ePIC Software & Computing Meeting at UIC

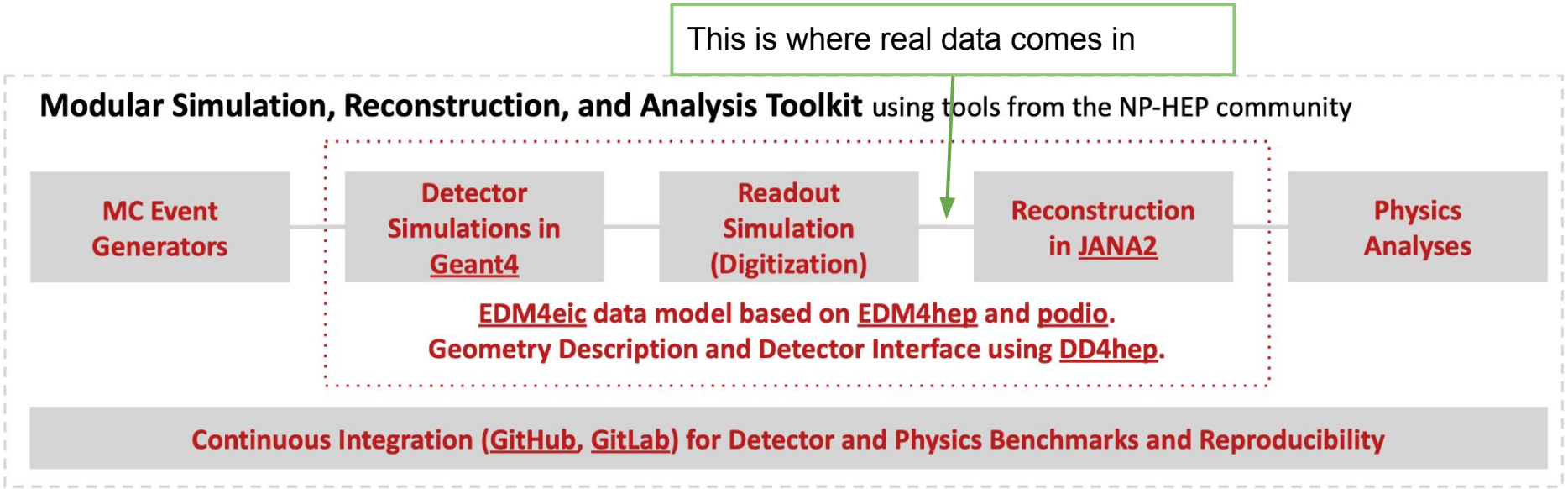
Reconstruction Talking Points

Sylvester Joosten

September 21, 2023

Offline software components

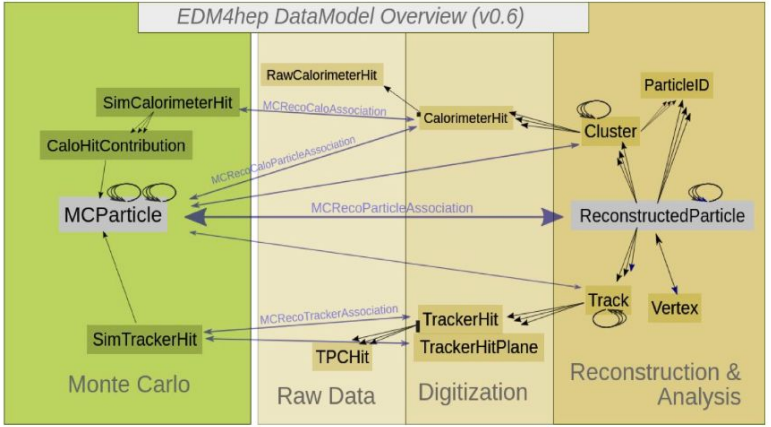
- Using open-source, community-oriented software components from NP-HEP, with focus on software sustainability in selection

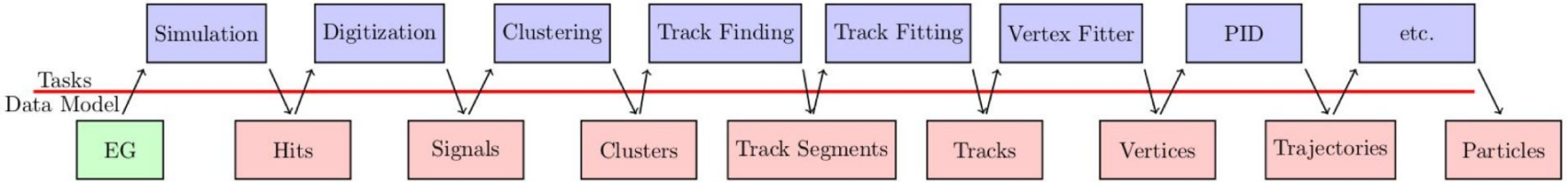


Data driven reconstruction

Use of **standard interfaces** between individual simulation, reconstruction, and analysis tasks **creates modularity** that allows **easy exchange of components**.

- `podio` (github.com/AIDASoft/podio)
  - Text-based definition of flat data models
  - Automatic C++ and Python interfaces
  - Stored inside ROOT files or other formats
- `EDM4hep` (github.com/key4hep/EDM4hep)
  - Designed as a standard for current/future HEP
  - `EDM4eic`: few EIC-specific extension data types
  - Struggled to define in EIC for several years





Issue 1: Transition from DAQ to reconstruction?

- DAQ provides large “frames” with many thousands of bunch crossings
- Reconstruction should be designed to operate on these frames (either single frames or triplets of frames to deal with edge effects)
- Discussion/action items:
  - Introduce readout frames in full simulation (large change to do frame-level reconstruction!)
  - Identify correspondence between DAQ and EDM4eic raw hits
  - How closely should EDM4eic RawHit structures mirror the DAQ structures?

```
edm4eic::RawCalorimeterHit:
  Description: "Raw (digitized) calorimeter hit"
  Author: "W. Armstrong, S. Joosten"
  Members:
    - uint64_t           cellID           // The detector specific (geometrical) cell id.
    - uint64_t           amplitude        // The magnitude of the hit in ADC counts.
      ## @TODO: should we also add integral and time-over-threshold (ToT) here? Or should
      ##        those all be different raw sensor types? Amplitude is
      ##        really not what most calorimetry sensors will give us AFAIK...
    - uint64_t           timeStamp        // Timing in TDC
```

Issue 1b: Geometry - DAQ to DD4hep `cellID`?

- DD4hep detector geometry defines 64-bit `cellID` to encode tree-like location of every readout unit in the detector.
- DAQ will likely (?) define readout identifiers based on readout chain
- Discussion/action items:
  - Translation from DAQ hits to EDM4eic raw hits includes translating the geometry identifier
  - Need to implement DAQ data model at some (future) point in simulation chain
  - Need (time-dependent!) correspondence between both systems

```
<readouts>
  <readout name="VertexBarrelHits">
    <segmentation type="CartesianGridXY" grid_size_x="0.020*mm" grid_size_y="0.020*mm" />
    <id>system:8,layer:4,module:12,sensor:2,x:32:-16,y:-16</id>
  </readout>
</readouts>
```

64 bit cellID

Issue 2: “Frames” and “events” in the reconstruction?

- DAQ “frames” contain thousands of bunch crossings
- Physics analyses expect events (single interactions) as output
- Reconstruction itself does need to run on partial or entire frames for reconstruction (e.g. tracking)
  - This can differ between subsystems (e.g. initial tracking will happen on large amounts of bunch crossings for pattern recognition, calorimeter clustering can/should happen for each bunch crossing separately)
- Discussion/action items:
  - Need to restate reconstruction algorithms to run on entire frames
  - Need to identify how and where we introduce events (likely near the end?)
  - Need to identify how this relates to data flow between algorithms (e.g. tracking to PID, tracking to calorimetry, event building, propagating event-level quantities to reconstruction “loops”, …

Issue 3: Growing our reconstruction flow?

- Reconstruction currently lacks many necessary steps
  - **Phase 1 - Digi**: SimHits → RawHits *present for most systems, but oversimplified (e.g. need charge-sharing for certain silicon detector, more DAQ-specific structures)*
  - **Phase 2 - Hit RC**: RawHits → (Reconstructed)Hits *present but lacking proper calibration infrastructure*
  - **Phase 3 - Independent RC**: (Reconstructed)Hits → Derived independent quantities (Track/Cluster/…) *many systems present and growing*
  - **Phase 4 - Dependent RC**: Derived quantities → secondary derived quantities (e.g. Track + RICH Hits → PID assumptions) *Design for this step currently lacking, e.g. straight from Track + dRICH to ReconstructedParticle*
  - **Phase 5 - Aggregation**: This is currently in the “hacky solution” stage and needs holistic design to aggregate information from *all* detector systems and build events
  - **Phase 6 - Optional Feedback**: Feed aggregate quantities back into phase 3 and/or phase 4 algorithms (reconstruction algorithm loop) for iterative improvements

Issue 3b: Growing our reconstruction service infrastructure?

- Integrate heterogeneous hardware resources (GPUs, FPGAs, …)
- Integrate AI inference infrastructure? We need a collaboration-wide policy/workflow here
- Conditions database!
- … ?

https://github.com/eic/EDM4eic/blob/main/edm4eic.yaml

Issue 4: Growing our data model?

- Data model has many edges that are either undefined, unused, and/or incomplete. Examples:
  - Role of the `Track` data structure?
  - Relation of `Vertex` to `Track`?
  - How do we communicate PID particle assumptions/likelihoods?
  - Can we use the same data model infrastructure for calorimetric PID (e.g. e/π, photon/π0) as we use for hadron PID?
  - Data structures for TOF
  - … (more become apparent as we grow our algorithm library)
- Data model changes can be expensive (except for the many parts that are not yet in use)

Lots to do!

We need to prioritize towards our imminent deliverables (TDR!), while also setting longer-scale milestones