ATLAS I/O Group News

NPPS Group meeting 25/03/2026

Marcin Nowak, BNL NPPS
ATLAS I/O Group & Core Software Group

Athena I/O

- Athena I/O components
  - APR (POOL)
  - T/P conversion layer
  - AthenaPool converters
  - AthenaPool conversion service
    - With SharedWriter
- Athena I/O Concepts
  - DataHeader
  - Event Collections
  - In-file metadata
- RNTuple migration

APR - Integration, Cleanup, Refactoring

- APR - originally common LCG POOL project providing object database - like interface to a chosen storage technology: ROOT TTree
- Integrated into Athena when ATLAS was the only user left
  - Always a little different (not based on the Gaudi component model)
  - Very generic
  - Some subpackages dropped already during the ‘takeover’ - e.g. object storage in relational databases
  - Integration by encapsulation
- Since then - several cleanup campaigns
  - Dropped all remaining relational features (collections) and dependency on CORAL, switched to Gaudi FileCatalog, some internal code cleanup
- Currently in middle of the largest cleanup and integration campaign
  - Aimed for HL-LHC, taking advantage of the long shutdown
  - Leaner code easier to maintain and improve

APR Code Reduction

APR subpackages, late 2025:

- POOLCore
- PersistencySvc
- StorageSvc
- RootStorageSvc
- CollectionBase
- CollectionUtilities
- ImplicitCollection
- RootCollection
- FileCatalog
- APRTests

APR subpackages now:

- PersistencySvc
- StorageSvc
- RootStorageSvc
- CollectionSvc
- RootCollection
- APRTests

Nearly half of the subpackages removed

50% of lines of code removed

All APR features used by Athena still present

- Started a validation campaign to assess performance

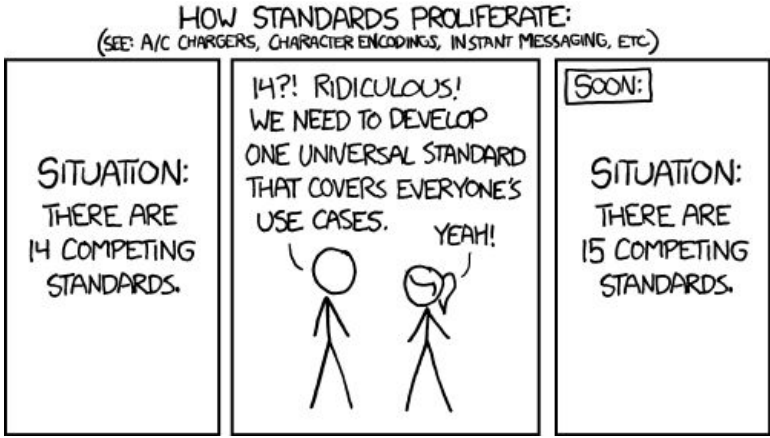
Slowing down now

DataHeader, Event Collections, Navigation

- DataHeader - Athena specific, per event navigational information for locating all objects that belong to the event (contains References, a.k.a. Tokens)
  - In std::string form - as technology independent representation
  - Long strings, similar to each other - can be compacted used dictionary coding on writing, in the converters
  - Certain workflows can still produce noticeable overhead (worst noticed: 5% filesize)
  - Tighter integration of APR into Athena opens possibilities to delegate the Token encoding to the APR StorageSvc, allowing a single dictionary per file - TODO his year
- Event Collections - small tuples containing Event Reference plus arbitrary metadata
    - Every input to Athena is treated as a Collection - even files
  - Recently merged 4 APR collection subpackages into 2
  - Fixed number of References per entry to 1
  - Currently 1 production use of Collections - they may be dropped in the future

In-file Metadata

- We now have several metadata objects that contain partially overlapping information
  - Result of trying to invent a new metadata standard every time when starting a new LHC Run
  - Some metadata objects migrated from per-event into per-IOV
- Aiming to drop legacy objects and start ‘fresh’ for HL-LHC
  - Object Registry (replacing DataHeader)
  - Event Table (replacing index)
  - File Summary (everything else)
  - The dictionary used in tokenization of References in DataHeaders would also be a file-global metadata



Moving RNTuple to Production

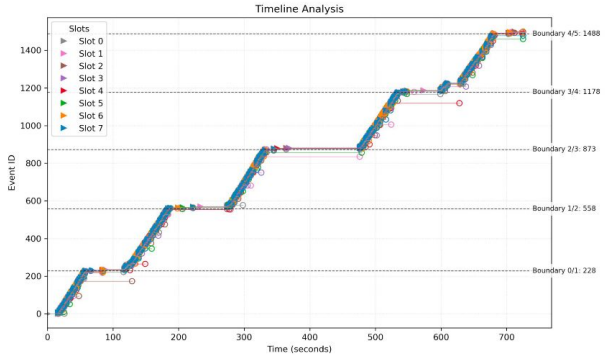
- Athena is now fully adapted to ROOT RNTuple
  - ‘Announced’ at the last CHEP already
  - Supporting all Athena production EDMs and workflows*
  - Transparent reading, single job switch for selecting technology when writing

- Currently testing performance: CPU and RAM
  - And finding issues

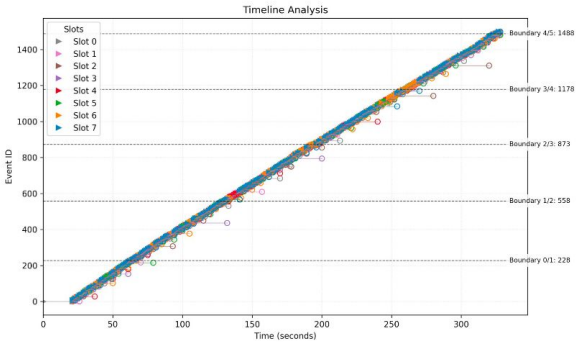
RNTuple Read Performance

- AthenaMT read cache trashing on cluster boundaries and unofficial fix

- **Job details:**
  - **Dataset** : 601229.PhPy8EG_A14_ttbar_hdamp258p75_SingleLep (on SSD)
  - **Input/Ouput** : **RNTuple AOD** (LZMA) → **TTree DAOD_PHYS** (ZSTD)
  - **Events** : 1500 (out of 10000)
  - **Threads** : 1 and 8
  - **Setup** : Athena, 25.0.52



Before



After

2

RAM Usage in AOD -> PHYS Production (27K Events)

	TTree->TTree	RNT->TTree	RNT->RNT
Max Vmem:	4.96 GB	7.67 GB	7.90 GB
Max Rss:	4.31 GB	6.46 GB	6.85 GB
Max Pss:	4.09 GB	6.45 GB	6.63 GB
Leak estimate			
per event Vmem:	5.00 KB	17.00 KB	24.00 KB
per event Pss:	-11.00 KB	8.00 KB	6.00 KB

Event	Vmem [kB]	Rss [kB]	Pss [kB]]
1	3217224	2647624	2413442
49	4922248	4324260	4089126
27671	5199720	3959548	3938549

Event	Vmem [kB]	Rss [kB]	Pss [kB]]
1	4140164	3570044	3333415
17	6747768	5715180	5702967
27553	8039096	6758480	6747092

Investigating Object Stores

- ROOT has a native RNTuple I/O interface implementation for DAOS
  - Was developed and demonstrated a couple of years ago when CERN OpenLab had a test DAOS cluster
  - Open source implementation of an ObjectStore on Linux: https://github.com/daos-stack/daos
  - DAOD is available at ANL (Aurora)
    - received access (through Robert Latham)
  - By default ROOT builds do not have DAOS enabled - requires building on a node with DAOS client libraries present
    - May require rebuilding the entire Athena stack
    - On my TODO list
- ROOT mentioned plans for providing S3 integration (~ end of 2026?)