EIC Detector-1 Software Decision

Topic: Data Model

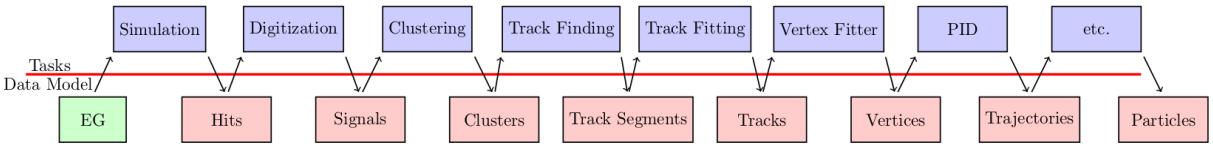
Point of Contact: Whitney Armstrong

Discussion Date(s): June 15, 2022

Meeting Link: https://indico.bnl.gov/event/15642/

Overview:

There is little doubt a good data model is key to all aspects of a simulation, reconstruction, and analysis workflow. The data model sits at the boundaries of each algorithm and each task in the software toolkit. The figure below shows how the data model serves as the defining input and output for various simulation and reconstruction tasks.



To facilitate a clear discussion, it is important to separate the (1) *data model* from the implementation details of (2) *data IO*, (3) *data structures*, and (4) the *data model tools* themselves. The data model (1) is the abstracted concept while the others (2-4) are implementation details. The decision before us today is this: **how should we define the data model (1) so the implementation details (2-4) can proceed?**

This abstraction of the data model allows us to define parts of the data model: “structs”, “components”, “objects”, or even just a single value. For example, a `SimCaloHit` might store a channel number and energy deposited. Furthermore, we can even include in the data model definition the precision for each quantity – in this example, say, a 64-bit integer and double, respectively.

The currently undecided event processing framework presumably will handle the details of data IO (2) and memory layout (3). Data model tooling (4), along with the appropriate framework hooks, should allow for algorithms to leverage cache friendly memory layouts for SIMD processing and other hardware accelerations in a heterogeneous computing environment.

Extending or modifying the data model should be easy and straightforward to use in continuous integration, however, the data model should be carefully maintained as to not break

workflows. This is slightly related to the scope of use for a data model. From a big picture view of things, data model definitions fall into one of three broad scopes or namespaces.

The first scope deals with input and output of geant4 simulation – let's call it `SimHits` since it is typically some form of raw simulation output that will be processed in a subsequent digitization step. The `SimHits` data model is expected to be quite static and see very little change over time.

The second data model scope deals with digitization and reconstruction algorithms and their development. The data model defines the inputs and outputs of the various digitization and reconstruction algorithms and therefore new algorithms often require extending the data model. Furthermore, it is best practice to keep algorithms short, simple, and generic whenever possible. Larger tasks such as tracking and clustering really consist of a chain (DAG) of algorithms. The data model between each algorithm is often transient and ultimately not saved to disk (a decision made within the event processing framework). For example, one might have a two step clustering method where the algorithm-1 determines the number of clusters and groups CaloHits, then the output is used in algorithm-2 to prune hits or reject clusters. A later modification might introduce an intermediate algorithm to refine the grouping with timing information. This example shows that quickly and easily extending the data model facilitates the development of new algorithms which should always be encouraged.

The third scope for the data model deals with physics analysis. Each physics working group could develop their own data models which would facilitate detector or configuration optimizations or refine analysis. This scope also reconnects simulation chains to the physics of the input events to produce performance metrics and plots of different detector and machine configurations.

Future discussions:

- Locality of data model definitions for physics?
- Monolithic data model(s)? EDM4hep + others?
- Input data model (assumed to be framework dependent) 1-to-1 translated (eg hepmc3 to EDM4hep)
-

Requirements:

1. A simple method to unambiguously define a (event) data model (eg PODIO yaml file) supporting the EIC science program.
2. Ability to easily modify or extend the data model (eg PODIO tool which uses yaml file) and include run configuration metadata.
3. A data model definition not specific to one programming language
4. Data model tools that can adapt to the changing landscape of computing hardware (e.g. heterogeneous computing).
5. A data model that does not constrain future IO implementations (2), output file formats or event data store designs.
6. Unconstrained data structure memory layout (3): array-of-structs/struct-of-arrays where tooling provides corresponding views on data.

7. Data model is independent of the chosen event processing framework (TBD), although it is a key ingredient defining the final framework.
8. Needs to support streaming readout data.
9. Data model needs to support full MC truth description.

Other features we might want:

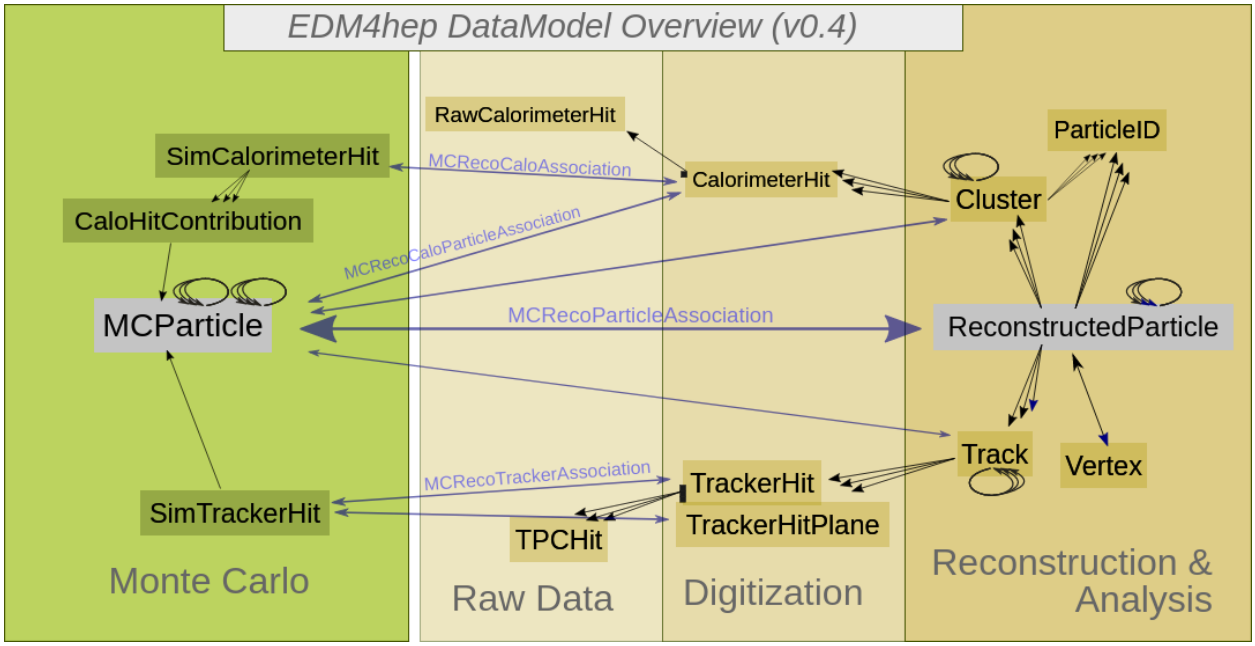
- Explicit Schema evolution mechanism. This may break the simplicity aspect of the data formats discussed in the software principles where simple versioning and release packaging would work all the same.
- If the data model defines input and output data, maybe in the future one could leverage something like C++ Concepts to define constraints on the data types, rather than explicit type names themselves. This could aid in writing more generic algorithms.

References:

- PODIO
  - PODIO: recent developments in the Plain Old Data EDM toolkit, Frank Gaede (DESY), Benedikt Hegner (CERN), Graeme A. Stewart (CERN), 2020
- EDM4hep
  - EDM4hep and podio - The event data model of the Key4hep project and its implementation, 2021
- PODIO/EDM4hep schema evolution:
  - https://www.epj-conferences.org/articles/epjconf/pdf/2021/05/epjconf_chep2021_03026.pdf
  - https://indico.cern.ch/event/1030566/contributions/4327279/attachments/2229135/3776735/PODIO%20Schema%20Evolution.pdf
  - https://indico.cern.ch/event/1030566/
- ACTS EventData
- AIDA2020 WP3: Advanced Software
- EIC Software Statement of Principles
- Eicd - Data model used by Athena simulation framework
- Data Model, (W.Armstrong) 1st EIC Software Consortium Meeting at BNL, OCt. 17, 2016

Options:

1. EDM4hep data model



2. Custom PODIO data model
3. PODIO-style yaml file without committing to data model tooling

Presenters:

- Thomas Madlener - PODIO and EDM4hep

Discussion:

Live notes

- The issue of schema evolution was discussed at length. The PODIO/EDM4hep developers are aware of this potential issue and have used ROOT's built-in mechanism for schema evolution so far.
- Support for streaming readout was emphasized by multiple people. Although, it is not clear to everyone how or to what extent streaming readout data places unique requirements on the data model. The difference between non-streaming data and streaming data are their relation to an "event", the latter requiring time sorting algorithms and an event building framework.
- The EDM4hep model is a good starting point. There is some value to adhering to it as a community standard over time so we should attempt to feed back any necessary changes to the official model.

Convener Summary:

- We will adopt PODIO as the tool for managing the EDM.

We will adopt the EDM4hep Data model as the initial Data Model