

Conductor: Database Driven Automated Processing Pipelines

Bradford Castalia and Christian Schaller, Planetary Image Research Laboratory, Lunar and Planetary Lab, University of Arizona, Tucson, Arizona.

Digital image data returned from space probe instruments are typically processed by sequences of the same procedures to generate standard data products. This pattern - data processing pipelines - is, of course, not unique to planetary science projects. Scripts encapsulating the standard production sequences are commonly used to automate the processing. For the High Resolution Imaging Science Experiment (HiRISE) of the Mars Reconnaissance Orbiter (MRO) mission, however, the demands of large data volumes and short product generation times require large processing throughput rates. The HiRISE team intends to achieve this throughput by clustering inexpensive computing systems, which complicates the employment of processing scripts. The use of a database as a central component of the HiRISE data processing facility offers opportunities for how the automated processing can be managed.

Conductor is a pure Java application that manages sequences of procedures applied to a queue of data source files. Conductor is designed to manage processing pipelines without requiring the pipeline implementor to write a script. Instead, the definition of the procedures in a sequence is provided in a database table. The names of the source files to be processed are entered, whenever desired, into a list in another database table paired with the procedures table. These two tables constitute the pipeline. Conductor takes care of confirming read access to each source file, executing the sequence of procedures in the proper order, checking the outcome of each procedure to determine if it completed successfully, and running an on-failure procedure in case a sequence procedure does not complete successfully. This mechanism keeps the user focus on defining the procedures of the sequence and the source files to be processed and leaves the drudge work (and its sometimes tricky logic) for Conductor to handle in a consistent and reliable manner.

For simple procedure sequences the specification of the entries in the procedures definition table is about as simple as writing a script file. All the procedure execution, completion checking as well as the generation of process status records and log files for each source processed is provided automatically by Conductor. For complex procedure sequences and demanding data production operations Conductor offers access to flexible capabilities that include dynamic specification of log files, procedures, success conditions and on-failure procedures; user-specified run-time variables from either a configuration file or database fields; multi-pipeline chaining and branching; manual batch operation, use as a daemon polling for available sources, or run with a real-time process monitor GUI; and safe operation of multiple Conductors on a single pipeline in multi-processing and/or multi-processor environments.

Conductor is not intended as a substitute for scripts, though it may be used this way. Conductor supplements scripts. Any of the procedures in a Conductor pipeline may be scripts as well as binary executables. Conductor is intended to keep the process pipeline manageable even when the procedures and processing environment become complex.

Conductor executables, support packages, documentation and source code is available from the <http://PIRL.LPL.Arizona.edu/software/Conductor.html> web site.