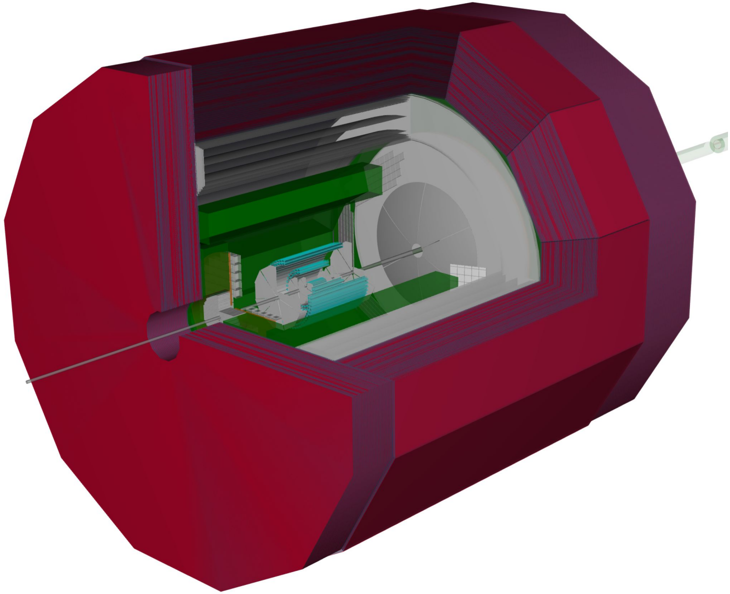


Discussion over software choices and

ATHENA Software for PID working group

Thursday 2021-06-07

The ATHENA S&C WG conveners:
Andrea Bressan (University of Trieste and INFN) ,
Dmitry Romanov (Jefferson lab) ,
Sylvester Joosten (Argonne National Laboratory) ,
Whitney Armstrong (Argonne National Laboratory) ,
Wouter Deconinck (The University of Manitoba).

Questions from PID group

1. What simulation software framework are we considering focusing on?
   a. I've been developing a port of the dRICh into fun4all, but it seems that focus in the SW group is more on dd4hep.
   b. I'm wondering if I should instead start working on a dd4hep port before getting too far in fun4all

2. Why are we moving to a new simulation framework (DD4hep) rather than using one of the run-in tools/frameworks (Fun4all, Escalate) that we might have started using for the Yellow Report?

3. Does DD4hep offer today all the features needed to do detector performance studies?
   a. Can I run tracking and get a reconstructed track length for TOF?
   b. Or the extrapolated / interpolated track position / direction vector at any place?

4. How does DD4hep handle complex material properties, like material optical properties and optical boundaries?

5. Who is supposed to take care of the software implementation within dd4hep?
   a. And the validation?
   b. What would be the reference framework for validation?

Software plan

Looking into the future of EIC

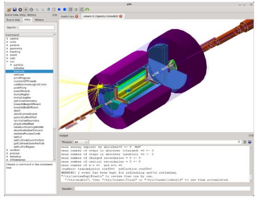
- Build forward looking software
- Old solutions (such as EicRoot or Fun4All) should go - computing paradigms are changing quickly!
- Focus on modern scientific computing practices.
- Avoid “not-invented-here syndrome”.
- Use modern, supported by large communities (like CERN) software components.
- Mature, well supported, and actively developed software stack to hit the ground running.

Immediate concern: successful (best!) detector proposal

As the main goal of ATHENA collaboration is to present successful (and the best!) detector proposal, we understand that there should be no disruption of WG simulation process after the DWGs and PGWs are formed. Previously WGs already invested their time into various simulation software and advanced with the simulations.

ESCalate merges the efforts

G4E – Geant 4 EIC

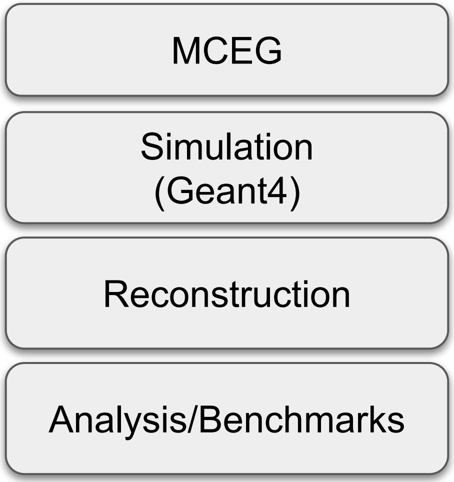


Now → Support for G4E while it is used. But no major development

Near term → Detectors and Beamline where merged into DD4Hep

Long term → G4E is a pure Geant4 implementation => code supposed to be consumed by EAST project as soon as it appears

Oversimplified software stack



Oversimplified software stack

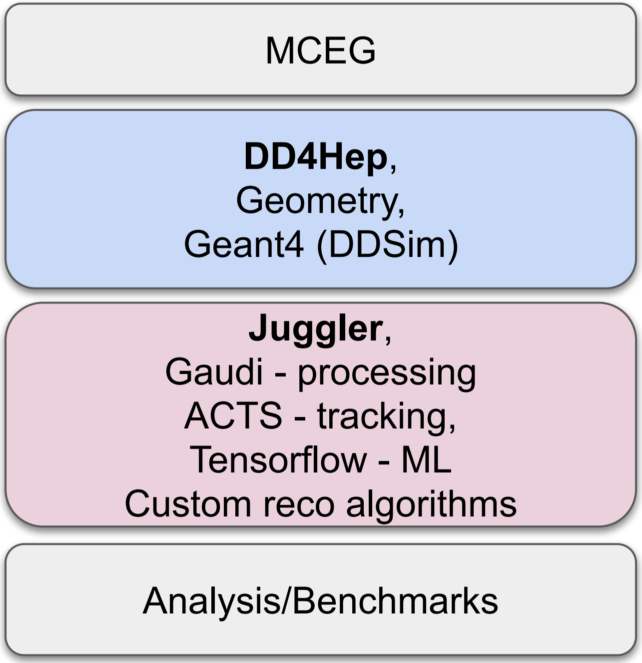
DD4hep: Geant4 geometry, detector plugin library, wrappers to run Geant4

Juggler: Digitization and reconstruction software (based on Gaudi with Podio-based data model and ACTS for tracking)

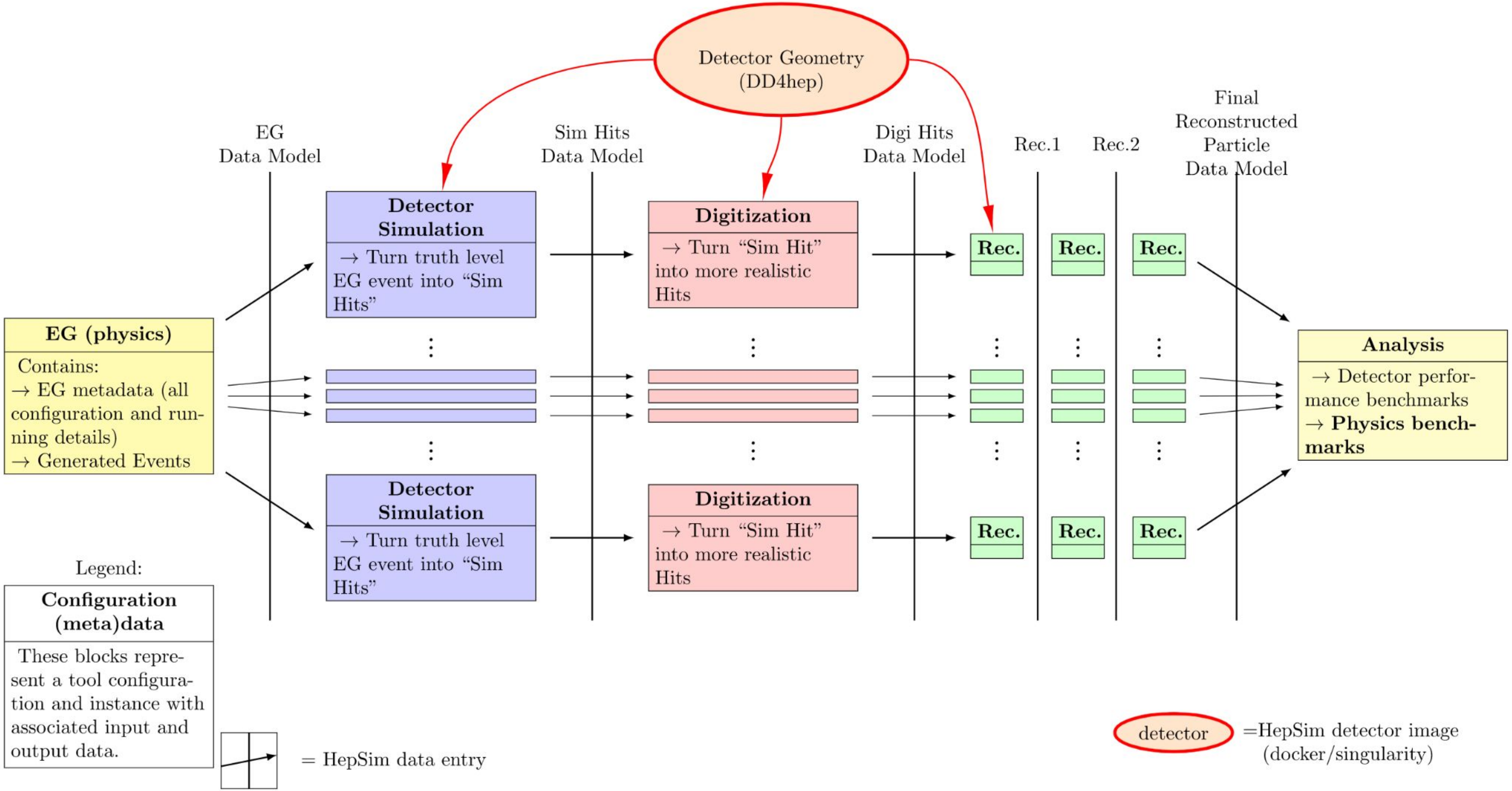
GAUDI: Generic open project for building event processing frameworks. Enables modern task-based concurrent execution in a heterogeneous computing environment. Used by ATLAS and LHCb.

ACTS: Experiment-independent tracking toolkit (ACTS' geometry constructed from DD4hep via plugin)

Podio: Robust data model definition to cross the boundaries between the tools



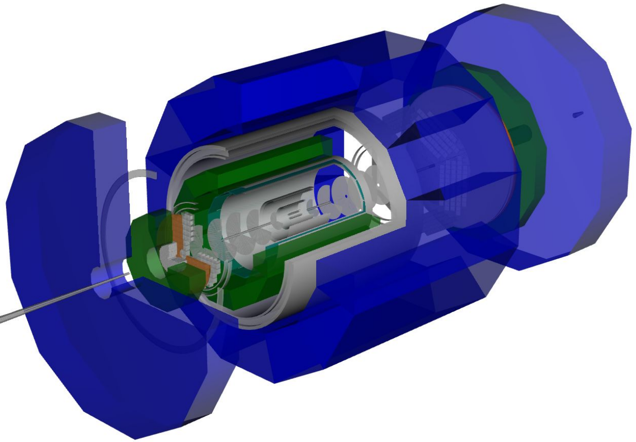
DD4Hep handling geometry problem



DD4Hep community

DD4Hep overview

- **Provides a complete detector description ˆ**
  - Geometry, materials, visualization, readout, alignment, calibration, …ˆ
- **From a single source of geometry informationˆ**
  - Used in simulation, reconstruction, analysisˆ
  - Easy integrates with plugins such as ACTSˆ
- **Comes with a powerful plug-in mechanismˆ**
  - Switch subdetectors for other existing onesˆ, customize via XML
- **More or less "industry standard" by now**
  - ˆILC, CLIC, FCC, CEPC, LHCb, …ˆ CMS is switching to DD4hep

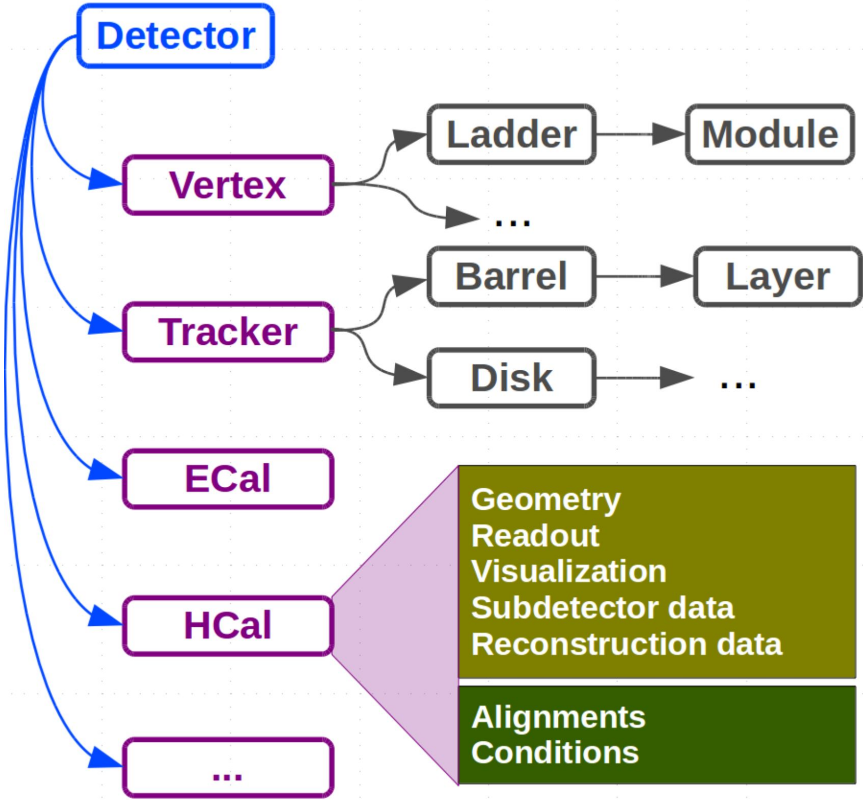


Only use what you need!

https://indico.jlab.org/event/198/contributions/1703/attachments/1471/1910/Gaede-DD4hep.pdf

DD4Hep detector description

- Description of a tree-like hierarchy of ‘detector elements’
  - Sub-detectors or parts of subdetectors
- Detector Element describes:
  - Geometry
  - Environmental conditions
  - Properties required to process event data
  - Extenstions (optionally): experiment, sub-detector or activity specific data, measurement surfaces…



How detectors are implemented programmatically

- Generic driver available → scalable and flexible
- Parameters are provided in compact XML files, e.g.

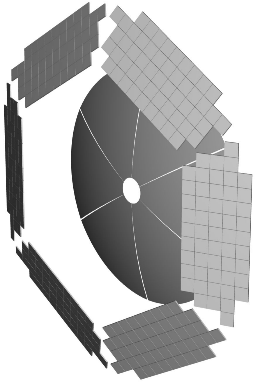
```xml
<detector id="15" name="HCal" type="GenericCalBarrel_o1_v01" readout="HCalCollection">
  <envelope vis="HCALVis">
    <shape type="PolyhedraRegular" numsides="HCal_sym" rmin="HCal_rmin" rmax="HCal_rmax" dz="HCal_dz" material="Air"/>
    <rotation x="0*deg" y="0*deg" z="90*deg-180*deg/HCal_symmetry"/>
  </envelope>
  <dimensions numsides="HCal_sym" rmin="HCal_rmin" z="HCal_dz*2"/>
  <layer repeat="(int) HCal_layers" vis="HCalLayerVis">
    <slice material="Steel235"    thickness="0.5*mm" vis="HCalAbsorberVis" radiator="yes"/>
    <slice material="Steel235"    thickness="19*mm"  vis="HCalAbsorberVis" radiator="yes"/>
    <slice material="Polystyrene" thickness="3*mm"   sensitive="yes"  limits="cal_limits"/>
    <slice material="Copper"      thickness="0.1*mm" vis="HCalCopperVis"/>
    <slice material="PCB"         thickness="0.7*mm" vis="HCalPCBVis"/>
    <slice material="Steel235"    thickness="0.5*mm" vis="HCalAbsorberVis" radiator="yes"/>
    <slice material="Air"         thickness="2.7*mm" vis="InvisibleNoDaughters"/>
  </layer>
</detector>
```

- You can scale, change layers, radii and compositions...
- Propagate visualization attributes to Display

```cpp
static Ref_t create_detector(LCDD& lcdd,
       xml_h e, SensitiveDetector sens) {

  xml_det_t x_det = e;
  Layering layering(x_det);
  xml_comp_t staves = x_det.staves();
  xml_dim_t dim = x_det.dimensions();
  DetElement sdet(det_name, x_det.id());
  Volume motherVol = lcdd.pickMotherVolume(sdet);

  PolyhedraRegular polyhedra(numSides, rmin, rmax, detZ);
  Volume envelopeVol(det_name, polyhedra, air);

  for (xml_coll_t c(x_det, _U(layer)); c; ++c) {
    xml_comp_t x_layer = c;
    int n_repeat = x_layer.repeat();
    const Layer* lay = layering.layer(layer_num - 1);
    for (int j = 0; j < n_repeat; j++) {
      string layer_name = _toString(layer_num, "layer%d");
      double layer_thickness = lay->thickness();
      DetElement layer(stave, layer_name, layer_num);
```

https://cds.cern.ch/record/2229554/files/AIDA-2020-SLIDE-2016-009.pdf

Complex material properties, optical properties



- **import of surface optical objects in compact input files:**
  - surface types and optical properties (refraction, absorption, . . .)
  - create TGeo surface objects and tabulated properties
- **translation from TGeo to Geant4**
- **physics components in DDG4, handling**:
  - scintillation, Cerenkov and transition radiation
  - reflection, refraction, absorption,wavelength shifting

As a backup, one can access to all Geant4 parts

Compact file

```
<opticalsurface name="/world/BubbleDevice#WaterSurface" finish="ground" model="unified"
                type="dielectric_dielectric">
  <property name="RINDEX"                coldim="2" values="2.034*eV  1.35   4.136*eV  1.40"/>
  <property name="SPECULARLOBECONSTANT"  coldim="2" values="2.034*eV  0.3    4.136*eV  0.3 "/>
  <property name="SPECULARSPIKECONSTANT" coldim="2" values="2.034*eV  0.2    4.136*eV  0.2 "/>
  <property name="BACKSCATTERCONSTANT"   coldim="2" values="2.034*eV  0.2    4.136*eV  0.2 "/>
</opticalsurface>
```

C++ side

```
// Now attach the surface
OpticalSurfaceManager surfMgr = description.surfaceManager();
OpticalSurface waterSurf  = surfMgr.opticalSurface("/world/"+det_name+"#WaterSurface");
OpticalSurface airSurf    = surfMgr.opticalSurface("/world/"+det_name+"#AirSurface");
BorderSurface  tankSurf   = BorderSurface(description, sdet, "HallTank",   waterSurf,
                                          tankPlace,   enclPlace);
BorderSurface  bubbleSurf = BorderSurface(description, sdet, "TankBubble", airSurf,
                                          bubblePlace, tankPlace);
```

https://cds.cern.ch/record/2718830/files/AIDA-2020-SLIDE-2020-027.pdf

Automated workflows with eicweb

GitLab server (eicweb.phy.anl.gov)

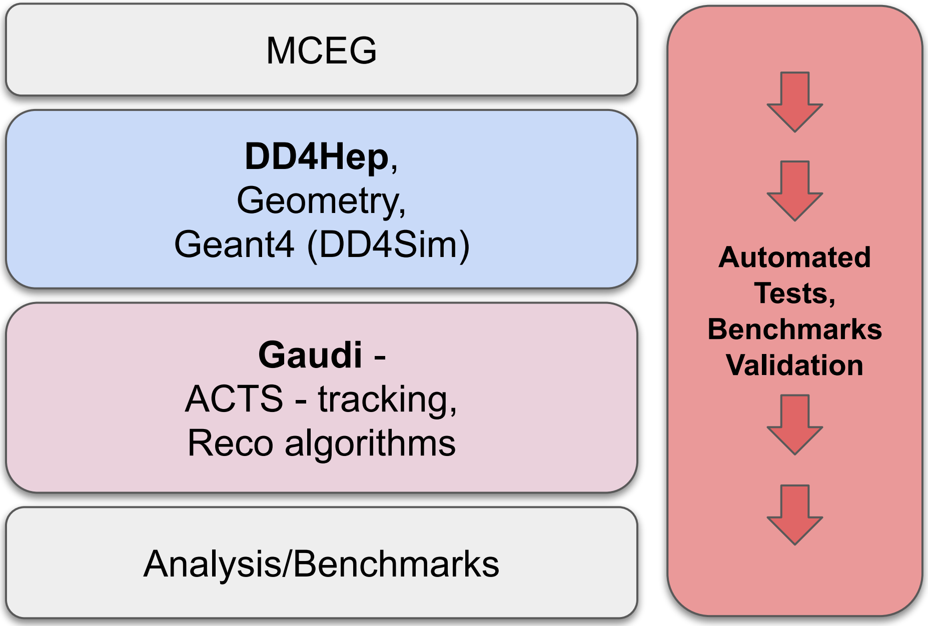
- continuous integration
- dedicated build cluster

Runs automatically on each user commit, executing workflows running multiple tests, benchmarks and analysis

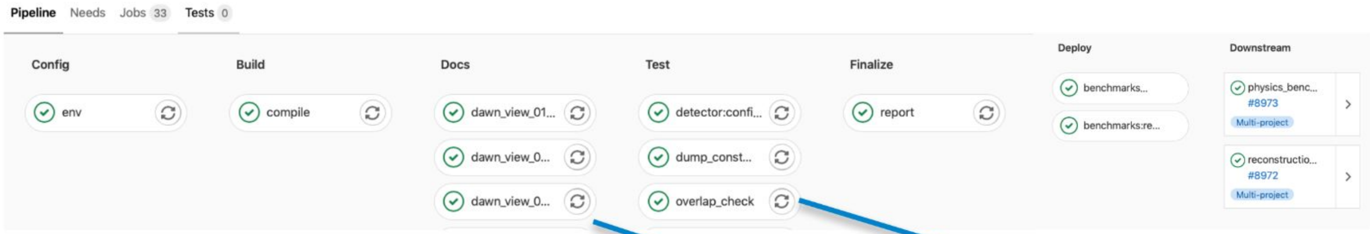
Automated containers

Both Docker and Singularity images are created nightly or on demand (commit) providing:

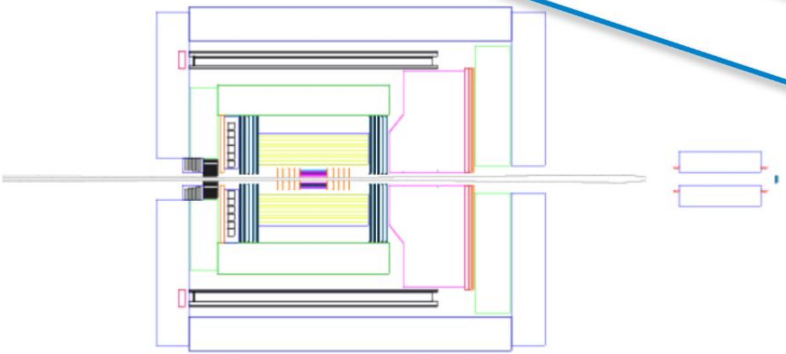
- reproducibility,
- production level images
- latest updates for those working locally



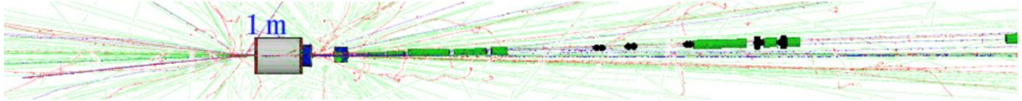
Automatic benchmarks, documentation, conterizatino

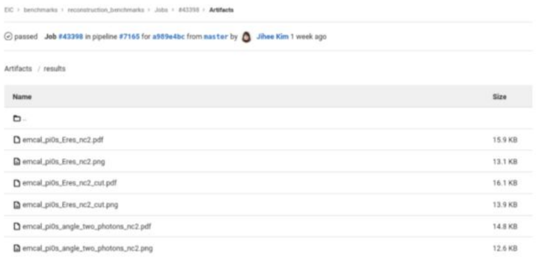


Geometry overlap checks running as part of every merge request



Automatic visualizations for detector geometries, saved as job artifacts (browsable!)





Detector status

ATHENA detector @ eicweb

- Implemented geometries for most subsystems
- New interactive in-browser geometry viewers:
  - Central detector
  - Full detector with beamline
- Reasonable detail in central detector tracking
- **Needs PID and working with PID groups!**

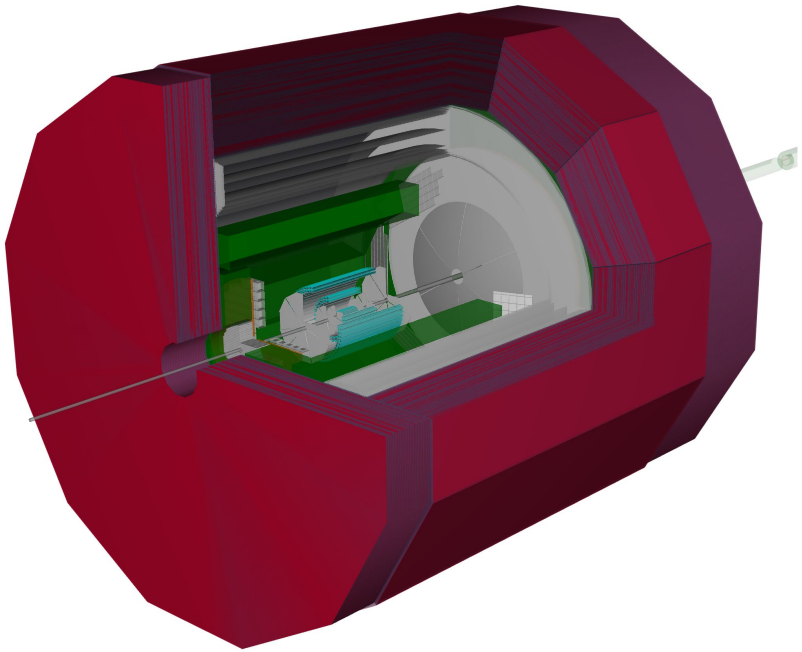
  Currently have implementation of:

  LGAD layer (Zhenyu Ye)

  Basic gas RICH

  Reconstruction: working Fuzzy-K Clustering for RICH

  DIRC only geometry so far (needs most work)



PWG ⟺ Software WG ⟺ DWG interaction

Our vision so far (very open to discussion):

1. Every group delegates 1 person to work with software groups

2. For PWG imminent TODO:
   a. What benchmarks/plots/studies should we have for report
   b. Workflow to integrate analysis in CI pipeline

3. For DWG imminent TODO:
   a. Work on detector implementation details
   b. How-to implement/modify a detector in DD4Hep

4. For all - gather information
   a. What are their plans and vision of now
   b. If additional resources/help needed for that

Detectors implementation plan

- **Implement it one by one**
  We work with detector working groups to help transfer the detectors. So they could pick up future development and details improvements

- **Hackathons**
  We plan to have hackathons, where we will sit together with users and try to help/implement everything missing

- **Engaging users**
  Doing tutorials on our software stack, instant help on SWG slack, etc

Full simulation tutorial

https://eic.phy.anl.gov/tutorials/eic_tutorial/getting-started/quickstart

Environment setup

BASH

```
curl https://eicweb.phy.anl.gov/containers/eic_container/-/raw/master/install.sh | bash
```

eicweb tutorial

Home

GETTING STARTED

Overview

TUTORIAL

Quick-start

Part 1: Simple Detectors

Part 2: Adding Detectors

Quick-start Guide

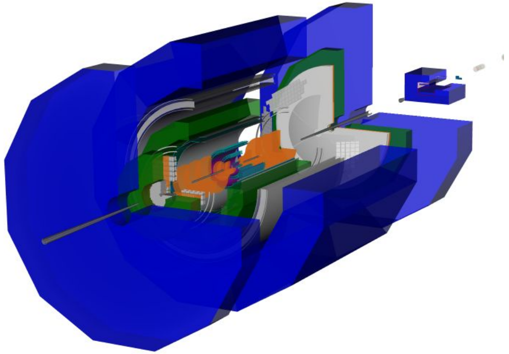
Prerequisites

We assume that you are somewhat familiar with:

- git and gitlab (https://eicweb.phy.anl.gov)
- working in the terminal over ssh and can copy files to your local machine.
- cmake, C++, python, and shell scripting

This tutorial requires that singularity is installed on the local system. Singularity ca
most recent version as of January 2021.

Geometry visualization



Realistic geometry descriptions

A more detailed GEM Tracker

Uncomment the include line in `gem_tracker.xml`

Copy

```
<include ref="compact/gem_tracker_endcap.xml"/>
```

And have a look at the detector part of `compact/gem_tracker_endcap.xml`.

Copy

```
<detector
  id="GEMTrackerEndcap_ID"
  name="GEMTrackerEndcap"
  type="MyGEMTrackerEndcap"
  readout="GEMTrackerEndcapHits"
  vis="BlueVis"
  reflect="false">
  <module name="GEMModule1" vis="GreenVis">
    <trd x1="GEMTrackerEndcapFoilX1/2.0" x2="GEMTrackerEndcapFoilX2/2.0" z="GEMTrackerEndcapFoilY/2"/>
    <comment> Going from HV side to readout side</comment>
    <module_component thickness="0.127 * mm" material="Mylar"/>
    <module_component thickness="50.0*um" material="Kapton" name="entrance_window"/>
    <module_component thickness=" 3.0*mm" material="Ar10CO2" name="entrance region" />
    <module_component thickness="50.0*um" material="Kapton"/>
    <module_component thickness=" 3.0*um" material="Copper"/>
    <module_component thickness=" 3.0*mm" material="Ar10CO2" name="drift region"/>
    <module_component thickness="30.0*um" material="Kapton"  name="gem_foil"/>
    <module_component thickness=" 3.0*um" material="Copper"  name="gem_foil_Cu"/>
    <module_component thickness=" 2.0*mm" material="Ar10CO2" name="transfer region I"/>
    <module_component thickness="30.0*um" material="Kapton"  name="gem_foil"/>
```

Introduction to geometry plugins

The corresponding detector constructor function is in `src/TrapEndcapTracker_geo.cpp`

```
static Ref_t build_detector(Detector& description, xml_h e, SensitiveDetector sens) {
  typedef vector<PlacedVolume> Placements;
  xml_det_t                     x_det    = e;
  Material                      vacuum   = description.vacuum();
  int                           det_id   = x_det.id();
  string                        det_name = x_det.nameStr();
  bool                          reflect  = x_det.reflect(false);
  DetElement                    sdet(det_name, det_id);
  Assembly                      assembly(det_name);
  Volume                        motherVol = description.pickMotherVolume(sdet);
  int                           m_id = 0, c_id = 0, n_sensor = 0;
  map<string, Volume>           modules;
  map<string, Placements>       sensitives;
  PlacedVolume                  pv;

  assembly.setVisAttributes(description.invisible());
  sens.setType("tracker");
```

User support and documentation

Documentation portal

athena-soft.readthedocs.io/

Central documentation hub.

Goal: one-stop-shop with easy to navigate content.

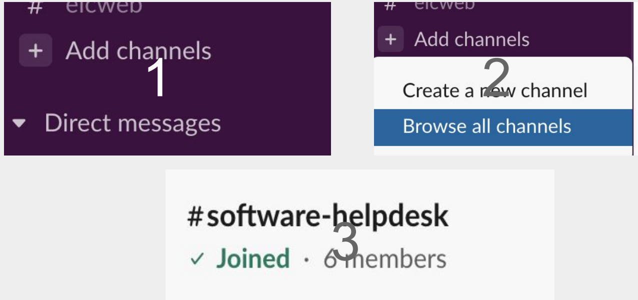
Full simulation tutorials

https://eic.phy.anl.gov/tutorials/eic_tutorial/getting-started/quickstart

Where should I ask questions about software and computing?

Your choice! either send to the eic-ip6-software-l@lists.bnl.gov mailing list, or use the brand new **#software-helpdesk** Slack channel.

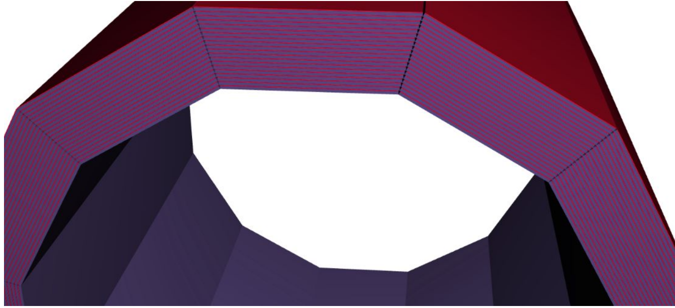
You can subscribe to this new channel by clicking “Add Channels, then “Browse all channels”.



Meetings: Every other Thursday, 12pm

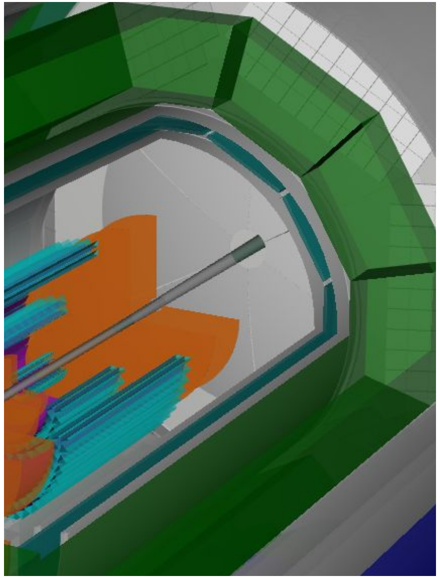
BACKUP SLIDES

Updates



Simulation

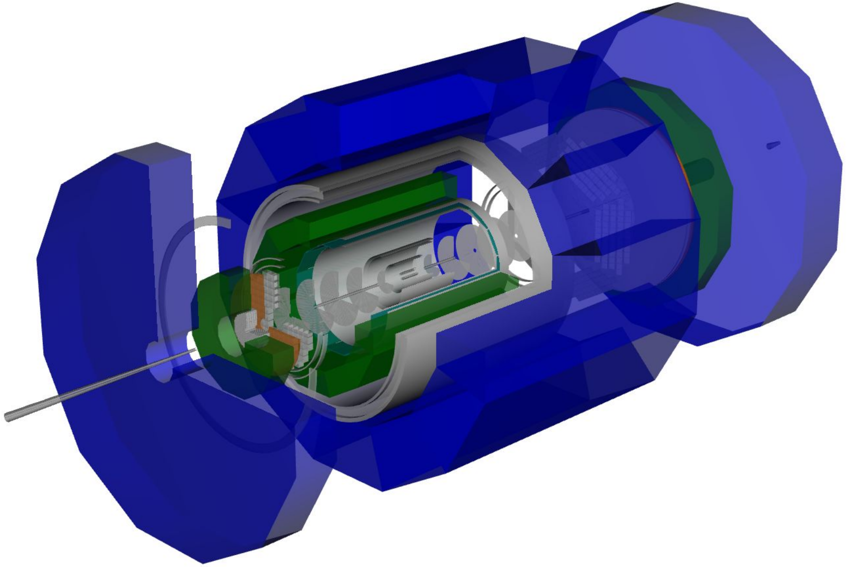
- Released first general purpose software container (used in the tutorial)
- Added more realistic parametrization for the silicon tracker barrels to the athena repository, fixed geometries for use in ACTS
- New HCAL layout (Miguel)



Misc

- Deployed web-viewer for ATHENA geometry implementation
  - (view with full beamline)
- Geometries ready for tracker validation
- Computing update → Wouter
- Documentation → Dmitry
- We have people attending all WG meetings, still need "Official" WG contacts to the S&C WG

New Athena repository (fork from reference_detector)



- Implemented geometries for most subsystems based on the reference detector.
- New interactive in-browser geometry viewers:
  - Central detector
  - Full detector with beamline
- Reasonable detail in central detector tracking
- In particular RICH, DIRC and HCAL need input from the WGs
- Next TODO
  - Update the far-forward detector system (had meeting yesterday with Alex Jentsch)
  - Add more detail to tracking system
  - Add updated magnet designs