Geant4 based Simulation Toolkit: DDG4

André Sailer

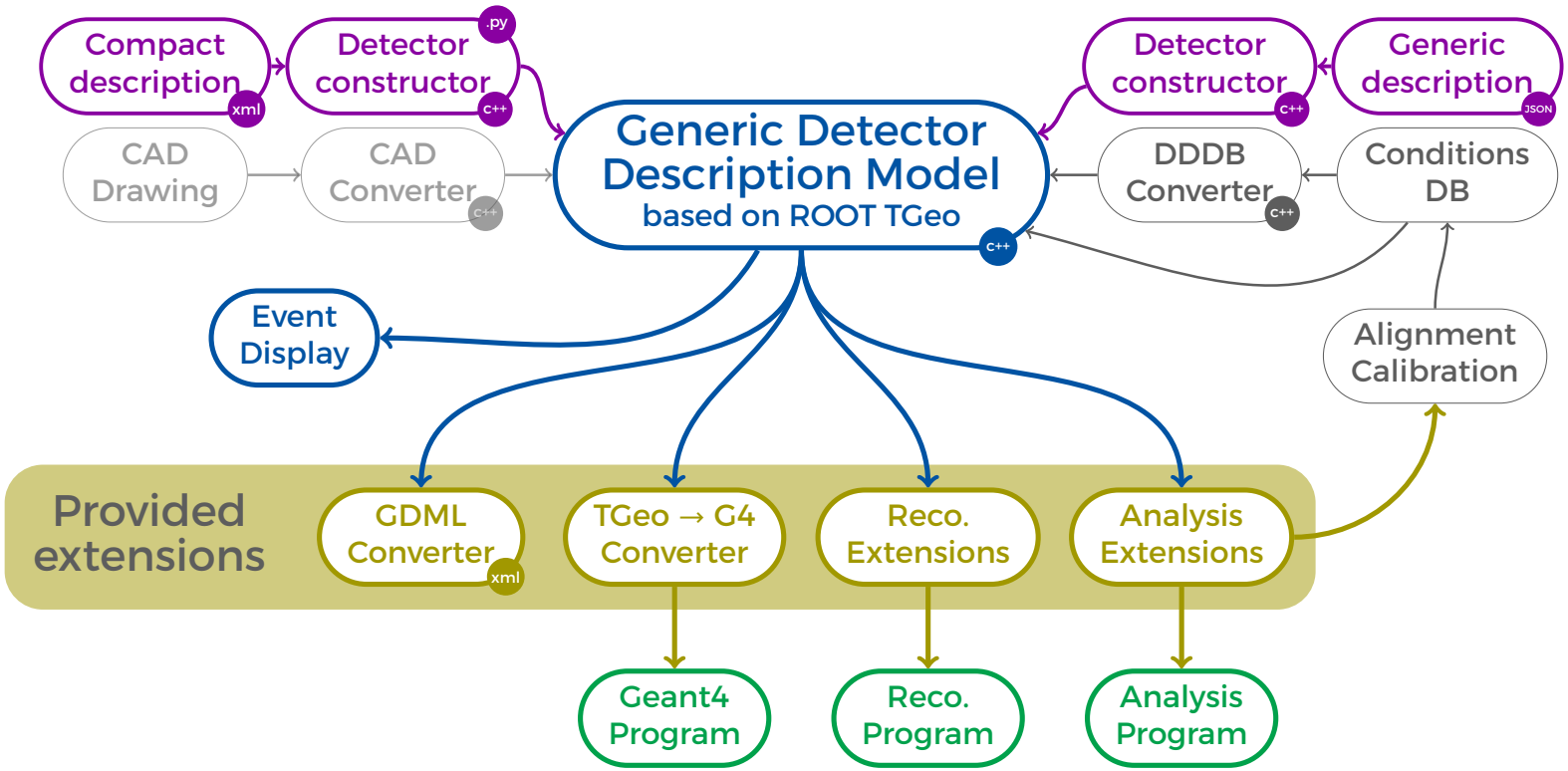
CERN-EP-LCD

Aida2020 – Fourth Annual Meeting
April 4, 2019
Oxford, United Kingdom

Table of Contents

Simulation with DD4HEP

- Convert `TGeo` based geometry to DD4HEP
- Steer GEANT4 workflow



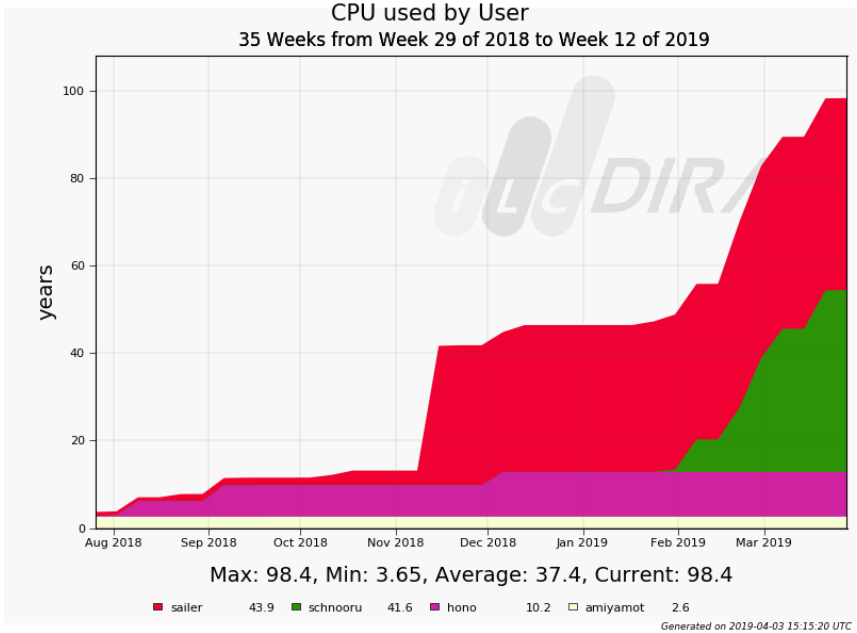
DDG4– DD4HEP's Gateway to GEANT4

- In-memory translation of geometry from TGeo to GEANT4
  - Materials, Solids, Limit sets, Regions
  - Logical volumes, placed volumes and physical volumes
- External configuration:
  - Plug-in mechanism
  - Property mechanism to configure plug-in instances
  - Supports configuration via XML, Python or ROOT-AClick
- Use plug-in mechanism to configure: Generation, Event Action, Tracking Action, Stepping Action, SensitiveDetector, PhysicsList...
- Provides MC truth handling with record reduction
- DDG4 is highly modular
  - Very easily configurable through python (configure actions, filters, sequences, cuts...)

```
#...
part = DDG4.GeneratorAction(kernel, "Geant4ParticleHandler/ParticleHandler")
kernel.generatorAction().adopt(part)
part.SaveProcesses = ['Decay']
part.MinimalKineticEnergy = 1*MeV
part.KeepAllParticles = False
#...
user = DDG4.GeneratorAction(kernel,"Geant4TCUserParticleHandler/UserParticleHandler")
user.TrackingVolume_Zmax = DDG4.tracker_region_zmax
user.TrackingVolume_Rmax = DDG4.tracker_region_rmax
```

DDG4 in Production

- DDG4 based simulation used by ILD, CLIC, FCCee in production
- CLICdp published note on Performance of CLICdet in December
  - Big activity in November to produce final samples for this note
  - Physics analysis productions starting
- FCC: Input to CDR with CLD detector
- ILD: Working on ILD Design Report



Current or Recent Developments

- Optical Surfaces
- DDSIM python program ported from LCGEO to DD4HEP
- Possibility to simulate all events in one file
  - Useful when number of events varies from file to file
- Bug fixes: input readers
- Keeping up with GEANT4

DDSim

- DDSim python executable is now part of DD4hep
- Get steering file `ddsim --dumpSteeringFile > mySteer.py`
  - Steering file includes documentation for parameters and examples
  - The python file contains a `DD4hepSimulation` object at global scope
  - Configure simulation directly from command line

```
from DDSim.DD4hepSimulation import DD4hepSimulation
from SystemOfUnits import mm, GeV, MeV, keV
SIM = DD4hepSimulation()
SIM.compactFile = "CLIC_o3_v06.xml"
SIM.runType = "batch"
SIM.numberOfEvents = 2
SIM.inputFile = "electrons.HEPEvt"
SIM.part.minimalKineticEnergy = 1*MeV
SIM.filter.filters ['edep3kev'] =
dict (name="EnergyDepositMinimumCut/3keV" ,
      parameter={"Cut" : 3.0*keV} )
```

```
$ ddsim
--action.calo                --filter.tracker             --part.keepAllParticles
--action.mapActions          -G                           --part.minimalKineticEnergy
--action.tracker             --gun.direction              --part.printEndTracking
--compactFile                --gun.energy                 --part.printStartTracking
--crossingAngleBoost         --gun.isotrop                --part.saveProcesses
--dump                       --gun.multiplicity           --physics.decays
--dumpParameter              --gun.particle               --physics.list
--dumpSteeringFile           --gun.position               --physicsList
--enableDetailedShowerMode   -h                           --physics.rangecut
--enableGun                  --help                       --printLevel
--field.delta_chord          -I                           --random.file
--field.delta_intersection   --inputFiles                 --random.luxury
--field.delta_one_step       -M                           --random.replace_gRandom
--field.eps_max              --macroFile                  --random.seed
--field.eps_min              -N                           --random.type
--field.equation             --numberOfEvents             --runType
--field.largest_step         -O                           -S
--field.min_chord_step       --outputFile                 --skipNEvents
--field.stepper              --output.inputStage          --steeringFile
--filter.calo                --output.kernel              -v
--filter.filters             --output.part                --vertexOffset
--filter.mapDetFilter        --output.random              --vertexSigma
```

Summary

- DDG4 stable and used in production for ILC, CLIC, and FCCee
- New users bring new requirements