Integration software suite: overview

- **A tool to model & generate EIC Central Detector "templates" in a way:**
  - the new geometries (models) can be generated "quickly" …
  - … and represented instantly in a WYSIWYG fashion
  - the sub-detector "container objects" are guaranteed to not overlap either with each other or with the IR vacuum chamber elements
  - they can be imported in GEANT frameworks in a consistent way and used perhaps as wrappers to the "real" sub-detectors
  - they can be exported in a CAD format to be used in the engineering design of the detector support structures and / or laying out services

- **Current version can do more than that:**
  - vacuum chamber prototyping and export in TGeo and GDML formats
  - B_T*dl integral evaluation for the endcap silicon trackers
  - beam pipe material scan at small scattering angles
  - models are persistent: can be saved and imported back *as a single entity*

Integration software suite: limitations

- **Four pre-defined detector "stacks": vertex, barrel, and two endcaps** …
- … **in a projective configuration (defined by the η ranges)**

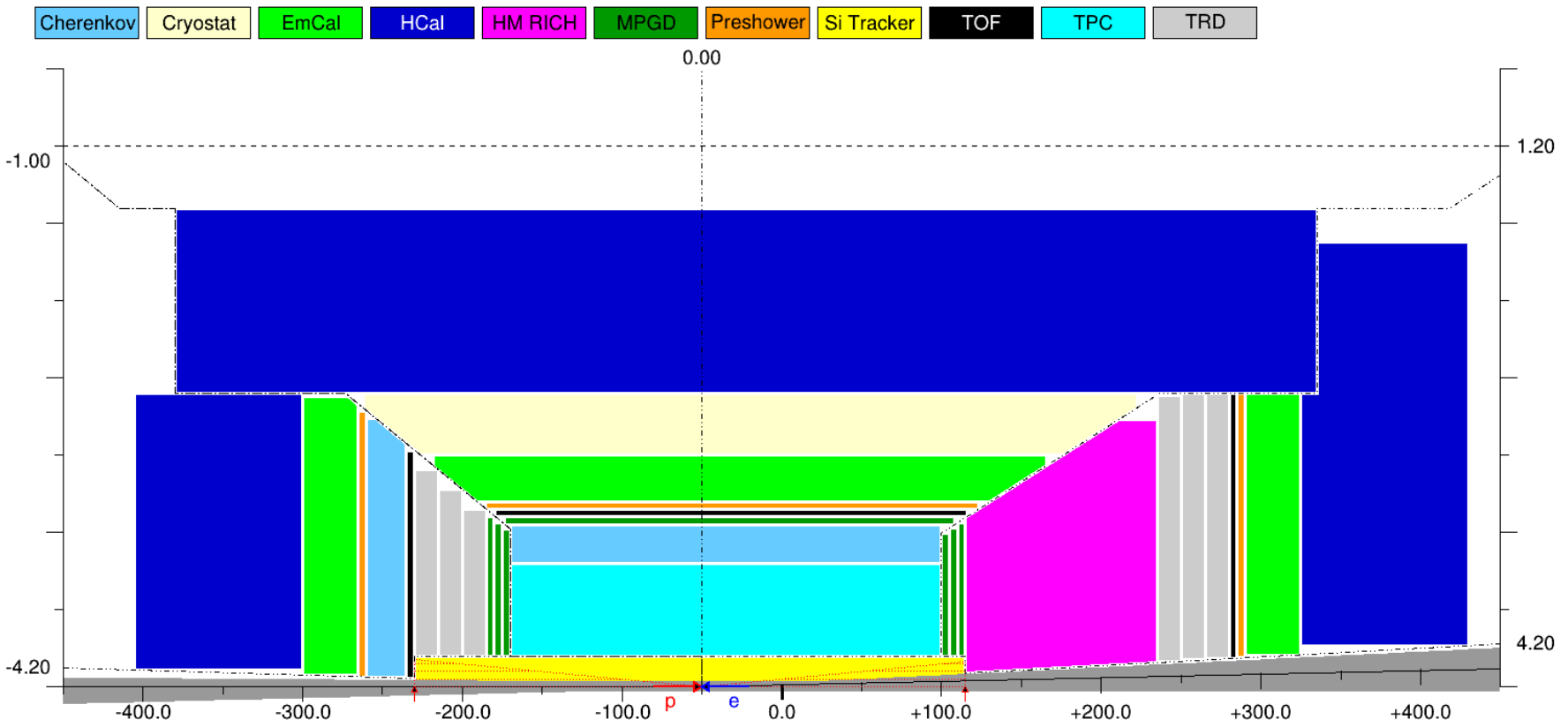
- **Detector tags (like "EmCal") and respective colors are hardcoded …**
  - … though custom ones can be generated dynamically, if really needed

- **Detector volumes in a given stack are placed as objects with flat front and rear sides, one after the other, strictly orthogonal to either the electron beam line axis or to a normal to this axis in a 2D view …**
  - ... although stack boundaries can be shaped up creatively, if needed

- **Exported objects are azimuthally symmetric polycones, although …**
  - … with an asymmetric cutaway representing the IR vacuum chamber
  - … polyhedra (segmentation in wedges) will be implemented as well

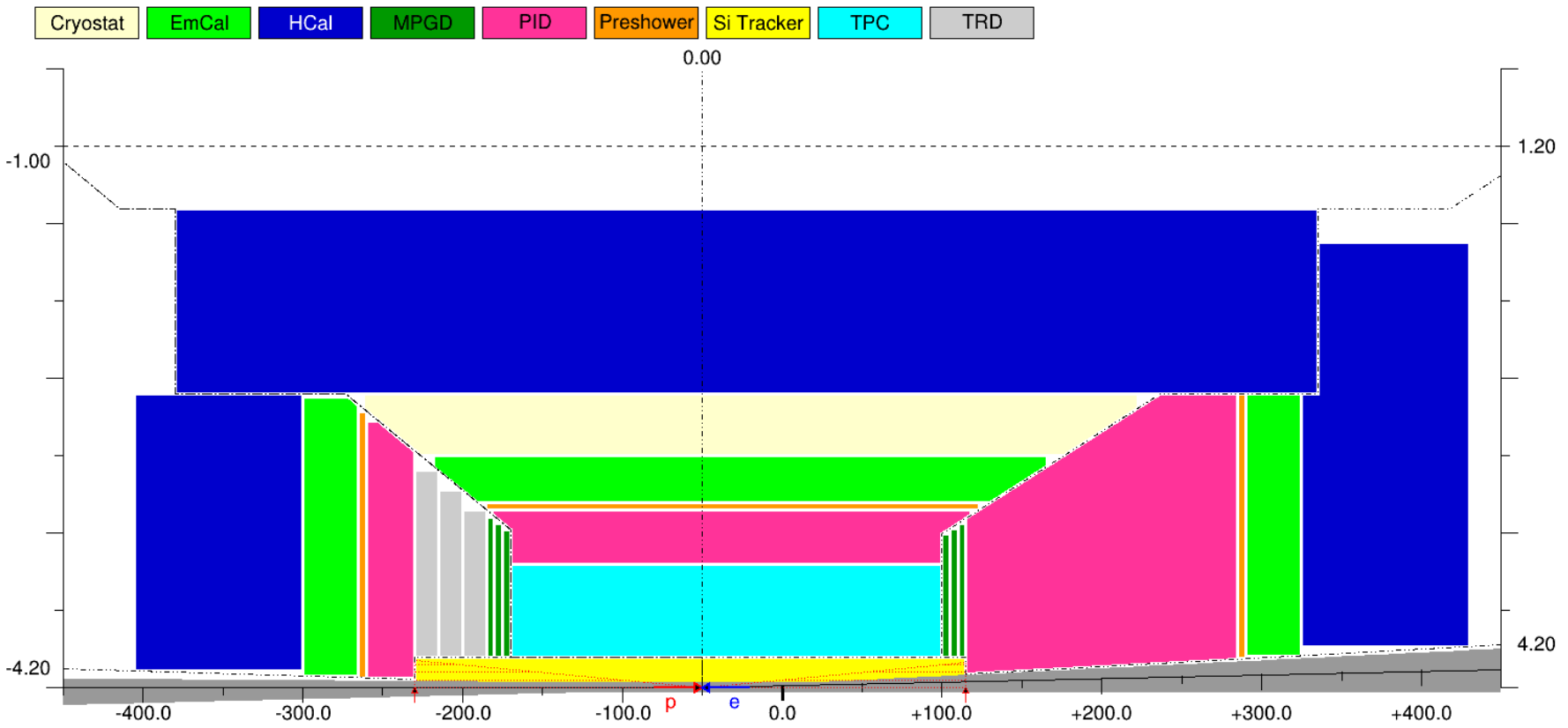
- **Stack boundary crack width (support, services) is still work in progress**

Integration software suite: 2D cartoon view



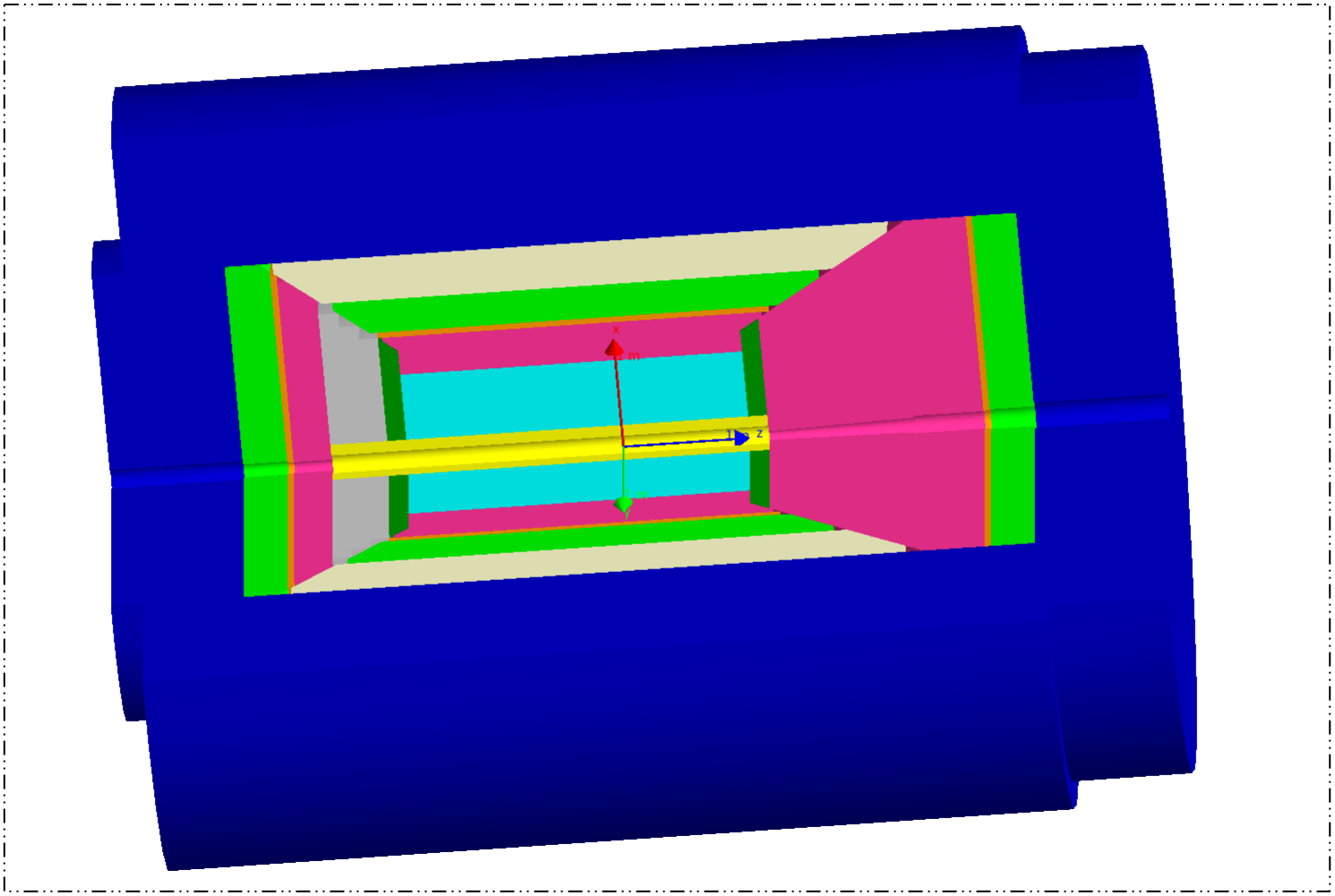
- **Repository:** https://github.com/eic/EicToyModel
  - has a README file, a calorimetry and a PID example
  - detailed API description is available

Same configuration, PID detectors merged



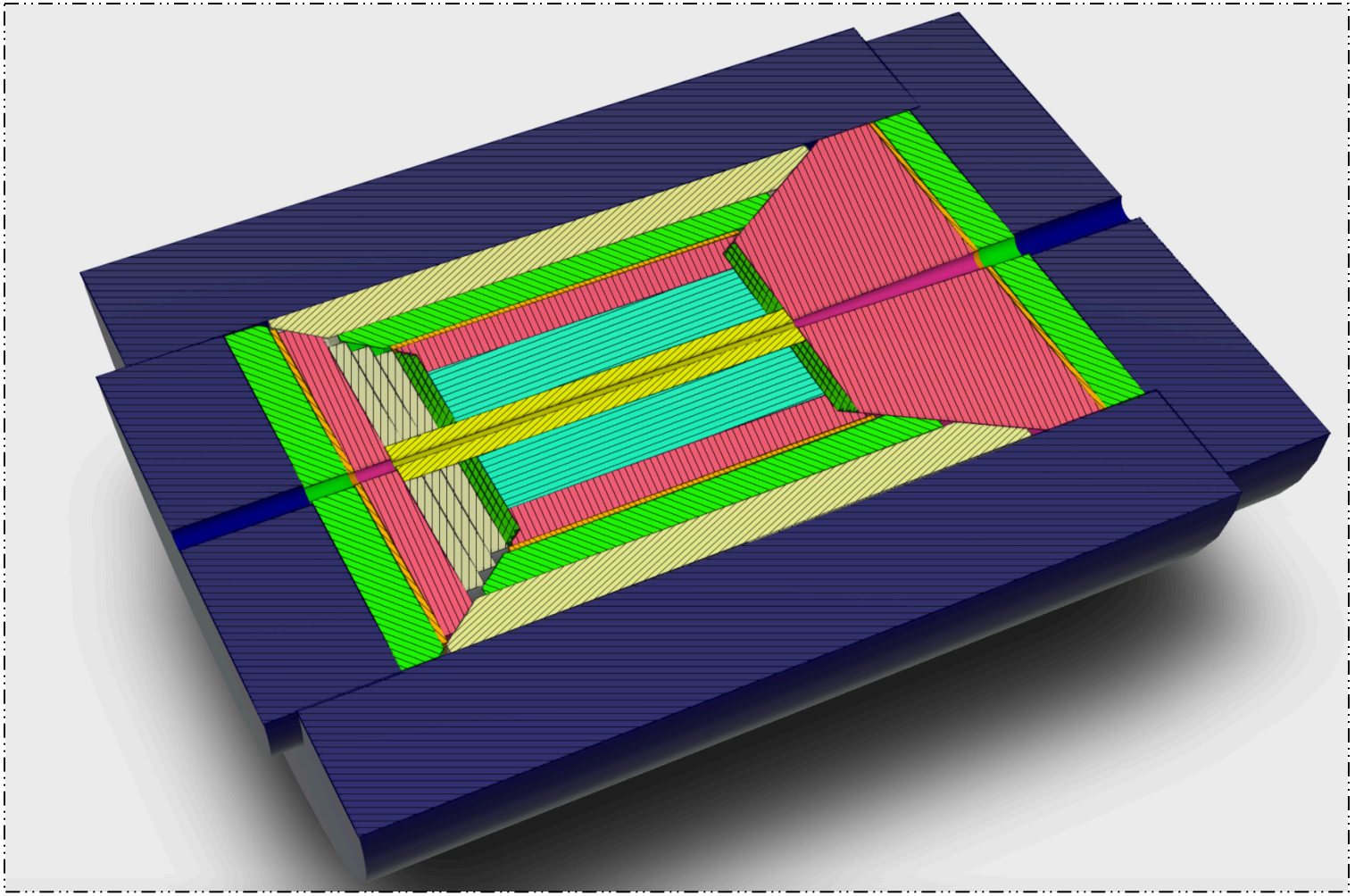
- **All the available stack layout & boundary options are shown here**
  - https://github.com/eic/EicToyModel/blob/master/scripts/pid.C
  - Is it flexible enough for PID detectors? Otherwise what is missing?
  - Which scheme is preferred for PID (detailed, merged, mixed)?

GEANT view (Qt event display)



- Bonus#1: this picture will look the same in fun4all and g4e
- Bonus#2: the volumes are consistent with the IR vacuum chamber layout, per construction (*not yet; in debugging stage*)

CAD view (3D model in Autodesk viewer)

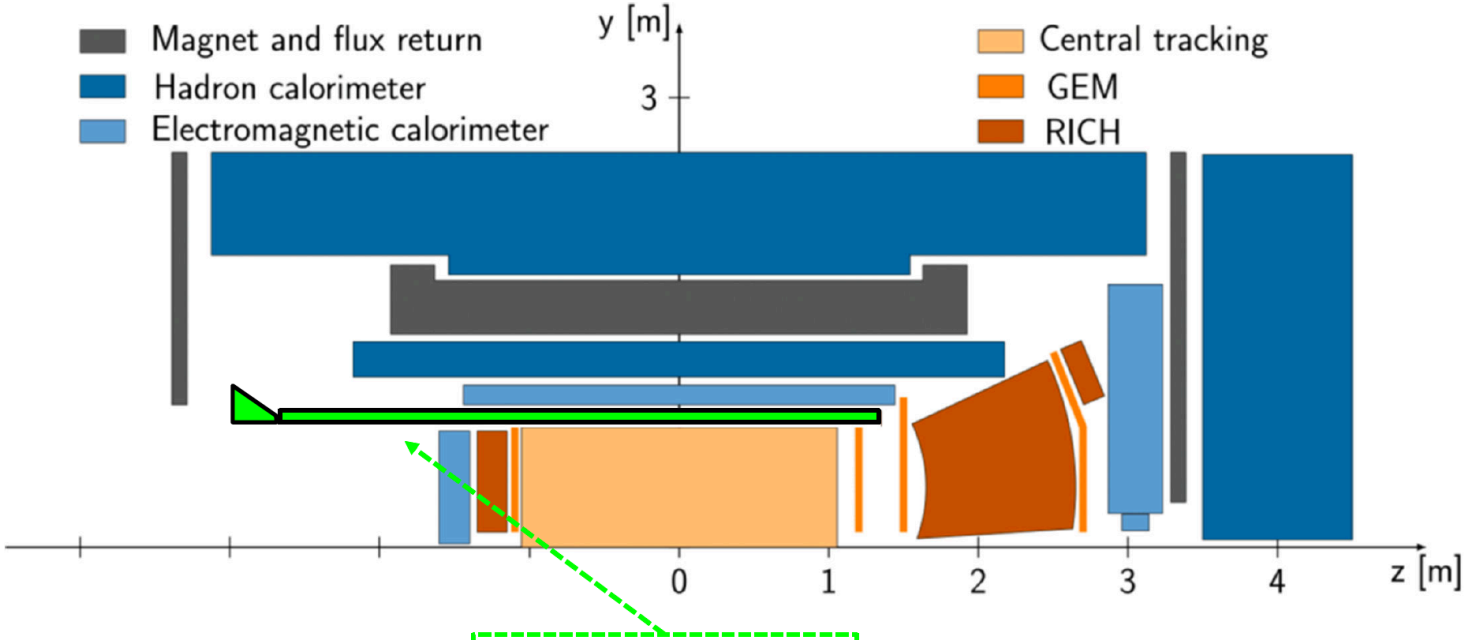


- Obviously looks identical
  - Services and support structure engineering design can start off the same configuration as used in GEANT

DIRC in this scheme

Day-one detector based on sPHENIX, 2018 LoI layout

Geant 4 visualization of particle track and Cherenkov photons in hpDIRC



Cartoon (to scale) of possible hpDIRC integration: example1

- Total bar length 4.2m at 0.85cm radius (4 short bars glued end-to-end)
- Prism expansion volume located outside Barrel EM calorimeter acceptance

- Expansion volume technically can be accommodated in a crack between the e-endcap and the barrel, without breaking the overall logic, but:
  - DIRC will seemingly disturb the "traditional" 4π detector layout, no matter what
  - Engineers need to be involved early enough to think about practicalities

Coding overhead

Excerpt from a modified working calorimetry code:

```
214   // Construct the integration volumes geometry, internally;
215   TFile fin(argv[1]);
216   dynamic_cast<EicToyModel *>(fin.Get("EicToyModel"));
217   eic->Construct();
218   // Populate G4 world by these volumes;
219   eic->PlaceG4Volumes(expHall_phys);
220
221   // Place "MyHCal" tower matrix into the integration volume bubble instead of the world;
222   new G4PVPlacement(0, G4ThreeVector(0, 0, zOffset), myhcal_log, "MyHCal", expHall_log, false, 0);
223   auto hcal_bubble_log = eic->fwd()->get("HCal")->GetG4Volume()->GetLogicalVolume();
224   new G4PVPlacement(0, G4ThreeVector(0, 0,       0), myhcal_log, "MyHCal", hcal_bubble_log, false, 0);
```

This part is taken care of by the framework

- Immediate migration is not mandatory for everybody
  - Integration bubbles can be imported into a framework one by one
- Bubble size (and location) can be polled (*not yet; coming soon*)
  - Parametric detectors can be implemented in a proper way
- If the community prefers to use GDML files instead, so be it (consistency?)