DUNE Data Management Presentation at FIFE mini-workshop

Steven Timm

FIFE mini data management workshop

January 17, 2024

Data Roles in DUNE

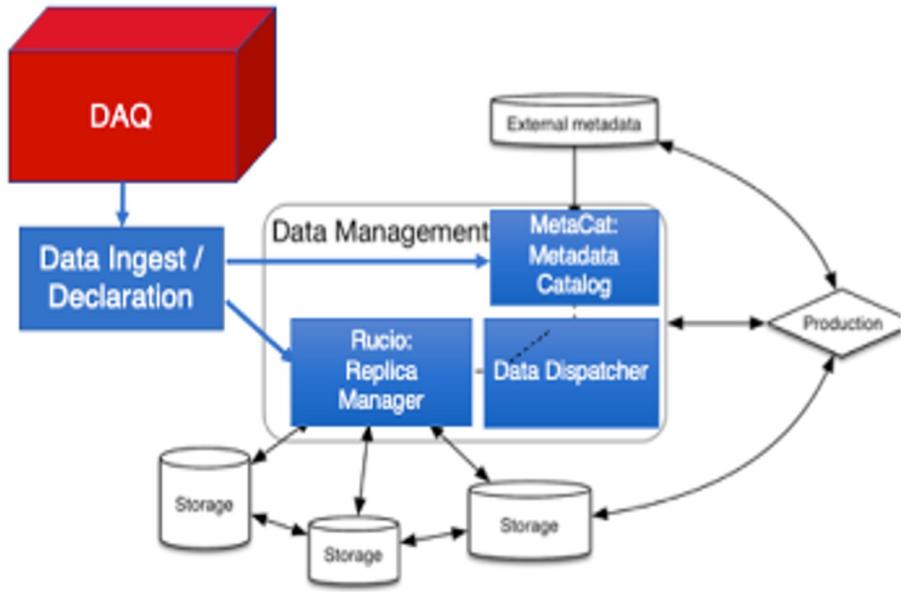
- Data Management Development and Operations
  - Steven Timm (FNAL) and Doug Benjamin (BNL) co-leads
  - Plus developers Igor Mandrichenko (FNAL), James Perry (Edinburgh)
  - Monitoring lead Wenlong Yuan (Edinburgh)
  - Rucio Service operations. Brandon White (FNAL)
  - Goal of this group to design, implement, integrate and operate the data management services
- Data Collections Manager. Aaron Higuera (Rice)
  - Deals with physics groups as to which data should be where
  - Creates public physics data sets.
- Data Access Taskforce A. Furmanski (UMN), J. Hays (QMUL), et al .
  - Responsible for ensuring long-term public access to final physics data sets.

[Proto]DUNE data management project to date

- Fermilab was asked to take the lead to get ProtoDUNE data back from CERN to Fermilab.
  - S. Timm, I. Mandrichenko, R. Illingworth
- First round in 2018 used Fermi-FTS and a new product called FTS-Light
  - (needed because CERN wanted to keep running even if SAM was down).
  - Still used SAM for metadata and file location.
  - It was agreed from the beginning though that SAM would be replaced.
- In current round:
  - FTS-Light -> Ingest Daemon (uses FTS3 for file transport)
  - Fermi FTS -> Declaration Daemon (declares files to Metacat, Rucio, and SAM)

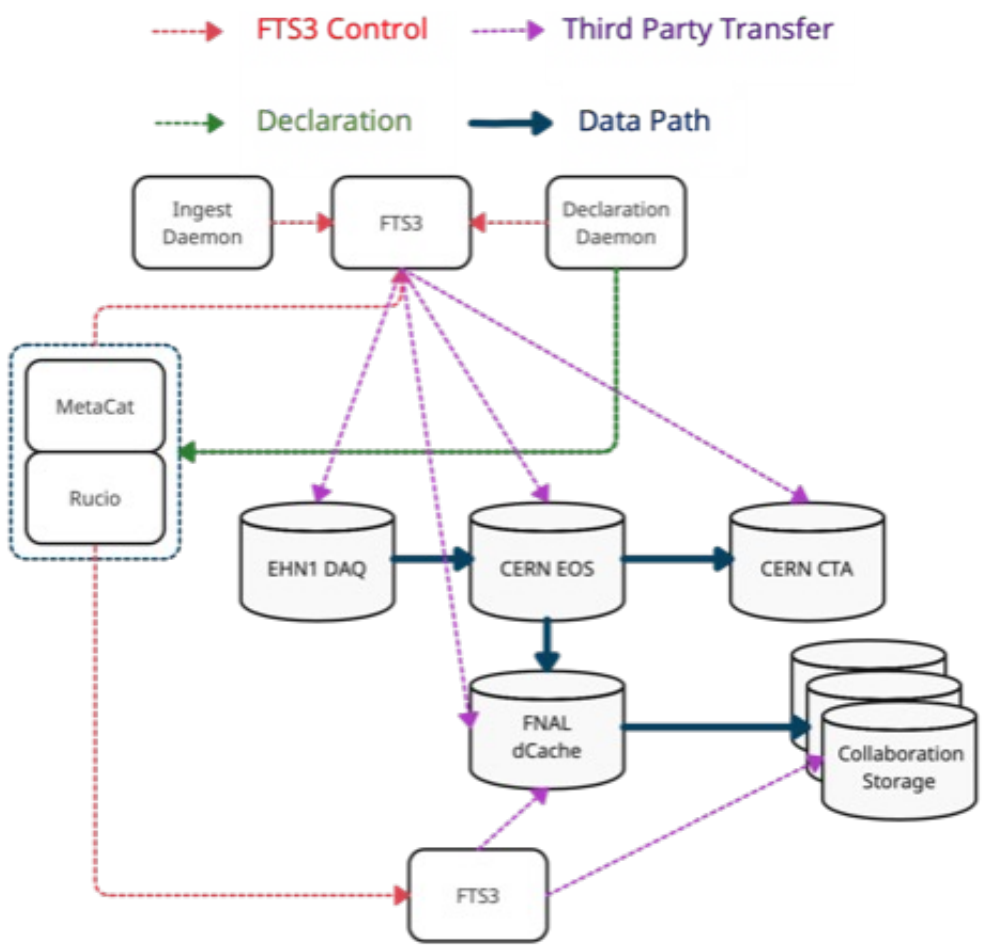
[Proto]DUNE data management currently

- Currently we have Ingest and Declaration daemons harvesting raw data from CERN and from 2x2-Minerva
- Data locations from Rucio
- Metadata in MetaCat
- Initial design of project also included Data Dispatcher
  - Thought DUNE would need this for an expected 2021 second run of ProtoDUNE
- Turns out DUNE Workflow group got the main workflow system JustIN done about the same time.
- DeclaD also deals with output of legacy POMS-submitted offline production.
- JustIN submitted jobs declare to MetaCat and Rucio directly without a daemon or dropbox.



Currently we are back-declaring input files from MetaCat/Rucio to SAM.
Works great except when you forget to do it.
We have not finished forward-declaring all the previous SAM locations to Rucio.

Data Pipeline Diagram



- Legacy file transfer system replaced by Ingest Daemon and Declaration Daemon
  - Ingest daemon brings files from experimental systems to dropbox
    - Can operate without connection to Fermilab.
    - All transfers done via FTS3
  - Declaration daemon declares them to MetaCat and Rucio and makes rules to get them to final destinations.
- 2 copies of raw data on tape
- 1 copy of sim/reco on tape
- 2 copies of sim/reco on disk distributed across global storage elements.

Legacy stuff with SAM support on which DUNE still depends (in priority order)

- POMS and associated fife_launch scripts. (Production group likes POMS and wants to interface it with justIN).
- Userspace client (samweb list-file-and-locations and similar). Do we want to expose Rucio clients to end-users? Still discussing.
- Sam4Users
- IFDH
  - Has many SAM-aware handles
- ART
  - getNextFile function (believe this relies on the ifdh functions above)
  - Embedding metadata in root files. [No such thing in the hdf5 raw data files from DUNE]
- Monitoring—Neither DD or JustIN has the ElasticSearch monitoring backend that SAM has got.
- Prestaging! (Both DD and JustIN utilize pinning to do it. But what's solution for interactive? At moment Rucio tape cache is read-all and allows anyone to stage however much they want.)
- Jobsub-lite -- Can automatically run over a SAM dataset.
- DUNE wants to make SAM read-only by mid-2024 and dispense with it altogether by mid-2025.

Mistakes DUNE has made so far:

- We waited too long to port our SAM DB into Rucio. It's still not done and could still take several more FTE-months of effort.
- All new stuff is going into Rucio but about half of our older files still aren't there.
- Lots of exceptions, one-off cases at Fermilab and CERN.
- We also made too many compromises with DAQ groups to get them to buy into our system in terms of non standard file locations which are now coming back to bite us.
- Mistake we haven't made yet:
  - Haven't decided if we will request to do as CMS does and have completely separate Disk-backed and tape-backed storage elements at Fermilab.
  - But in any case our tape-backed storage elements have what is called a "non-deterministic" path which is generated from the metadata of the file and has English-readable directories in the path name. Our disk storage elements elsewhere have a hashed directory structure.

justIN Workflow Basics:

- See https://justin.dune.hep.ac.uk/docs/ for more info
- User submits a Request to justIN system which specifies (among other things):
  - A set of input files to run over, as a MetaCat MQL query. (or specify the MonteCarlo)
  - A predefined jobscript to run on each one
  - The distance from which you are willing to read the files
  - Memory and disk usage
  - Output file patterns and destinations
  - Requests can have more than one Stage and run a different jobscript in each Stage

justIN Workflow Basics 2

- The justIN system queries Rucio to find all the available locations of the files in the Request
- The justIN system submits a number of “Generic jobs” proportionate to the number of jobs needed in the Request. These run with permissions of the production user
- The DUNE global pool generates glideins based on the number of idle generic jobs in the queue.
- When the generic job starts at a site, it will query the justIN server to determine the best request to work from (there could be several running at once) and the best file from within that request to work on.
- The generic job then runs the user jobscript. When it exits
  - The file is declared to MetaCat
  - The file is uploaded to the “nearest” Rucio storage element with Rucio upload.
  - The file is declared to be a member of a Rucio dataset which has a Rucio rule to bring it back to a central site, usually Fermilab.

Error handling:

- We are not perfect yet!
- If jobscript gets aborted, no metadata is made, no upload is done,
  - justIN will detect that it never got a status and eventually try to process the same file again
- If first upload attempt fails—we retry up to N different RSEs
- Have seen cases in times of very overloaded Rucio server where it fails altogether.
- Each file is offered six chances at processing before being marked “failed”
- Recent production discovered some bugs in the error handling process (if the last file upload of N is success all output is marked successful).
- Timestamps in our file names should take care of most clashes—if the file is submitted it gets a second timestamp the second time around and thus won’t overwrite.
- It should not be possible to get the same file name produced twice *but* we have seen it happen and don’t quite understand how.

Legacy POMS/jobsub processing

- DUNE also still does some legacy POMS/jobsub submission
- In this case the metadata file is produced along with the data file on the worker node and then copied to a dropbox directory tree.
- The dropbox directory tree is traversed by the Declaration Daemon which declares the files to SAM, MetaCat and Rucio. The declaration daemon does not use Rucio upload, rather it calculates where Rucio would put the file and just puts it there and then declares the replica.
- In all cases both justIN and legacy we have to make the first Rucio hop to a disk-only RSE.. We have 1PB on persistent dCache which is disk-only for a landing space at Fermilab.

Big Challenges Remaining

- Have at least four different current data formats right now that we know about, 2 flavors of HDF5 (far and near detector) plus whatever Minerva is writing plus Root.
- Want to make a unified metadata generator for all of them rather than the ad-hoc set of scripts we have now.
- In HDF5 there's no facility (yet) for embedding metadata as there was in ART-Root. Plus we're designing a new framework that is in early stages.
- In far detector we will be dealing with trigger records that span across multiple physical files, mechanism to keep track of those trigger records yet to be finalized. (in case of Supernova the readout will be split in both space and time).
- RNTuple—could be a game changer, may require object stores to store it.
- The smart ML guys who will probably come up with I/O loads previously unseen in HEP experiments before all is said and done.