide input and output modules for LCIO
  - others are possibly, e.g. ROOT I/O
- this allows DD4hep to:
  - be independent of LCIO for non LC communities
  - implement logic such as MC-Truth link only once, independent of the data model that is used to write the hits
- implemented MC-Truth linking algorithm
- implemented 'canonical' sensitive detectors for trackers and calorimeters that work with LCIO SimTracker/CalorimeterHits and "arbitrary" segmentations

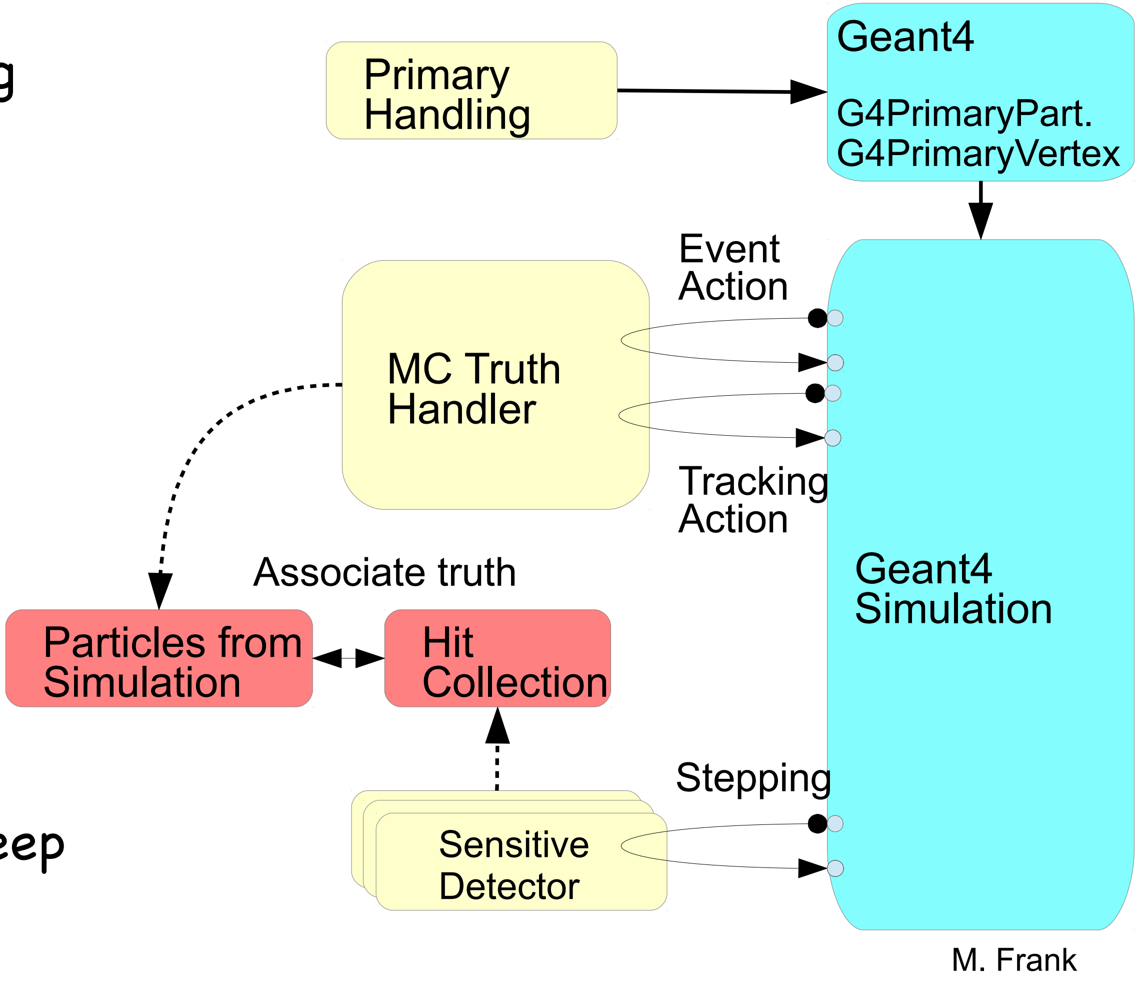
DD4hep input/output modules

- new modules for treatment of MC truth particles
  - IP-Smearing
  - merging (overlay)
  - MC-truth linking
- logic can be reused for any type of (compatible) input format:
  - LCIO (done)
  - stdhep (done)
  - HepEvt (ongoing)
  - HepMC (not yet)
  - ...



MC-Truth handling

- keep track of particles depositing energy in stepping action
- add new particles as new particles are created by geant4:
  - delta electrons
  - EM & nuclear interactions
  - decays
- decide at end of tracking actions which particles to keep and persist in output file (MCParticle collection)
- assign hits to the particles that are kept



MC-truth link logic

- new particles, created in simulation are added to the MCParticle list if:
  - they are created (decay, interaction,...) inside the tracking volume
  - and their energy E > E_cut
    - e.g. low energy delta electron are not created, but Bremsstrahlung photons are
    - decays in flight (K0_s) are always created
- shower particles are not created, hits are assigned to parent particle (the one entering the calorimeter)

  exceptions:

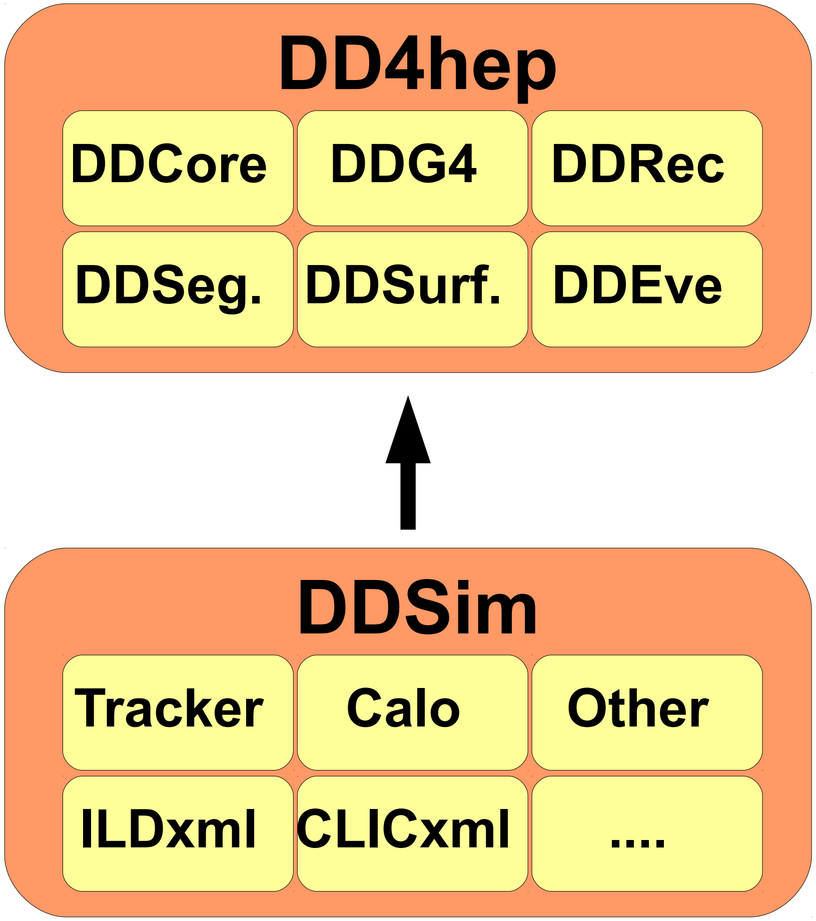
  - particles scattered back into the tracking volume
  - default shower mode is activated for dedicated studies
- algorithm very similar to the one implemented in Mokka (however implementation isn't)

DDSim simulation package

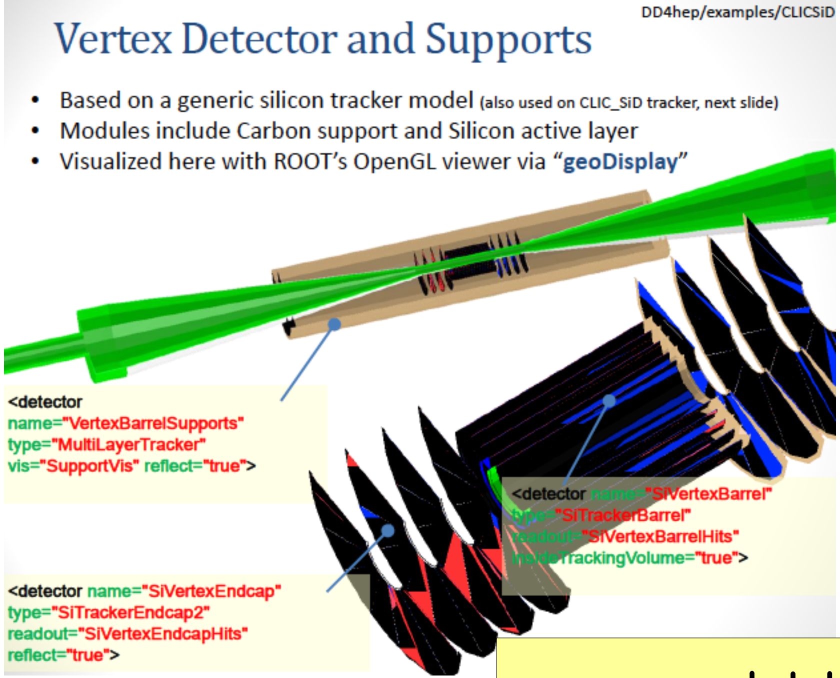
- created package DDSim as a common LC simulation package for ILD and CLIC (and SiD)
- eventually want to preserve all current Mokka models - started with ILD_o1_v05:
  - extract DB params to xml and
  - line-by-line port of geometry drivers
- will soon add CLIC models from DD4hep/examples
- for details and code:

https://svnsrv.desy.de/viewvc/ddsim/DDSim/

- NB: code and structure still somewhat experimental and targeted at sw-experts



CLIC detector model in DDSim



DD4hep/examples/CLICSiD

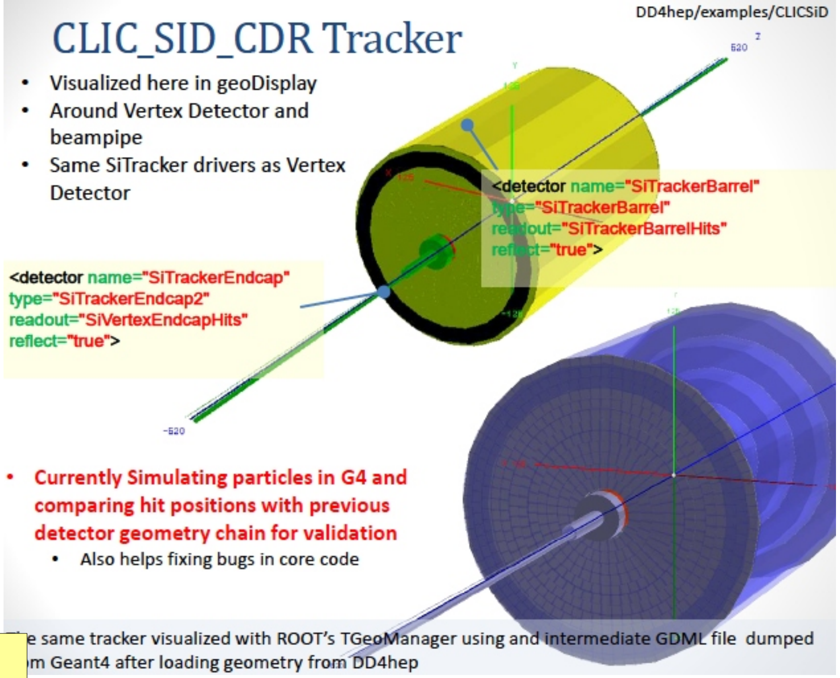
Vertex Detector and Supports

- Based on a generic silicon tracker model (also used on CLIC_SiD tracker, next slide)
- Modules include Carbon support and Silicon active layer
- Visualized here with ROOT's OpenGL viewer via "**geoDisplay**"

<detector name="VertexBarrelSupports" type="MultiLayerTracker" vis="SupportVis" reflect="true">

<detector name="SiVertexBarrel" type="SiTrackerBarrel" readout="SiVertexBarrelHits" insideTrackingVolume="true">

<detector name="SiVertexEndcap" type="SiTrackerEndcap2" readout="SiVertexEndcapHits" reflect="true">



DD4hep/examples/CLICSiD

CLIC_SID_CDR Tracker

- Visualized here in geoDisplay
- Around Vertex Detector and beampipe
- Same SiTracker drivers as Vertex Detector

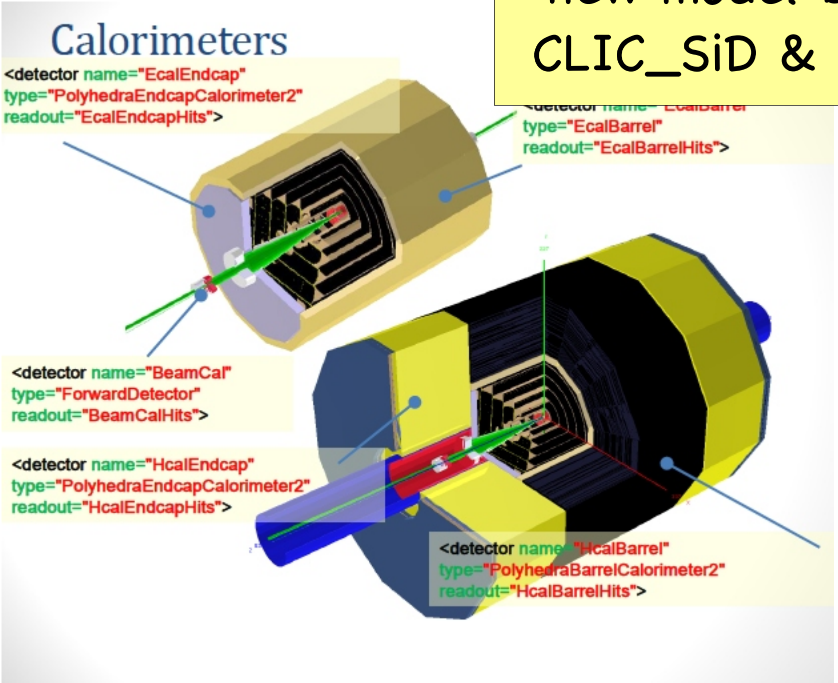
<detector name="SiTrackerBarrel" type="SiTrackerBarrel" readout="SiTrackerBarrelHits" reflect="true">

<detector name="SiTrackerEndcap" type="SiTrackerEndcap2" readout="SiVertexEndcapHits" reflect="true">

- Currently Simulating particles in G4 and comparing hit positions with previous detector geometry chain for validation
  - Also helps fixing bugs in core code

e same tracker visualized with ROOT's TGeoManager using and intermediate GDML file dumped
m Geant4 after loading geometry from DD4hep

new model based on
CLIC_SiD & CLIC_ILD



Calorimeters

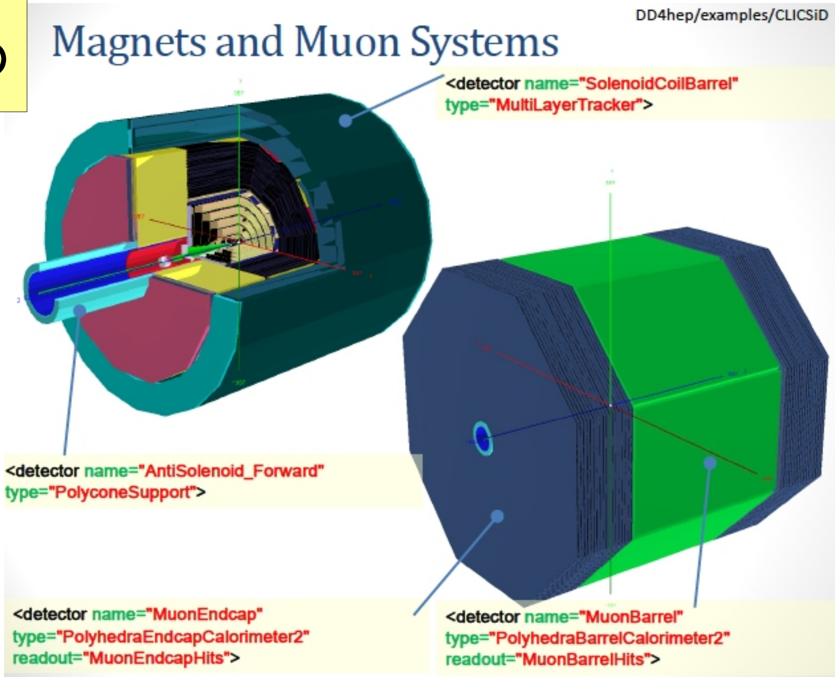
<detector name="EcalEndcap" type="PolyhedraEndcapCalorimeter2" readout="EcalEndcapHits">

<detector name="EcalBarrel" type="EcalBarrel" readout="EcalBarrelHits">

<detector name="BeamCal" type="ForwardDetector" readout="BeamCalHits">

<detector name="HcalEndcap" type="PolyhedraEndcapCalorimeter2" readout="HcalEndcapHits">

<detector name="HcalBarrel" type="PolyhedraBarrelCalorimeter2" readout="HcalBarrelHits">



DD4hep/examples/CLICSiD

Magnets and Muon Systems

<detector name="SolenoidCoilBarrel" type="MultiLayerTracker">

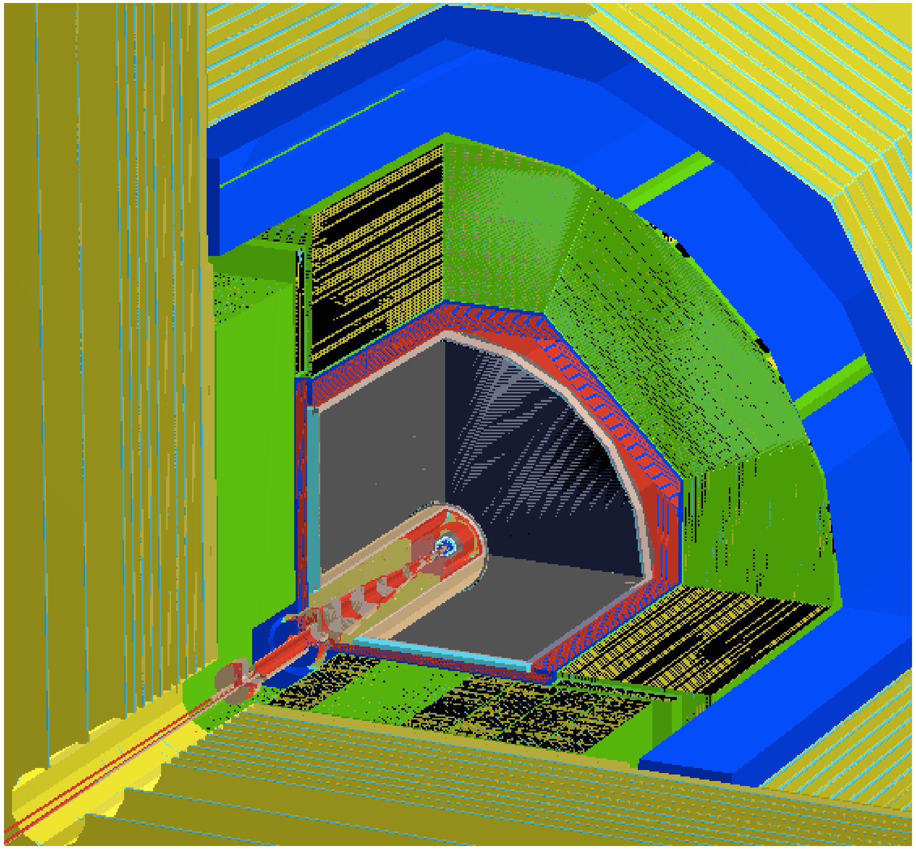
<detector name="AntiSolenoid_Forward" type="PolyconeSupport">

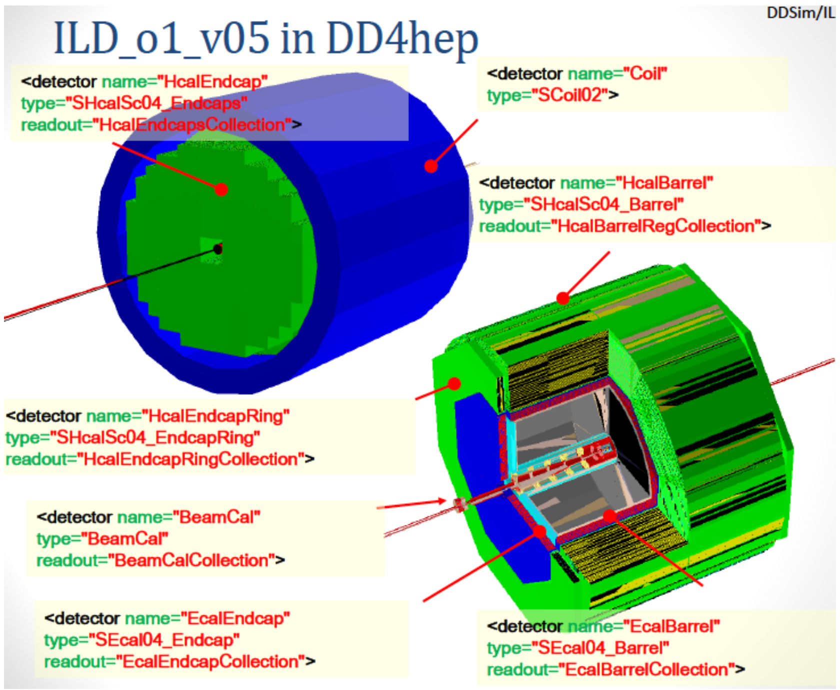
<detector name="MuonEndcap" type="PolyhedraEndcapCalorimeter2" readout="MuonEndcapHits">

<detector name="MuonBarrel" type="PolyhedraBarrelCalorimeter2" readout="MuonBarrelHits">

ILD_o1_v05 in DDSim

- complete Mokka model ILD_o1_v05 ported:
- VXD, FTD, SIT, TPC, SET, beam pipe
- Ecal, Hcal, Yoke
- Beamcal, Lcal, LHcal
- so far only two sensitive detectors





DDSim/IL

ILD_o1_v05 in DD4hep

<detector name="HcalEndcap" type="SHcalSc04_Endcaps" readout="HcalEndcapsCollection">

<detector name="Coil" type="SCoil02">

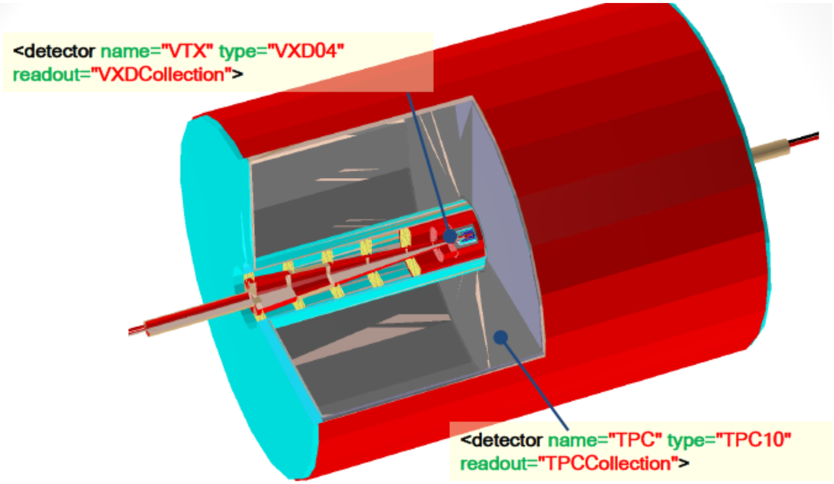
<detector name="HcalBarrel" type="SHcalSc04_Barrel" readout="HcalBarrelRegCollection">

<detector name="HcalEndcapRing" type="SHcalSc04_EndcapRing" readout="HcalEndcapRingCollection">

<detector name="BeamCal" type="BeamCal" readout="BeamCalCollection">

<detector name="EcalEndcap" type="SEcal04_Endcap" readout="EcalEndcapCollection">

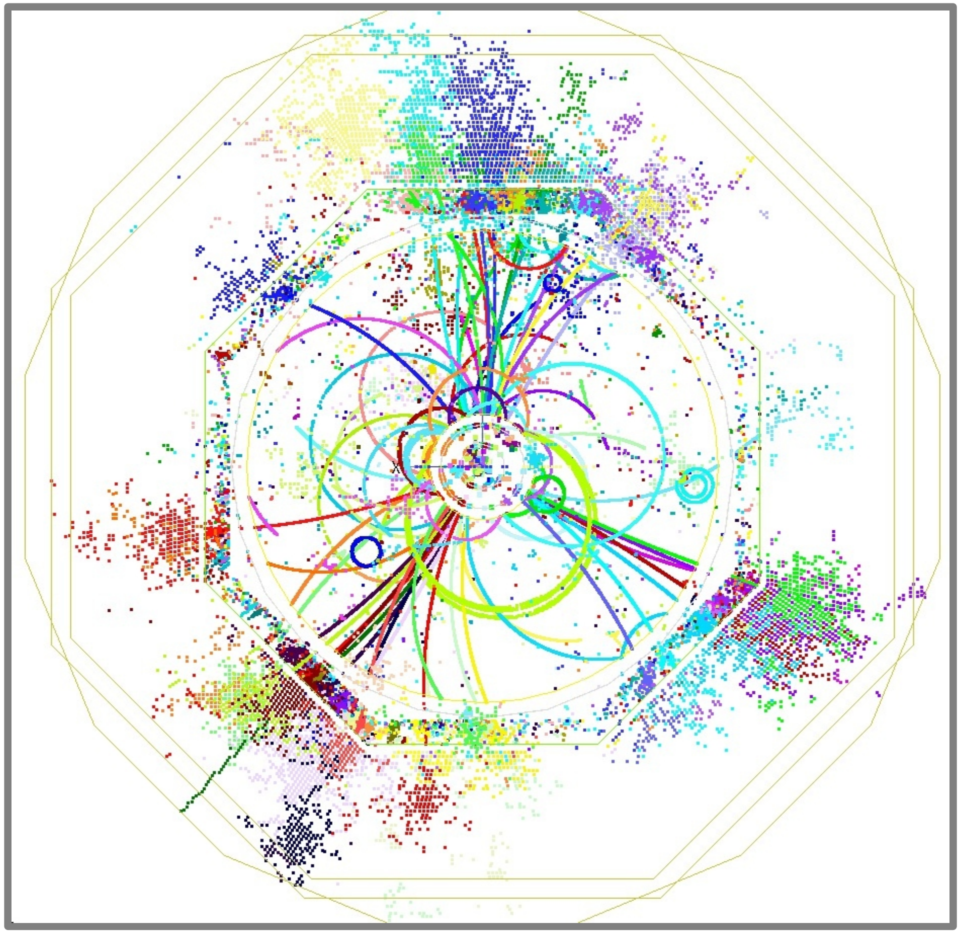
<detector name="EcalBarrel" type="SEcal04_Barrel" readout="EcalBarrelCollection">



<detector name="VTX" type="VXD04" readout="VXDCollection">

<detector name="TPC" type="TPC10" readout="TPCCollection">

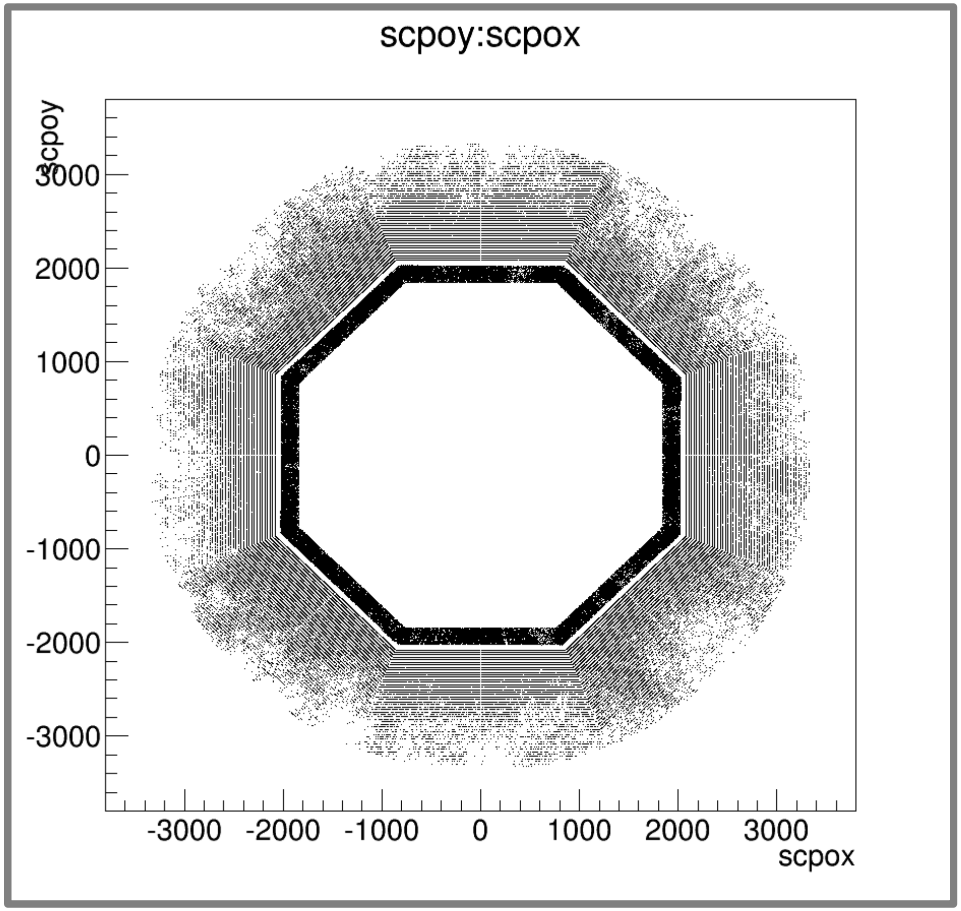
running the ILD simulation I



note: hits colored with MC-truth information in Sim-Hits

- we have now for the first time the complete basic functionality implemented in DD4hep to run a simulation:
- 'first' event simulated in DD4hep model of ILD_o1_v05
- ttbar->bbudsc @ 500 GeV
- first look reveals now major problem:
  - need to look in more detail...

running the ILD simulation II



- first look at hit maps in barrel calorimeter with new ILD_o1_v05
- use standard LCTuple to create hit maps
- work just started:

  need to systematically study all sub detectors ...

Summary & Outlook

- DD4hep now has basic functionality implemented
  - complete MC-truth treatment
  - binding to LCIO SimHits and MCParticles
  - first sensitive detectors using DDSegmentation for Trackers and Calorimeters
- DDSim - new simulation package
  - has first complete simulation model ILD_o1_v05 ported from Mokka
  - soon will have CLIC detector simulation model (from DD4hep)
- started to Test and Debug new functionality and models
- **To Do:**
- interface to GEAR (backward compatibility of reco)
- finalise DDRec and port reco-software
- testing, testing, testing
- optimize detector and detector models ...