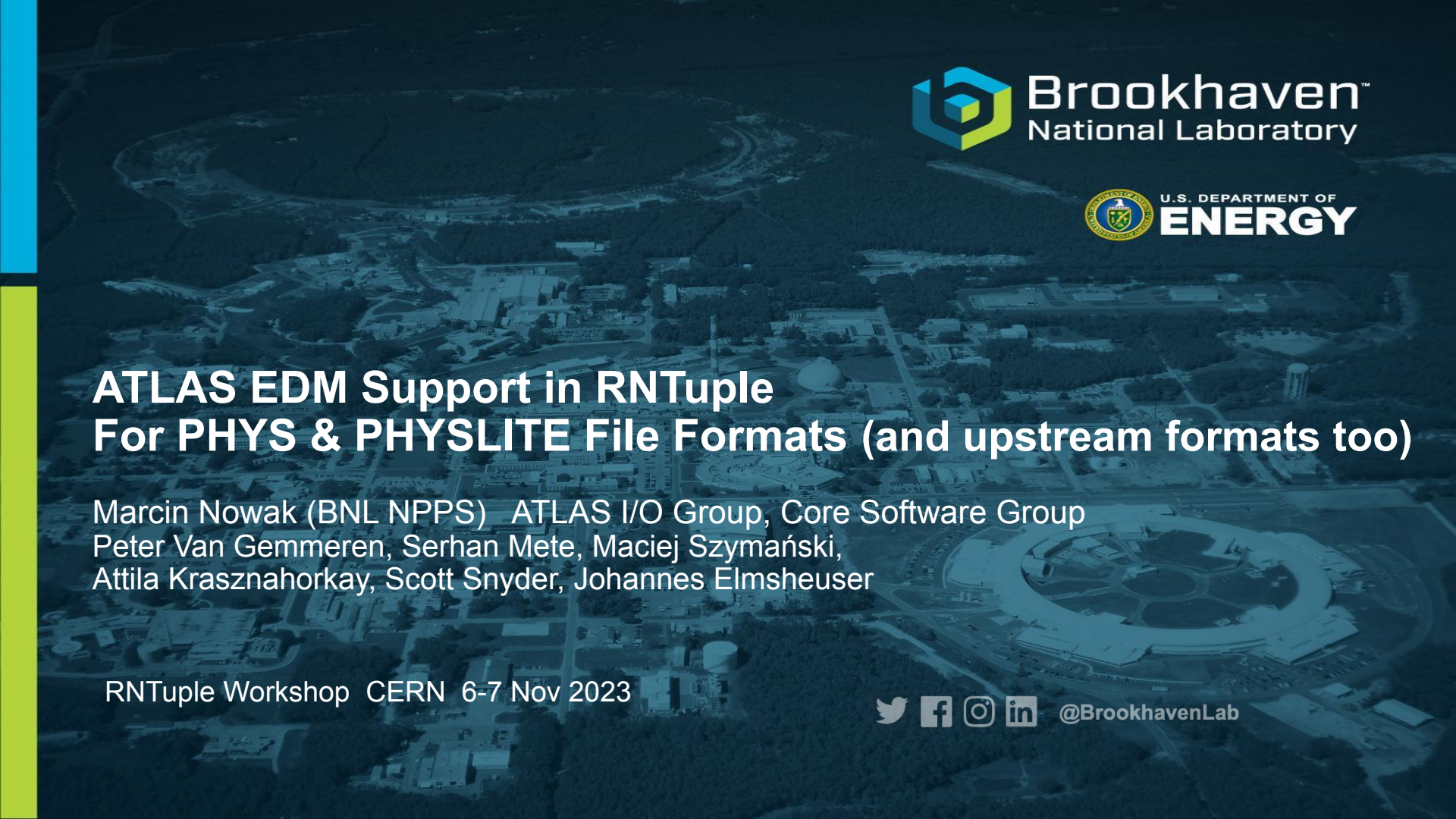

ATLAS EDM Support in RNTuple
For PHYS & PHYSLITE File Formats (and upstream formats too)

Marcin Nowak (BNL NPPS) ATLAS I/O Group, Core Software Group
Peter Van Gemmeren, Serhan Mete, Maciej Szymański,
Attila Krasznahorkay, Scott Snyder, Johannes Elmsheuser

RNTuple Workshop CERN 6-7 Nov 2023

This Talk

Athena Event Data Model support in RNTuple

- Athena - this presentation concerns ATLAS production framework Athena
  - not addressing Analysis usage here
- Event Data Model - focus on handling of the Event data in Athena
  - A separate MetaData talk by Maciej Szymański after lunch today
- And after that:
  - Infrastructure for production workflows presentation by Serhan Mete

ATLAS EDM

- Single Event transient data consists of objects in the Event Store (StoreGate)
  - Each object has unique name
  - Most of them are collections
  - No regular C pointers going out from the object to the other objects in SG
    - Sometimes there are internal pointers
  - References between objects in SG represented by **ElementLinks**
    - Which we persistify as well
- EDM can (and does) change over time - **Schema Evolution**
- At the beginning (Run 1) support for complex EDMs was not as good in ROOT as today, so we used T/P separation to hide the complexity
  - Also supporting schema evolution at the T/P level
- RNTuple would not need any extra features at that time

xAOD Data Model

Since Run 2 the EDM for AOD and DAOD data (PHYS, PHYSLITE) is based on xAOD containers

- Each xAOD type has its own xAOD container
  - Single type collections
- Containers store xAOD type attributes member-wise in external storage arrays
  - This external storage supports adding new attributes at any time at runtime (**dynamic attributes**)
- Containers appear to the user as std::vectors of objects, but internally they are implemented as vectors of pointers - DataVector
  - User collection type

No T/P separation for xAOD - TTree can now handle much more complex EDMs

  - And xAOD simplified the data model in some respects

Higher bar for RNTuple!

ATLAS EDM - Requirements for the I/O

List of features that are currently used in TTree-based PHYS/PHYSLITE formats (using genreflex with rules defined in selection.xml):

- Nested vectors support - probably up to 3 levels (xAOD attribute storage adds 1 extra nesting level)
- ‘After read’ callbacks on selected classes
  - Objects initialization after reading, e.g. for transient data members
    - Mainly ElementLink, also some other classes, schema evolution
- Adding new RNTuple columns after writing the first Event - for the dynamic attributes (AKA RNTuple schema extension)
- DataVector support
  - Transparently handling vectors of pointers
- Some schema evolution support
  - Similar to what TTree provides

RNTuple Status in ATLAS EDM Support

After 2 years of productive cooperation:

- Nested vectors - OK
- Adding new columns after the first event - OK
- After-read callbacks - OK - using existing configuration from selection.xml(!)

```
<read sourceClass="ElementLinkBase" version="[1-]"
      targetClass="ElementLinkBase" source="" target="" >
   <![CDATA[ if(newObj) newObj->toTransient(); ]]>
</read>
```

- Schema Evolution support - OK - using current read rules from selection.xml

```
<read sourceClass="CLHEP::Hep3Vector"
      source="double dx; double dy; double dz;"       version="[1-]"
      targetClass="Eigen::Matrix<double,3,1,0,3,1>"   target=""
      include="GeoPrimitives/CLHEPtoEigenConverter.h">
 <![CDATA[
     *newObj = Amg::Hep3VectorToEigen (*(CLHEP::Hep3Vector*) (oldObj->GetObject()));
  ]]>
</read>
```

RNTuple Status in ATLAS EDM Support (2)

DataVector support - OK - using existing CollectionProxy definitions in Athena

- RNTuple uses `TDVCollectionProxy` to access `DataVector` elements:
  - Direct access to the elements through the Proxy ensures there are no unnecessary copy operations performed

```
#define ADD_DV_PROXY( TYPE )                                          \
   int register_##TYPE##_CollectionProxy() {                          \
     xAOD::AddDVProxy::add< TYPE >( ROOT::GenerateInitInstance( ( TYPE* ) 0x0 ) ); \
     return 1;                                                        \
   }                                                                  \
   static int _R__UNIQUE_( dummy_##TYPE##_Var ) =  register_##TYPE##_CollectionProxy();   \
   R__UseDummy( _R__UNIQUE_( dummy_##TYPE##_Var ) )

namespace xAOD: class AddDVProxy:
template < typename T >     static void add( ROOT::TGenericClassInfo* clInfo ) {
      // Create the collection proxy instance:
     TDVCollectionProxy* proxy = new TDVCollectionProxy( ClassName< T >::name().c_str() );
     proxy->SetResize( Helper< T >::resize );
     clInfo->AdoptCollectionProxy( proxy );
     return;
 }
```

EDM - Recap of Encountered Unforeseen Issues

Some things were not included in the original requirement list because they were taken for granted

- Typedefs - were not recognized at the beginning, now working OK

- Long long -> gets converted into int64_t which is now defined by gcc as ‘long’ - so effectively a type change (different typeid) when reading back
  - Considered aRNTuple bug
  - We performed schema evolution to get rid of long longs from the EDM by converting them to int64_t when it mattered
    - Considering long long as not defined precisely enough for persistency

PHYS/PHYSLITE EDM Support Ready

All the EDM-related requirements are working

Athena can write regular PHYS and PHYSLITE files

- Major milestone reached

Tested running real production jobs

- We build Athena nightly on top of LCGDEV3
  - Nightly build with weekly rotation, can be unstable sometimes
    - Thank you Johannes for looking after the build!
  - Contains ROOT HEAD version and the latest RNTuple code
  - Installed on CVMFS
  - Can be set up with ATLAS Setup command like every other Athena release

Beyond the Original Requirements

Given the existing feature set, can we use RNTuple with the other ATLAS data products?

- AOD, EVNT, HITS - OK
  - (std::set used in EventInfo)
- RDO - should be also OK, but one DataVector is failing 😕
- ESD:
  - ‘Naked’ pointers present
  - Could be changed into unique_ptr (will TTree automatic schema evolution handle it?)
    - Actual T/P schema evolution possible but cumbersome
    - Could we pretend they are ‘unique’?

I am fairly sure that with some effort we can get rid of std::set and the pointers, but we will not complain if they are simply supported, as long as the cost is acceptable

  - Looking forward to the discussion at the workshop

RNTuple API for Athena

Quick recap of how RNTuple APIs used now in the Athena RNTuple prototype (to make sure it does not disappear)

- Athena I/O layer deals with void* pointers + TClass type information
  - No profits from the templated API
- Every Event consist of freshly allocated object
  - When writing, memory addresses change and need to be set again every Event
- Reading is on-demand, SG object at a time, xAOD dynamic attribute at a time
  - Into existing object (overwrite)
  - Or asking RNTuple to create a new object and pass ownership to Athena
    - void* pointer being cast to the actual type
- Event Store is cleared between events
  - Athena deletes also the objects created by ROOT

Summary

- ATLAS requirements for PHYS and PHYSLITE format support by RNTuple are all fulfilled
  - Many thanks!
  - Athena can write RNTuple format following runtime configuration on per-file basis
    - Reading is fully transparent
  - Possible because we are able to use the same EDM for TTree and RNTuple based storage (!)
  - The feature set is already good enough for all other Athena data products except ESD (pointers) and RDO (unexpected DataVector problem)
    - It's still a better starting situation that we expected
  - Still - It is just a prototype and not any near production mode - now that the functionality is there, we are only beginning scaling tests (including MT) and performance tests
    - see the presentation by Serhan later today

Summary Cont.

We are confident that with continued development ATLAS can write all production event data to RNTuple. This will require:

- Fixing failing DataVector in RDO
  - maybe just T/P separate?
- Eliminating bare pointers issue in ESD
  - Schema evolve to unique:ptr?
  - Direct support in RNTuple?
- Continued support for std::set by RNTuple or replacing one occurrence in AOD on the ATLAS side
  - Schema evolution + std::set functionality emulation - possible (but maybe not necessary?)
  - Understand the cost of support in RNTuple
  - Discuss with RNTuple and xAOD experts (Attila et al).