Metadata Considerations for ATLAS Distributed Computing

David Malon
malon@anl.gov
Argonne National Laboratory

ADC Technical Interchange Meeting
University of Tokyo
16 May 2013

Introduction

- Metadata encompass many things—luminosity information is metadata, data quality information is metadata, trigger configurations are metadata, …
- Developments are needed on a number of fronts, not all of which are directly relevant to ADC
  - And expert developer effort (and effort from non-experts willing to learn) is sorely lacking!
- A metadata “review” is just beginning
  - Quite limited in scope, but much of that scope is germane to ADC—more about this later
- But several other core software metadata developments have implications for ADC, and I will mention a few of these as time permits
- Given the nature of this meeting and the limited time, focus should probably be on metadata considerations for prodsys2 development

The metadata “review” mandate from the OAB (“Metadata developments during LS1”)

“This group is in charge of defining improvements on some of the metadata treatment during LS1, based on run 1 experience.

Topics to be addressed include:

- nomenclature: revisit nomenclature conventions and update the nomenclature document, incorporating experience from group production, reviewing each constituent field, including in particular the usage of the physics short name field, the proliferation of data types, the length of configuration (AMI) tags, and expected overall length limitations
- production tag definition: move toward a single tag definition system
- metadata integration with the evolving ProdSys and DDM infrastructure
- job information and other metadata sent back to AMI
- possible extension of AMI search interface (for instance, reverse search of production tag)
- better integration of the ATLAS geometry definitions in the metadata infrastructure.”

Three areas of the mandate, really

1. Nomenclature and related issues (the first bullet)
2. Opportunities for metadata and related infrastructure integration, in areas strongly related to ADC (the next three bullets)
3. Feature requests for AMI (the last two bullets)

- Without preempting the review team, perhaps the first two categories might profitably be discussed a bit at this workshop as possible input to the review process
  - The third, AMI feature requests, are probably better discussed in a different forum
- I will at least introduce some of the issues
  - And interject my own opinion in places, in the interests of advancing the agenda
  - And we are not starting from zero on every point of the charge
  - And two ADC big bosses have elected to be on the review team in any case

Nomenclature

- "revisit nomenclature conventions and update the nomenclature document"
  - Fine. Just part of the job.
- "incorporating experience from group production"
  - Good idea. Many of our nomenclature issues arise downstream of collaboration-wide production.
- "reviewing each constituent field"
  - Of course. We should be thorough.
- "including in particular the usage of the physics short name field"
  - In much better shape since an April 2012 half-day workshop with the simulation folks (explicit agreed character limit + documented conventions for physics short—perhaps the OAB was unaware of this), but perhaps we can do better.
- "the proliferation of data types"
  - Definitely. More on this later.
- "the length of configuration (AMI) tags"
  - A specific proposal was made several months ago jointly by Data Preparation Coordination and metadata infrastructure providers, but it was put on hold pending the review.
- "and expected overall length limitations"
  - Good start to this in mid-2012, with ADC input from Simone

On proliferation of data types

- Experts should correct my misimpressions, but
- It seems that much of the proliferation is needless
  - by which I mean that when a job produces several output files, they must of course be distinguished by *name*, but this does not mean they must be distinguished by *type*—PROVIDED we have naming conventions that support this
- And this proliferation causes needless complications for transforms and perhaps for other components as well
- And even when files are not produced the same job, some of our type distinctions may be needless given appropriate name distinctions
  - And we might want to revisit our D and M conventions in any case (DAODM_ and so on)

Note to self

- There may be more difficult nomenclature issues not explicitly itemized in the charge.
  - Example: Consistent file naming between Tier 0 and distributed production
- The review team should decide whether to consider them.

"Production tag definition: move toward a single tag definition system"

- Configuration tags ("AMI" tags) are, according to experts, not defined in quite the same way in Tier 0 and distributed production.
- This is not an area in which I have had any involvement, so I will try to characterize my understanding of the differences orally rather than in writing
  - And someone will no doubt correct me
  - And someone else will tell me there are good reasons for the differences
  - But my impression is that the Tier 0 way is preferred by people who are experts, if it could indeed be adopted in distributed production
  - And that a common approach would be a Good Thing
- More generally, some of us should (re)consider what is and is not meant to be conveyed by an AMI tag, and whether we are accomplishing this
  - Related comment from Graeme: "Should not need to query prodsys to use a prodsys-defined AMI tag"
- A reconstruction job can in principle (and is often in practice) configured via an AMI tag alone
- Loosely related: I presume the legacy of positional and non-positional parameters and treatment of defaults in the black magic of configuring tasks, a headache for several people, is on the list of items being addressed in Prodsys2 development

“Metadata integration with the evolving ProdSys and DDM infrastructure”

- This is the challenging and interesting part, I think.
- New opportunities arise with the redefinition of task request infrastructure, for example—more on this in a later slide
- And with the advent of Rucio, questions of what metadata belong in Rucio versus AMI should be addressed
  - Operational versus semantic (production-related versus physics-related)?
    - This is a natural candidate with a data management system intended to be agnostic regarding the content of the data it manages
    - But {key, value} pairs allow the possibility of semantic metadata as user-maintained value-added extensions, of course
  - And there are other possibilities
  - We may choose to put some metadata in both—there are always borderline cases--but in such cases, we should determine which is authoritative and how to maintain consistency

Job information and other metadata sent back to AMI

- Long history here, of truncated returned metadata and so on—much better now
- Transform team has done a good job of cleaning up job-produced metadata files
  - Adding structure, adding content, reducing redundancy, better supporting multistep processing, …
  - Agreement to record such metadata in JSON files
- Such files are being produced but not yet being returned
- Return machinery should of course be clearly defined and routinely supported
- The topic is, of course, potentially broader than what a job returns to AMI
  - There are many actual and prospective flavors of returned metadata, and metadata destinations
    - Rucio, prodsys databases, AMI, …, event index repositories perhaps?, …
  - commonality of infrastructure, where possible?
- (Not for ADC): There have also been discussions of interfaces by which a job might emit a metadatum (beyond the standard information provided by the transform layer itself). More core software work and thinking is needed here.

Dataset-level metadata, in-file metadata, and task definition

- ATLAS does an impressive amount of file peeking
  - In pre-execute() transform steps, in Athena initialization, sometimes (though this is improving) when the first event is read, …
- Some of this peeking (in multiple passes) is for the purpose of job configuration
- Conjecture: Some job configuration done by file peeking could instead be done if dataset-level metadata could be used for configuration
  - Conjecture: For many kinds of datasets, configuration should be the same for all files in the dataset
    - And hence should be feasible in advance—with dataset-level metadata
    - IOV information may be different, but occurs in a different level of configuration
    - If one processes a dataset of 1000 files via 1000 jobs and one of the job configurations is different, is this more likely to be correct, or an error?
- In our next generation of infrastructure, could we make use of dataset-level metadata at the task request or task-to-job mapping step?
  - Improving performance and consistency by reducing file peeking?

Dataset-level metadata, and externalized file-level metadata

- Much dataset-level metadata already exists and resides in repositories and is queried by tools not usually directly connected to production processing
  - AMI, COMA, …, data quality and data-preparation-supported tools
- We might think about how we might support and take advantage of dataset-level metadata more generally—task definition is just one example
- Another example: Automatic dataset-level integrity checks (more than the union of job integrity checks?)
- A desideratum:
  - If you have a file, you should be able to determine what is in it and how to read it without reference to an external database;
  - Conversely, if you cannot reach a file, you should be able to determine what you are missing
- We plan to improve the infrastructure in support of the latter
  - Which lumi blocks? (Which events?)
  - We know the latter (and hence, implicitly, the former) already via the TAG database, but we have the opportunity to do better

In-file metadata handling

- While this is a core software responsibility, there are many reasons ADC might care
- Example 1: more efficient and consistent job configuration (see previous slides)
- Example 2: file merging—a major component/nuisance/fact of life in production processing
  - Merging event data is "easy"—generally like extending an array of N elements to have N+M elements
  - Merging metadata, on the other hand, may require semantic knowledge
  - This is why we have so-called hybrid merge
  - Tools for this (in ROOT and in ATLAS) are improving
- Example 3: Event server bookkeeping
- Example 4: Getting bookkeeping and luminosity block completeness and accounting right when a single stream writes a sequence of files
  - Or when a writer gets events from several sources
  - May arise in several scenarios, including increasingly-many-core processing and AthenaMP

In-file metadata infrastructure

- ATLAS employs a sophisticated in-file metadata infrastructure that goes far beyond machinery to allow storage of non-event data in event data files
- Example: incident-handling features, as input file boundaries are artificial from the point of view of physics metadata, encountered asynchronously to state transitions of the event-processing framework
  - These artifacts of storage organization do not infiltrate ATLAS event and physics metadata handling
  - Infrastructure supports the general case of an event collection incarnated as a list of references to events that may reside amid other events in multiple files
  - In such a case, the appropriate metadata to propagate to the output may not correspond to the complete metadata record of each input file, as only selected events are processed
- Challenge: as part of the AOD/DPD merger effort, we need to make our metadata objects and supporting infrastructure work in both Athena and non-Athena environments
  - Robustly with and without the tools (e.g., the incident handling) that Athena supports
  - A non-negligible (daunting?) amount of work
- AND we need to make it work in less-than-trivial multiprocessing environments
  - BUT this should have salutary side effects for efforts like the event server work

Metadata considerations for ADC discussion, participation, action?

- Participate in efforts to reduce the needless proliferation of data types
  - Some of this is (boring(?) but necessary) nomenclature
- Contribute to nomenclature discussions more generally
- Participate in revisiting how configuration tags are defined, what is implied by an configuration tag (and the completeness thereof), and in ensuring consistency with Tier 0 practice
- Take advantage of the opportunities presented with Prodsys2 , DDM (Rucio), and AMI evolution to improve integration of and communication among components
- Take advantage of dataset-level metadata in task definition and task-to-job mappings
  - And use it to improve automatic dataset-level integrity checks
- Support efforts to ensure the robust delivery of returned metadata
  - In a consistent manner to multiple clients?
- Integrate the substantial core software efforts related to robust bookkeeping and in-file metadata management