Data Butler Meeting 2017-01-26

(Notes taken by KTL; all errors and omissions are his responsibility.)
John Parejko: Use cases/questions from jointcal:

- Given a butler repo, what datasets does it contain?
- Specify a subset of the repo, how do I know what metadata needs to go with those things in order to be useful?
- After generating output, what got generated? Clarification: Might need to combine pieces of data differently: need to compare visits that overlap on the sky, all measurements from a given CCD - Further clarification: find all datasets with something in common
- How do you get at various pieces of metadata?
- What should be available from a ButlerDataRef without saying get()?

Butler dataIds and DataRefs are supposed to be opaque; you should get datasets with them, not look into them

What information is needed from a dataId? Most if not all can be defined as datasets

- [ ] UW to define what is needed (but may wait until after detailed discussion about short-term composite access)

Jim Bosch: Iterating only over datasets that actually exist means iterating only over things that succeeded, which might have problems if there needs to be some kind of different handling for things that failed as opposed to things that were never supposed to exist
DataIdGenerator concept:

- Have a separate type of object, distinct from a Task, that can generate a list of dataIds based on a source of truth that might not be in the Butler repo and might possibly use non-Butler mechanisms
- Useful for generating input and output dataId lists in SuperTask workflow
- Possibly also useful for generating dataIds for loading calibration data
- General-purpose concept that could also help UW work on using Apache Spark
- Useful to have queriable metadata database within the repo for transportability

Ingest into a common model for metadata/registry and even paths?

- Have ExposureInfo and VisitInfo standardized already
- Currently can't query unless we read in all the exposures; need a better way
- There is metadata about SDSS that is not brought into VisitInfo (camcol)
- Some algorithms won't work for SDSS

Create a common metadata database that has extensions for particular cameras (esp. SDSS)
Have some pieces already:

- Registry, Serge's exposure metadata
- Need a better query interface (Butler DB interface has had some design effort), possibly the DataIdGenerator

- [ ] UW to define table schemas for various genres/"contexts" for dataset types:

- "raw" is potentially strange, coadds vs. single exposures vs. coaddTempExps vs. diffIms (template may be an identifier); need to be able to define new dataset types with new metadata fields
- Don't care if filenames are forced into a common model

Dynamic dataset type definition:

- Shouldn't be within the algorithmic part of a Task
- Needs to be observable by the workflow system before the Task is actually executed
- DES has information flow from algorithm via external communication to operators who create a well-known schema (kind of like an obs_ package) that then applies to the workflow and is checked
- Metadata and schema could be defined before the campaign or at workflow generation time
- Dataset type names must be configurable for each SuperTask. They are processing-dependent and so shouldn't be in `obs_`; they should be in something that is production-specific
- But dataset type definitions should likely still be close to the algorithm (might change with Wil's data interfaces push)

Having separate metadata databases is adequate to deal with transportability; the information that John wants to transport would be in there and doesn't need to be in separate datasets