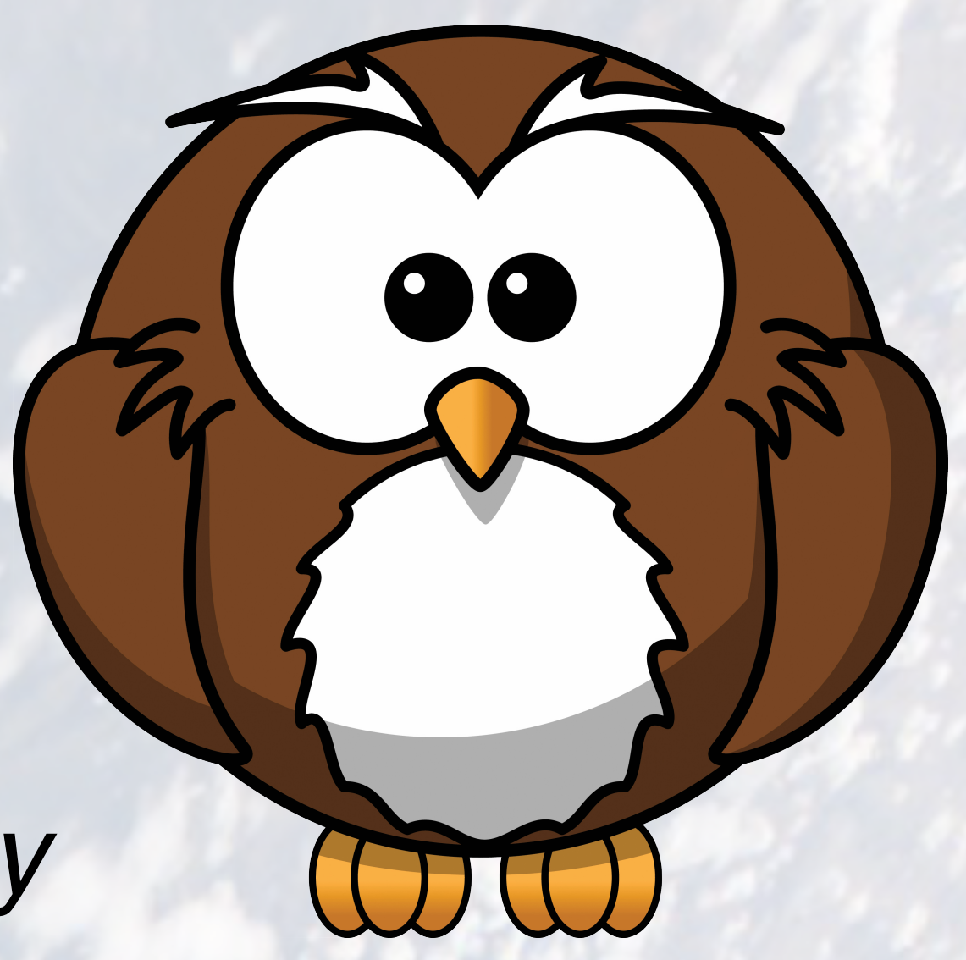
OWL



Francesco Pierfederici, Mike Swam, Gretchen Greene, Mark Kyprianou, Niall Gaffney
Space Telescope Science Institute, Baltimore, USA

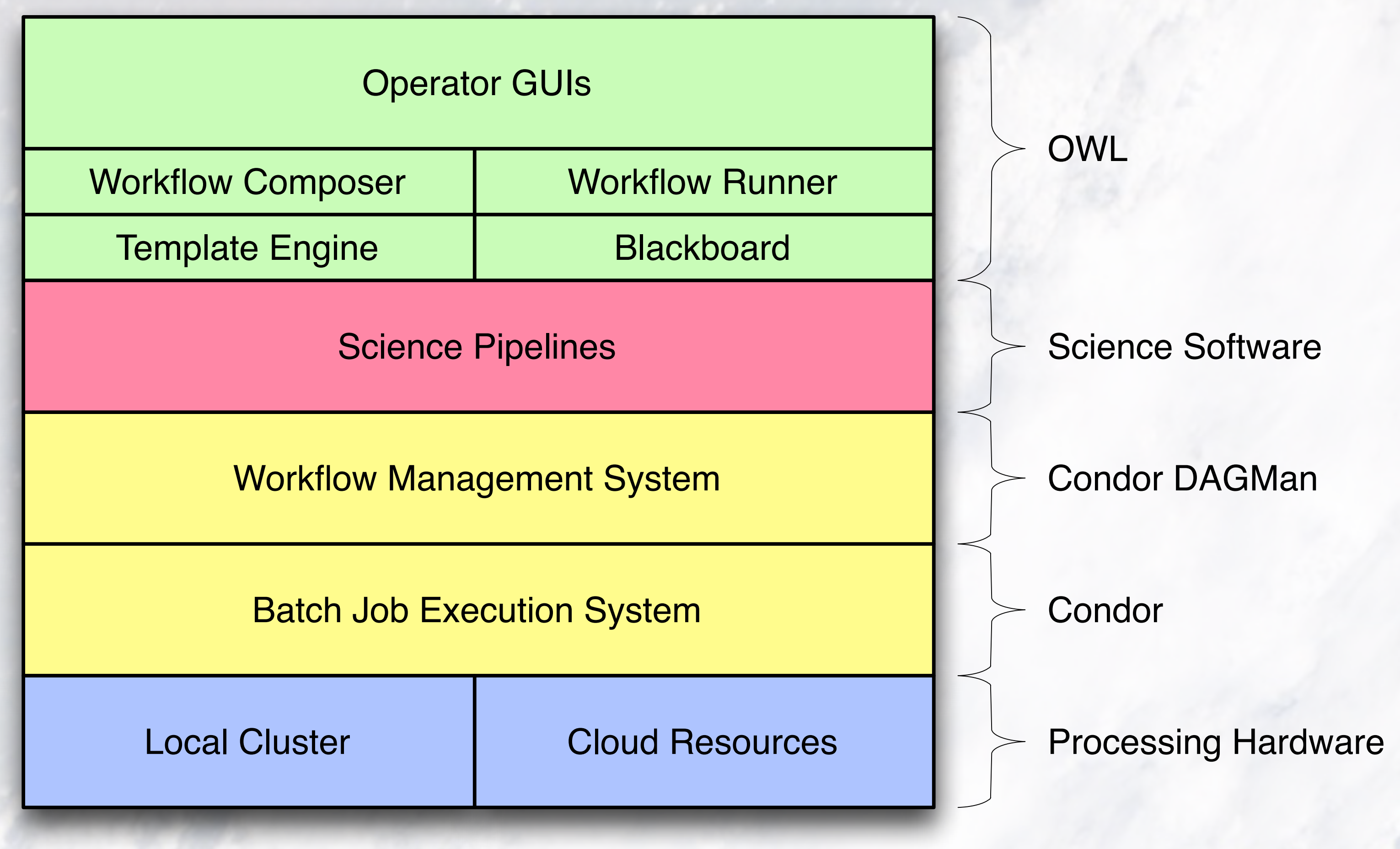
Workflow Management System

OWL

OWL is a Python layer on top of a grid middleware. OWL supports Condor, Apple XGrid and simple Makefiles out of the box. Adding support for another middleware involves writing a single Python plugin.

OWL provides operator GUIs, workflow templates and a blackboard architecture. The job scheduler provides the interface to the compute hardware.

Both HST and JWST projects have chosen Condor as the baseline job scheduler because of its wide adoption, history and stability. They have chosen OWL as the workflow management system.



Workflow Templates

Workflow templates allow customization of processing at run-time, based on the characteristics of the data (i.e. some pieces of data might not need to go through some processing steps) and/or the environment (e.g. the availability of compute resources such as GPUs, the version of the software being used etc). They essentially make the mostly static Condor workflows fully dynamic.

OWL includes a library of workflow templates which are organized by observatory, telescope, instrument and observing mode. Each template defines the steps of a workflow, their requirements and dependency rules. Since each workflow must ultimately be able to be mapped into a DAG (a requirement imposed by the Condor DAGMan meta-scheduler) loops are not allowed.

Workflow and step templates can include variables pointing to Python objects. These variables can refer to a data model for the data being processed (e.g. a MosaicImage object composed of FocalPlaneArray objects) and/or the environment (e.g. the path to the data, the version of the pipeline being executed, the name of the current user, the name of the execute machine etc).

Given a piece of data, OWL instantiates the appropriate workflow templates producing an abstract workflow. Abstract workflows are not tied to any particular grid middleware and can be turned into Condor DAGs, Apple XGrid workflows or simple Makefiles and then be executed.

Blackboard Architecture

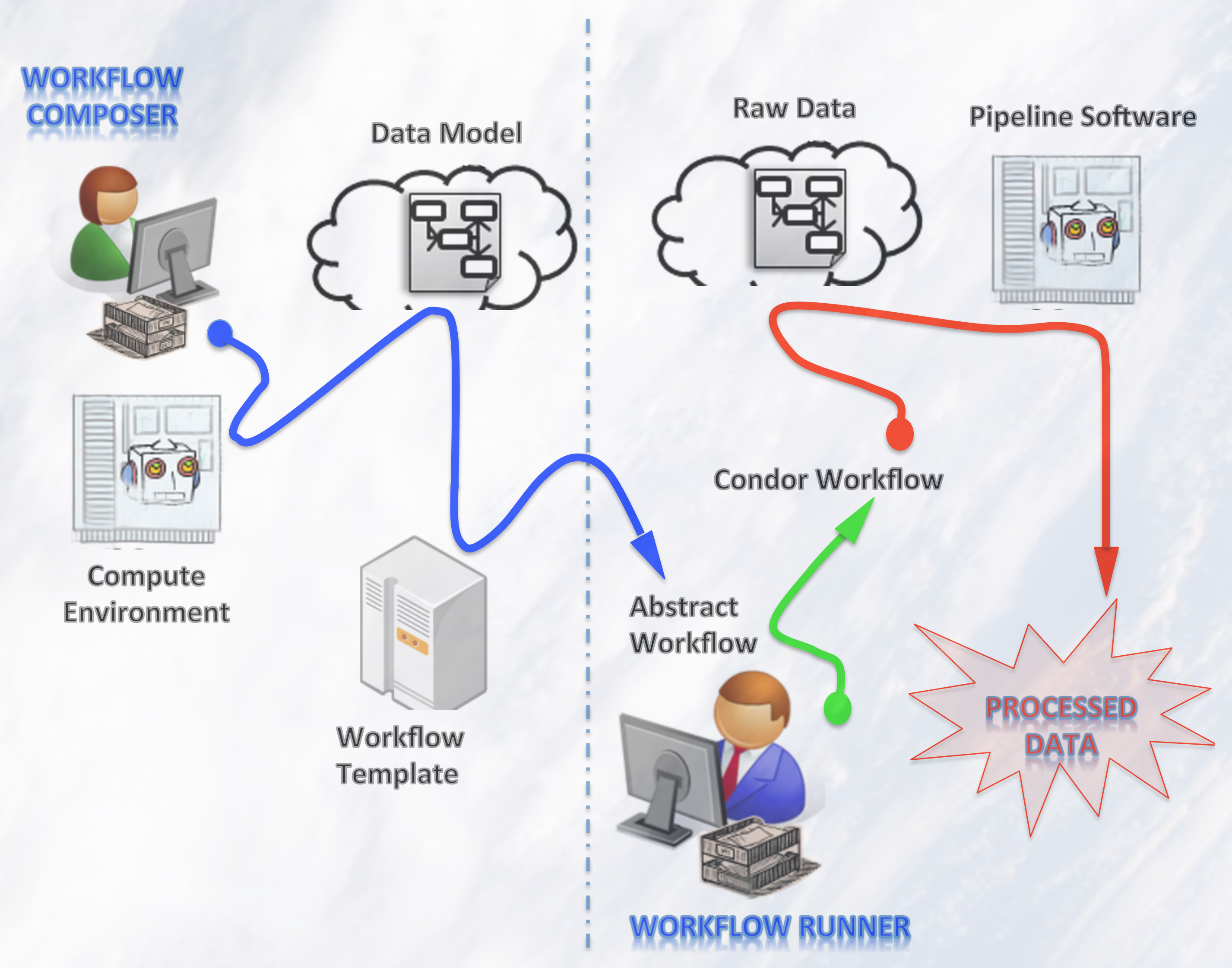
Blackboards are data structures that hold both the instantaneous and the historical state of the system. OWL provides a process-centric blackboard and a dataset-centric blackboard. These together allow access to critical information such as:

- Which pieces of data are being processed by the system.
- The processing history of any given dataset.
- Statistics on the system performance and latency.
- The composition of the processing cluster.
- The resources (e.g. RAM, disk space, processing power) available on any given processing machine.

This functionality is considered essential for JWST operations, and has been reinforced by years of experience with HST data processing on STScI OPUS blackboards.

Interfaces

The two interfaces that OWL offers developers and pipeline operators are a web-based GUI and a set of command line tools. The GUI allows users to inspect the blackboards, submit workflows, monitor them and stop/restart them.
The command line tools are what the GUIs rely upon and provide facilities to process data and suspend/restart running workflows. These utilities have access to both the data to process and optionally a rich data model which describe both data and hardware resources.



Status and Availability

OWL is currently still a prototype but it is actively used in the support of JWST data management system design. It has been chosen as the JWST workflow manager and is being developed as a replacement for HST OPUS. It is open source and interested parties should feel free to contact the authors to obtain a copy.

James Webb Space Telescope
JWST
NASA • ESA • CSA

SPACE TELESCOPE SCIENCE INSTITUTE
Operated for NASA by AURA