Astronomy and Computing 16 (2016) 41–53

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

# The Calibration Reference Data System

P. Greenfield[*], T. Miller

*Space Telescope Science Institute, 3700 San Martin Drive, Baltimore, MD 21218, USA*

ARTICLE INFO

*Article history:*
Received 7 January 2016
Accepted 11 April 2016
Available online 21 April 2016

*Keywords:*
Calibration
Pipelines

ABSTRACT

We describe a software architecture and implementation for using rules to determine which calibration files are appropriate for calibrating a given observation. This new system, the Calibration Reference Data System (CRDS), replaces what had previously used for the Hubble Space Telescope (HST) calibration pipelines, the Calibration Database System (CDBS). CRDS will be used for the James Webb Space Telescope (JWST) calibration pipelines, and is currently being used for HST calibration pipelines. CRDS can be easily generalized for use in similar applications that need a rules-based system for selecting the appropriate item for a given dataset; we give some examples of such generalizations that will likely be used for JWST. The core functionality of the Calibration Reference Data System is available under an Open Source license. CRDS is briefly contrasted with a sampling of other similar systems used at other observatories.

## 1. Introduction

When running automatic calibration pipelines one important capability needed is the ability to identify automatically the appropriate calibration files that should be used when calibrating the data. The kinds of calibration files are typically numerous, and there may be many that have to be applied to a specific dataset. Examples include dark current, bias, flat field, photometric, and geometric distortions, among many others. Typically calibration files depend on particular observing modes or parameters, or may be time variable. Calibration files also differ across instruments both in purpose and specific instantiations of a particular reference type.

## 2. Prior solution

The system that was used for the Hubble Space Telescope (HST) was called the Calibration Database System (CDBS, Cox and Tullos, 1994; Lubow, 1997; Cox et al., 1998; Lubow et al., 1997; Swam et al., 2004). It was used successfully for many years (with some design changes documented in the cited references). The experience with it revealed that there were some limitations that proved to constrain its capabilities, and as a result a newer system that removed these constraints was sought. We will briefly describe the design of CDBS at a high level and the constraints that it posed. This provides the necessary background for discussing the design goals of CRDS.

The basic design of CDBS was centered around a database that contained information on each reference file (the generic STScI term for files used to calibrate data whether they be flat fields, darks, or file containing appropriate parameters to use in calibration). Typically the database contained information about the relevant observational parameters (mostly instrument mode settings and date of applicability). At the highest level, the basic approach was to run a program for each new dataset being processed (or reprocessed as the case may be) that essentially performed a query on the database for each type of reference file needed. The result of the query was a specific reference filename that would then be set as the value of a particular keyword in the dataset's header for use by the calibration pipeline.

The calibration pipelines retrieved the names of the appropriate reference files from the dataset's header, and used those to open that reference file for use in calibration. The reference files were kept in a standard location in the pipeline processing system. The program that updated the headers with the selected references files was called "bestref".

In reality, there was much more to CDBS than what has been briefly described. The system handled the addition of new reference files, first by validating the files met certain requirements, and then the archiving of those submitted files. The process for submitting new files involved a good number of manual steps and checks. One goal of the replacement system was to streamline the submission process somewhat.

The most critical aspect of the validation requirements for reference files concerned the keyword values used to match

[*] Corresponding author.
*E-mail address:* perry@stsci.edu (P. Greenfield).

http://dx.doi.org/10.1016/j.ascom.2016.04.001
2213-1337/© 2016 Published by Elsevier B.V.

datasets to the appropriate references. Unlike dataset parameter keywords that describe a specific instrument configuration for one observation, reference file keywords circumscribe a range of parameter values for which the reference file applies. In CDBS, ranges of parameter values were often represented by special values that expand into the discrete values used to define specific instrument configurations. As part of submitting reference files, these special values were combinatorially exploded into multiple database rows that represented each possible combination of parameter values intended to match to the specified reference file. As an example, a dataset APERTURE keyword would describe the specific aperture an instrument was configured to use for one observation. In contrast, the same APERTURE keyword specified in a reference file would describe, indirectly, the combined set of different specific APERTUREs that reference file was intended to be used for. When multiple parameters were used to match datasets to references, the number of distinct rows in the CDBS database was equal to the number of all possible combinations of parameters.

Furthermore, there were occasions for which standard SQL queries were insufficient for achieving the desired selection results. In such cases, software was run to generate custom queries to get around such limitations.

CDBS was used for 24 years of HST's operation. During that period, upgrades and enhancements were made to how it worked, but for the most part, the design did not change in any major way.

## 3. Limitations in CDBS

Twenty four years provides a long time to learn about what could be improved. This section highlights what issues proved the most problematic in using CDBS.

(1) **Difficulty testing new reference files.** The use of a database essentially limited the system to one set of rules for selecting reference files. Once a reference file was delivered to the system, effectively the rules for selection were modified in the operation system immediately. There were occasions when reference files were deemed to have passed functional testing (i.e., they performed the correct calibration for the designated cases), but were submitted with incorrect information about which modes or dates that they should be used for. As a result, the operational pipeline would only discover that there was a problem when processing real data, thus resulting in having to deal with correcting mis-processed data (or in some cases, the failure to even process). Problems in the operational pipeline are significant disruptions requiring significant work to correct. In the last few years a test version of the database was created so that tests could be performed without affecting operations, yet it still did not satisfy all testing needs.

(2) **Difficulty in undoing mistakes.** A delivery of a file with mistaken parameters could be difficult to undo. No reference file could be removed from the database for various reasons. (It is possible to address this easily by changing the schema; nevertheless, the existing design made such a change difficult.) In a number of cases, particularly if the date that it was targeted for was incorrect, it could seriously corrupt the selection logic. To take the most common error case: Files often had a "useafter" date associated with them that indicated that it should be used for a date after the specified one *if no other file had a later date that was still before the date of the observation*. If one submitted a file intended to be used for data after 2000-01-01 but mistakenly provided 1999-01-01 then two new file submissions are required to correct the problem. One to resubmit the new file with the correct date; the second to resubmit the file intended to be used after 1999-01-01 so that its useafter date is one second after the mistaken submission.

There are other cases where one mistaken submission may require a series of resubmissions of already submitted files to correct for the effect of the wrong submission. One such example is if one mistakenly supplies a "wildcard" for a parameter that should only apply to one case. In this event, all the other reference files intended for each of the other possible parameter values must be resubmitted. This makes the potential effects of mistakes serious and thus requires extreme care in submission.

(3) **Difficulty in supporting remote usage by astronomers.** Using the bestref facility remotely requires providing a web service so that remote processing can get the most up to date recommendations on the best reference file to use. There is one serious drawback in this though. Frequently the versions of the calibration software in the operations pipeline have not yet been publicly released, and there are times when the latest reference files require the latest software. Such a web service has the potential of recommending reference files that either are inappropriate for previous versions of the calibration pipeline software, or simply will not work with previous versions. As a result, this service was not provided.

(4) **Difficulty in allowing customized variants of selection rules.** Sometimes observers or teams have specialized calibration files that they wish to substitute in place of the standard ones. There was no simple way of allowing a team to run a customized version of CDBS to support this. The entrenched software requirements were too numerous (when one develops a system to run in only one environment, dependencies on that environment easily become entrenched in the system). Even if no customization was desired, it makes running the calibration pipeline remotely at another institution very difficult since there is no simple way to get the most recent recommendations.

(5) **Difficulty in providing remote pipelines a consistent bestref environment.** There are times observers *do not* want changing references files if they wish comparisons to previously computed results. They want the same rules applied, even if they are not the very best version. As a "single state" database expressing only the current best reference assignment rules, reproducing historical results in CDBS was difficult or impossible.

(6) **Difficulty in understanding what the effective rules are.** The exact selection rules are embodied in a database with a history of transactions for which many supersede previous ones. One really only knows the net effect by running queries on the database for specific cases, even when something quite simple could summarize what is desired for the current situation. The net effect of this is that people do not really understand what the rules are, and that they are not what was intended. Finding mistakes in this situation can be quite difficult. And once found, they can be difficult to rectify.

(7) **Difficulty in adding new kinds of selection rules.** The database schema effectively constrains what kinds of rules can be used. More complex rules either lead to horrendously complex queries, or custom software to generate custom queries. The use of the custom software/queries ends up making the system even more opaque.

## 4. The crux of a different approach

It became apparent that many of the limitations ultimately came down to the dependence on a database as the repository for the selection rules. This has a number of drawbacks in this particular application: