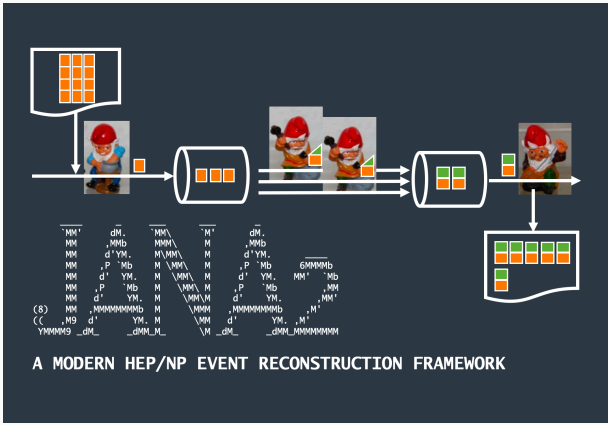
JANA & PODIO Integration: Status update

Nathan Brei
nbrei@jlab.org

ePIC Software Meeting
March 15, 2023



Jefferson Lab

U.S. DEPARTMENT OF ENERGY | Office of Science

JSA

Moving parts

1. Merge multifactories PR into JANA
2. Cut a new JANA release
3. EICrecon uses new JANA release with USE_PODIO enabled
4. EICrecon PR: Use new PODIO source, processor, factories
5. Update EICrecon to use multifactories, object associations as necessary (this is where other people can help)
6. PODIO PR: Add type relations and collection visitor

Details

1. Merge multifactories into JANA
   - These are experimental
   - I'd like to test them more before declaring them finished
   - If you don't use them, they shouldn't affect anything else in JANA
   - Making them available to EICrecon now would increase our velocity
   - Edge case: PODIO objects that are owned by a another collection

Details

2. Cut a new release of JANA
   - Necessary because of issue #202: Segfault at program end due to double-free
   - We are going to include the experimental multifactory support

3. Update EICrecon dependency to use new JANA release
   - We should set USE_PODIO=On
   - This will pull in all of the PODIO features but not make them mandatory
   - The existing EICrecon PODIO code should continue to run, so this change should be transparent to the end users
   - This involves PRs to eic-spack and eic-container

Details

4. EICrecon PR
   - This pulls in the new PODIO functionality
   - We update the data model glue and replace the event source and processor
   - To enable deep PODIO integration in a factory, switch the base class over from JFactoryT to JFactoryPodioT
   - Factories which do not inherit from JFactoryPodioT will probably run just fine, but their data will not be written to the output file
   - However, I expect that using the new JFactoryPodioT will break the old JEventProcessor, so we should migrate all the factories in one go
   - We need to verify that we get the same histograms and data multiplicity before/after, and run valgrind

5. Update EICrecon to use object associations and multifactories
   - This is where other people can help
   - Do whatever needs to happen for the next production run **first**
   - Object associations follow PODIO idioms exactly
   - If you want you can work with PODIO collections directly, though you shouldn't have to
   - For multifactories, see MultifactoryExample under JANA/src/examples
   - Important to manually inspect that associations are intact inside the PODIO file, and to get a clean bill of health from valgrind. PODIO still has problems with collection IDs which could cause memory corruption or segfaults (see issues 379, 381, 382)

6. Improve PODIO

- None of these are necessary for the production run
- datamodel_glue.h currently requires a *`PodioTypeMap<T>`* template trait in order to express that *`Hit`* is always stored in *`HitCollection`*. This should be a *`using`* statement on the *`Hit`* class. Fixing this will remove an unpleasant quirk where the datamodel glue header file always has to be included before JFactoryPodioT.h and its ilk.
- In order generically insert a Frame full of type-erased PODIO *`CollectionBase`* into a set of typed *`JFactoryT<T>`*, we generate a *`PodioCollectionVisit`* helper class which reverses the type-erasure for all classes in the data model and works with user-defined *`Visitor`* classes similar to *`std::visit`*. This is generated in the data model glue but ideally would be generated by PODIO itself.

Thank you! That is all.