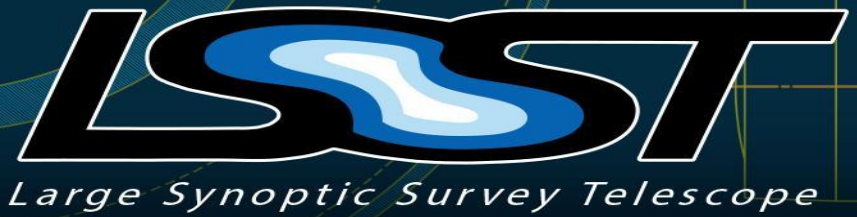
Batch Production Framework Futures

DMLT F2F 11/6/2018



Why DESDM?

- Using a working framework gets us from as-is to to-be.
  - Batch Processing Service may need to run Gen2 tasks for a while (8 months? 1 year?)
    - Adopting Pegasus requires ButlerG3, plus development of code that converts generic pipeline descriptions/configurations to the description that the 3rd party workflow understands.
  - Time to implement better design and prototyping of the more advanced features of Pegasus
  - Time for Gen3 middleware to mature
  - Ability to test Gen3 features as they are delivered, esp. those that will be needed in initial Pegasus development
  - The only work on DESDM framework will be in LSST-specific plugins or unexpected changes needed to get by
    - Some of these LSST plugins will be reused in the future framework

Requirement	DESDM	Pegasus
Integrating pipeline changes	works today, but might need tweaks	Very dependent upon Gen3 work. could need changes also for new pipeline
Pipeline configuration language	works; has ability to have global variables and override variables at different scopes	Needs to be designed and probably implemented.
Convert Gen3 pipeline definition + input data sets into set of work to do and apply execution configs	works so far for LSST drp pipeline	**Gen 3 QuantumGraph generator work** + design and implementation
Run Gen2 task assuming repo exists	Runs cmdlineTask as executable	Previous prototype had an activator which called the Task's run method.
Run Gen3 task assuming repo exists	Currently plugin calls Gen3 slac command line tool.	Needs to be designed and implemented.
Monitoring	Besides normal HTCondor mechanisms, the framework itself has some monitoring tools and the operators have written campaign and pipeline monitoring tools.	Normal HTCondor and Pegasus mechanisms. Additional monitoring would need to be designed and implemented
Get data out of DBB from inside pipeline job	Uses plugins. Current: http from inside job.	Needs to be designed and implemented
Set up Gen2 repo inside job	Has ability to run wrapper plugins. One was written for Gen2 tests that has pre-existing HSC sqlite3 files that treats like data file.	Previous prototype only worked on the first step of a pipeline when raw images were involved. There were no Gen2 tools to help create repo from scratch for other steps.
Set up Gen3 repo inside job	Has ability to run wrapper plugins. Making repo file(s) on the fly is either **Gen3 work** (subset functionality or ingest) or hacking something temporarily. **This is work that is similar to what would be needed in future workflow system.**	Needs to be designed and implemented. Making repo file(s) on the fly is either Gen3 work (subset functionality or ingest) or hacking something temporarily.
Save data to DBB from inside pipeline job	Uses plugins. Current: http from inside job.	Needs to be designed and implemented
Save Metadata	works today. need to change to use astro_metadata_translator	**requires Butler G3 work** to save metadata.
Save Provenance	works today.	doesn't exist. Need to design and implement storing of BPS provenance
Track single-site location	works	
Chaining Gen3 pipelineTasks in memory	Has ability to run wrapper plugins. Needs **Gen3 work** (Butler in memory that can be passed between pipelineTasks). Change wrapper plugin to call Gen3 pipelineTask directly instead of subprocess and see if multiple execs in wrappers works well enough to chain the pipelineTasks. **This work would be very similar to future workflow needs.**	Needs to be designed and implemented.
Running multiple tasks in single compute job for efficiency ("depth first")	Can run multiple tasks in single job in parallel where desired.	Needs to be designed and implemented. Not a known feature of Pegasus beyond clustering which appears to group tasks into a single black box (for success or failure).

Technical Meeting for ButlerG3?

- Started google doc with most important questions from LDF for production batch processing
  - Based on operational experience and knowledge of HTCondor, Pegasus, DESDM, Gen2 and Gen3 middleware, and LSST pipeline designs
- Iterations now include Nick and Michelle G from LDF and Jim Bosch
- There is some low hanging fruit. Some questions are documentation and how would you do “blah.” Some more complex. Examples:
  - Need examples of pipelineTasks that take config files, save Gen3 provenance, create and use composite datasets, need updated headers.
  - Concerns with the QuantumGraph generator design as it relates to flexibility for operations, e.g., blacklist images for a particular step in a pipeline but not other steps; write complex queries as multiple SQL queries if more performant.
- We will closely monitor progress throughout November/December, with an evaluation of total progress in January.

Roadmap

2019

- What pipelines will be available? January? New pipelines?
- DESDM “production” runs of Gen2 and Gen3 (using some of the Gen3 interfaces)
- January: begin testing of advanced Pegasus features (bundling, scale, restarts)
- Finish design details based upon testing results
- Pegasus run Gen3 pipelines with feature set v1 (Pegasus monitoring only)
  - (unclear what Gen3 features the pipelines themselves will need)

2020

- ButlerG3 -- Pegasus v1 good enough for “production” runs (commissioning, ops rehearsals)
- Implement Pegasus v2 features (retry, restart, more monitoring and runtime provenance)
- Integrate any new/modified Gen3 features required by pipelines

2021

- Scale and operational improvements Pegasus v3
  - Basic campaign manager
- Integrate any new/modified Gen3 features required by pipelines
- Feature freezes 2-4 months before any ops rehearsals and processing campaigns for data previews

2022

- Hardening of operational use
- Commissioning data processing
- Feature freeze 2-4 months before last ops rehearsals and processing campaigns for data previews