CRDS to CRDS Pipeline
for JWST
Interface Control Document

2015-08-16

Overview and Scope

This document describes the shared file system interface used to transfer CRDS files from the CRDS file submission component to the CRDS Pipeline (archive ingest system for reference files). This document does not describe in detail the contents of files or the ways in which CRDS or the CRDS pipeline perform their respective tasks. Instead, it focuses on the exchange of files between CRDS and the CRDS pipeline and their respective obligations and restrictions in complying with the file exchange interface.

Shared File Directory

Files are delivered from CRDS to the CRDS pipeline by placing them in a file delivery directory. First, each delivered file is copied into the shared delivery directory. After copying the delivered files, CRDS writes out a file catalog which lists each delivered file of the delivery, one file per line. After noticing the existence of a new catalog file, the CRDS pipeline ingests the delivered files into the archive, and on completion, removes all delivered files and the catalog from the delivery directory.

Order of Delivery

CRDS shall deliver reference and rules files first, then write out the catalog which lists the delivered files as a complete set. The creation of the catalog file in the delivery directory signifies that CRDS has completed it's half of the delivery.

Order of Ingest

The CRDS Pipeline shall ingest catalogs according to the order of sequence numbers, not according to the order of the lexical sort of catalog files. Within each catalog, the CRDS Pipeline shall ingest files in the order they are listed.

Catalog Name and Format

The file manifest (catalog) shall list each delivered file, without a path, one per line. The catalog shall be named matching a regular expression “jwst_\d+.cat” where the name consists of the string “jwst_” followed by a sequence number consisting of 6 or fewer decimal digits followed by the extension.cat. The sequence number is not zero-filled and so the lexical sort order of the catalog files is not guaranteed to be in numerical sequence; it is possible for a file with fewer digits to sort-by-name after a file with more digits.

Example Catalog Name

An example JWST catalog name is:

jwst_4.cat

Delivery Catalog Example Contents

Catalogs list delivered files one per line:

jwst_0009.pmap

jwst_nircam_0005.imap

jwst_nircam_linearity_0002.rmap

jwst_nircam_linearity_0020.fits

jwst_nircam_linearity_0021.fits

...

Acknowledgment of Archiving

The CRDS Pipeline acknowledges the receipt and successful archiving of delivered files by removing each file and the catalog (in all forms) from the shared file delivery directory. Removal of the catalog, including any alternate extensions, signifies that delivered files are fully in the archive and available and released for distribution and use.

Users and File Permissions

The CRDS server and delivery subsystem runs as user “crds” and has read-write access to both catalog files and delivered files. The CRDS pipeline accesses the delivery directory under the group “crdsoper” and has the capability of copying, moving, and removing files in the delivery directory only. The CRDS server maintains hard linked references to all files in the delivery area so it is paramount that the CRDS Pipeline make not changes to files (other than renaming) prior to making private copies.

Delivered File Properties

Delivered File Naming

Files delivered by CRDS are already named in an official final form which is tracked in the CRDS catalog and associated with file metadata. Further, rules files delivered by CRDS are a network which are interconnected on the basis of official file names; Rules files refer to other rules files on the basis of unique versioned names. The CRDS Pipeline shall not rename delivered files. The CRDS Pipeline may rename the delivery .cat file during the course of processing the delivery. The CRDS Pipeline shall be independent of the names of delivered files and capable for storing binary files as well as text.

Delivered File Contents Don't Change

The exact contents of delivered files are tracked and verified in CRDS using sha1sums. The CRDS Pipeline shall not change the contents of delivered files in any way. All files shall be

stored verbatim, effectively treated as binary.

Extra or Extraneous Files

Other than renaming the file delivery catalog, the CRDS Pipeline shall not create temporary files in the file delivery directory. The file extension of the delivery catalog may be changed to signify different states of CRDS Pipeline processing, but the extension shall always contain the sub-string ".cat". Since the CRDS pipeline shall be independent of file contents and file names, additional formats and extensions for delivered CRDS files

Filename Length

CRDS filenames up to a length of 128 characters shall be supported by the archive database.

File Permissions

As stated earlier, all delivered and catalog files shall be read only for group "crdsoper" (and the CRDS Pipeline).

File Size

The archive shall accept reference files up to a length of 16G for a single file. This number is a worst case estimate and may change depending on calibration software development.

Catalog File Extensions

The stage of CRDS Pipeline file delivery processing is reflected by renaming the delivery catalog file with different extensions.

Extension	Meaning
.cat	Initial list of delivered files from CRDS.
.cat_submit	CRDS pipeline has noticed file delivery.
.cat_proc	CRDS pipeline is ingesting files.
.cat_err	CRDS pipeline has encountered a fatal error which requires trouble shooting.
(all forms of catalog removed from delivery area)	All delivered files are in archive ready for distribution.

Pipeline CRDS Cache Synchronization

The current configuration of CRDS for both HST and JWST depends on a cache of CRDS files and configuration information which is local to the pipeline. During normal pipeline operations and best references computations, in this mode, CRDS runs fully decoupled from the CRDS server and relies on the cache for rules and configuration information. When files are delivered, there is process of several steps used to update the pipeline's local CRDS

cache.

First cron_sync

After successfully archiving files preparing them for distribution, and acknowledging file archiving success to CRDS, the CRDS Pipeline shall sync delivered files to the pipeline's CRDS cache by running the cron_sync script. This keeps lengthy file transfer times for reference files as a background process which is not encountered by pipeline operators interactively. The CRDS context does not change at this time, the delivered files become available in the pipeline but are not the default.

Set Context

After the CRDS Pipeline performs the first cron_sync, the CRDS system is ready for the crds_team and pipeline operator to update the default operational context on the CRDS server using the Set Context page. This changes the central definition of the operational context and makes it available for synchronization to remote pipeline's

Second cron_sync

After the pipeline operator has changed the CRDS Server master default context using Set Context, the pipeline shall re-execute the cron_sync script to update the locally cached default context.

Pipeline Context Switch Verification

This second cron_sync invocation will nominally be called with the switches –verify-context-change and –push-context. --verify-context-change asserts that it is an ERROR if the pipeline's local default context does not change following this cron_sync. The –push-context switch uploads the default context now defined in the pipeline's CRDS cache to the CRDS server for display in the Remote Caches display which compares server state to remote cache states.

No Direct Access to CRDS Cache

The CRDS cache in the pipeline is an integral part of the CRDS design. The CRDS pipeline shall not change the structure, permissions, or contents of the CRDS cache in any way. All routine (i.e. not failure recovery) changes to the CRDS cache in the pipeline shall be performed using the CRDS wrapper script cron_sync.