Using Nextflow to Manage Processing Workflows. R. S. Heyd[1], J. Perry[1], A. M. Fennema[1], and M. Read[2], [1]University of Arizona, Lunar and Planetary Laboratory 102A Sonett Space Science Building, 1541 E. University Blvd., Tucson, AZ 85721-0063 rod@pirl.lpl.arizona.edu, [2]University of Bern

Introduction: Developing systematic image processing pipelines is an ongoing challenge for orbital camera missions. Each mission has unique requirements, staffing, and hardware resources that place a number of restrictions on the systems that need to be developed to process the large volume of data acquired from these cameras. The HiRISE[1] team at the University of Arizona was tasked with developing the geometry pipeline to map-project and mosaic the image data acquired from the CaSSIS[2] camera on board the Trace Gas Orbiter. After examining several workflow management systems, a tool developed for bioinformatics turned out to have the best feature set and low overhead needed by the CaSSIS operations team. This tool is called Nextflow[3], an open source workflow management system (WMS) developed by the Comparative Bioinformatics Group at the Barcelona Center for Genomic Regulation.

Workflow Concept: Nextflow is a relatively light weight java application that a single user can easily manage. Additional support may be needed to set up processing resources, if using a job scheduling or cloud processing system. However, minimal support is needed for running the application on a single processing host. The application executes a processing workflow written in the nextflow domain specific language (DSL), this language is an extension to the java-based Groovy scripting language. The workflow definition file is a combination of nextflow directives with short unix-shell "script-lets" that define what each process does in the workflow. Nextflow's processing paradigm is loosely based on the concept of unix pipes, where input and output channels are defined that pipe the data from one process to the next until the workflow has completed. One of the great benefits of this approach is it allows the pipeline developer to focus on specific processes in the workflow without having to worry about how to stage data products in and out for processing; the nextflow's channel system takes care of staging the data to be processed for the developer. This allows for relatively rapid prototyping of the initial pipeline without the need of developing a lot of additional tools for staging data in and out of the workflow system.

Nextflow Features: Built-in to nextflow are robust reporting features, realtime status updates (posted to a url) and workflow failure or completion handling and notifications via a user-definable on-completion process. In addition, nextflow can support workflows running on a single host (with or without multiple cores), or via processes submitted to job scheduling systems such as PBS/Torque and/or SLURM as well as various cloud computing systems.

Running Nextflow: Launching a nextflow processing run consists of calling the nextflow command-line application with the workflow definition file, plus any options specific to the workflow definition file or to nextflow itself. Upon launch, the nextflow application analyzes the workflow definition to determine the processing path(s) for the input data. A working directory structure is created where each process has a separate subdirectory and nextflow's channel system stages the data for each process into the respective working directories as the data products become available from previous processes in the workflow. In addition, any products that are defined as output products can be specified and published to mass storage outside of the temporary work area.

Workflow completion. As each process of the workflow completes, log files are written to each process directory for inspection, if necessary. A completion handler can be called when the last process finishes to run any directory clean-up or logfile archiving that is needed. The completion handler can also be configured to send email notifications, or any other post-processing procedure that may be necessary.

Automated/Systematic Processing: As nextflow is designed to manage an individual workflow, another layer on top of nextflow is needed to manage multiple runs of the workflow in an automated or semi-automated fashion. In the case of CaSSIS, this has been accomplished via a wrapper script coupled with a back-end database where a queue of data to be processed is stored. The wrapper script can be used to queue individual images or groups of images to be processed, and launch the nextflow process to begin processing. Finally the completion handler is configured to call the wrapper script, to relaunch nextflow on the next image as soon as the previous image has completed processing.

References:

[1] McEwen A. et al. (2010) *Icarus*, 205:1, 2-37. [2] Thomas N. et al. (2017) *Space Science Reviews* 212:3-4, 1897-1944. [3] Di Tommaso P. et al. (2017) *Nature Biotechnology, 35:4,* 316–319.