Implementation of an HDF5 IO layer for PODIO

Shrey Aryan

August 28, 2019
PODIO, or plain-old-data I/O, is a C++ library to support the creation and handling of data models in particle physics.

It is based on the idea of employing plain-old-data (POD) data structures wherever possible, avoiding deep-object hierarchies and virtual inheritance.

Implementation involves three PODIO layers[1].

- user layer: handles to EDM objects (e.g. Hit) and collection of EDM object handles (e.g. HitCollection).
- object layer: transient objects (e.g. HitObject) handling vector members and references to other objects.
- POD layer: the actual POD data structures holding the persistent information (e.g. HitData).
Hierarchical Data Format (or HDF5) is a generic data format used for scientific data in many fields.

Since it enjoys wide support, it is an ideal format to store PODIO data in.
GSoC Tasks

- Developing a data model description for PODIO in HDF5, reading the PODIO YAML data description in python and outputting HDF5 C++ code.

**Challenges:**
- How to encode deeply nested structs in HDF5?
- Issues with namespace dependencies etc.

- Developing two versions of persistency code for PODIO to serialize a data model into HDF5 format along with tests.

**Challenges:**
- How to construct a map between the data model and its HDF5 code?
- How to write/organize variable-length data in HDF5?

Some attempts at developing persistency code for PODIO to deserialize from HDF5 format into PODIO in-memory layout.
GSoC Tasks

- Developing a data model description for PODIO in HDF5, reading the PODIO YAML data description in python and outputting HDF5 C++ code.
  **Challenges:**
  - How to encode deeply nested structs in HDF5?
  - Issues with namespace dependencies etc.
- Developing two versions of persistency code for PODIO to serialize a data model into HDF5 format along with tests.
  **Challenges:**
  - How to construct a map between the data model and its HDF5 code?
  - How to write/organize variable-length data in HDF5?
Developing a data model description for PODIO in HDF5, reading the PODIO YAML data description in python and outputting HDF5 C++ code.

**Challenges:**
- How to encode deeply nested structs in HDF5?
- Issues with namespace dependencies etc.

Developing two versions of persistency code for PODIO to serialize a data model into HDF5 format along with tests.

**Challenges:**
- How to construct a map between the data model and its HDF5 code?
- How to write/organize variable-length data in HDF5?

Some attempts at developing persistency code for PODIO to deserialize from HDF5 format into PODIO in-memory layout.
F. Desy

*EDM Toolkit - PODIO.*

AIDA2020 WP3 Meeting, Feb 2018