The Chandra X-ray Observatory Calibration Database (CalDB): Building, Planning, and Improving

Dale E. Graessle*[a], Ian N. Evans[b], Kenny Glotfelty[b], X. Helen He[b], Janet D. Evans[b], Arnold H. Rots[c], Giuseppina Fabbiano[d], and Roger J. Brissenden[e]
Smithsonian Astrophysical Observatory, 60 Garden Street
[a]MS-70, [b]MS-81, [c]MS-67, [d]MS-06, [e]MS-02
Cambridge, MA 02138-1516

ABSTRACT

The calibration database implemented for the Chandra X-ray Observatory is the most detailed and extensive CalDB of its kind to date. Built according to the NASA High Energy Astrophysics Science Archive Research Center (HEASARC) CalDB prescription, the Chandra CalDB provides indexed, selectable calibration data for detector responses, mirror effective areas, grating efficiencies, instrument geometries, default source aim points, CCD characteristics, and quantum efficiencies, among many others. The combined index comprises approximately 500 entries. A standard FTOOLS parametric interface allows users and tools to access the index. Unique dataset selection requires certain input calibration parameters such as mission, instrument, detector, UTC date and time, and certain ranged parameter values. The goals of the HEASARC CalDB design are (1) to separate software upgrades from calibration upgrades, (2) to allow multi-mission use of analysis software (for missions with a compliant CalDB) and (3) to facilitate the use of multiple software packages for the same data. While we have been able to meet the multivariate needs of Chandra with the current CalDB implementation from HEASARC, certain requirements and desirable enhancements have been identified that raise the prospect of a developmental rewrite of the CalDB system. The explicit goal is to meet Chandra's specific needs better, but such upgrades may also provide significant advantages to CalDB planning for future missions. In particular we believe we will introduce important features aiding in the development of mission-independent analysis software. We report our current plans and progress.

Keywords: Chandra X-ray Observatory, calibration, CalDB, pipelines, analysis software, x-ray astronomy, x-ray missions

1. INTRODUCTION

The Chandra X-ray Observatory is the most thoroughly calibrated space-borne X-ray mission to date. Its calibration requirements were known to be extensive from the beginning of the project,[1] and many of these requirements have indeed been met. It stands to reason, then, that Chandra should require the most extensive and detailed calibration database (CalDB) of any X-ray mission. To wit, the Chandra X-ray Center (CXC) CalDB contains more than 300 files, and the index files contain collectively over 500 FITS file extensions, each uniquely selectable using the CalDB index software interface.

The Chandra CalDB has been built according to the HEASARC[2] CalDB standard model (version 1.1).[3] As such, it includes a standard directory structure, indexing according to instrument (INSTRUME header keyword), and is accessible and maintainable using the CALTOOLS subset of the FTOOLS library.[4] The index is populated according to

* Send correspondence to DEG; dgraessle@head.cfa.harvard.edu

the specification given for Calibration Index File (CIF) version 1.1. The CIF incorporates dates, times, paths, filenames, extension numbers, and the values of certain header keywords contained in the CalDB data files.

The Chandra CalDB serves the following purposes:

- To store and archive calibration data files.
- To maintain a naming convention and header structure for all calibration files.
- To provide an index of calibration data, based on FITS header keywords, for software access.
- To permit updates of calibration data independent of software updates, while maintaining configuration control.
- To provide a traceable history of calibration data in the database by maintaining versioning.
- To maintain the prescription for translating calibration products into formats suitable for processing and/or analysis.
- To interface with Chandra software for (1) Standard Data Processing (SDP)[5,6] (2) Chandra's special analysis package known as CIAO[7], and (3) other CalDB-compliant software.

How well the CalDB meets these requirements depends on the due diligence of three CXC teams, namely the Chandra Calibration teams for the various instruments and spectrometers, the Science Data Systems Planning team (SDS), and for the purposes of this paper, the CXC Data Systems (CXCDS) staff, including the CalDB manager (DEG). It is an ongoing effort, mainly due to the dynamics of the Chandra calibration. The CalDB manager must participate in meetings and interactions with all three of these teams in order to keep up with the changing specific calibration requirements and results as they are derived.

The Chandra CalDB is maintained online at the CXC for use by the DS pipeline software, as well as for the local CIAO users. For those users not located at the CXC, a downloadable version of the CalDB is kept online alongside the Chandra DATA archive[8]. Figure 1 shows the number of downloads over monthly intervals of CalDB from the archive server. The volume is significant; currently the main CalDB tar file is approximately 1.0 GB in size, so this corresponds to approximately 40-100 individual downloads per month.

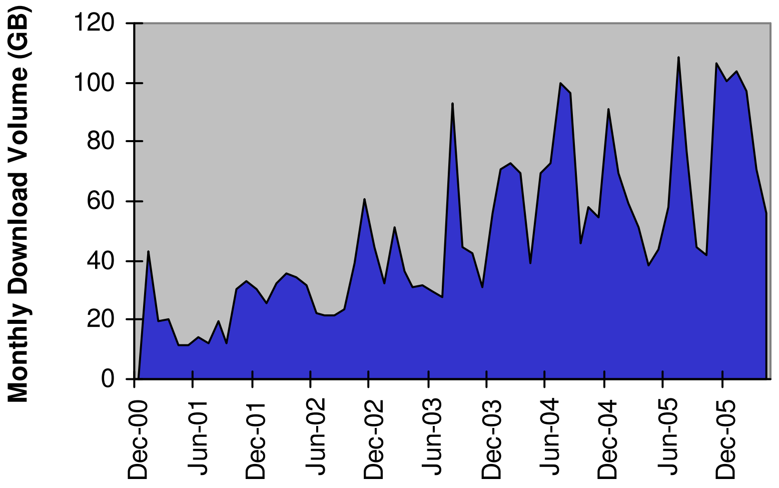


Fig. 1: CalDB download volume for each month of the mission beginning with the release of CalDB 2.0 in December 2000.

In Section 2, we present the current CalDB definition and implementation. Section 3 details some difficulties we have encountered in reconciling the CalDB structure with the Chandra calibration requirements. Section 4 relates how the CXC, with support from HEASARC, is planning to upgrade the current CalDB system to address the issues raised in section 3. The fifth section details some of the current progress toward this upgrade.

2. THE CURRENT CHANDRA CALDB SETUP

2.1. Basics of the HEASARC CalDB model

The HEASARC (High Energy Astronomy Science Archive Research Center) at the Goddard Space Flight Center, exists in part as a repository for archival data for any and all high-energy astrophysics space-borne missions. Naturally the ability to perform research analyses on data from the same or similar sources using similar procedures would be of paramount importance for the facility. The purpose served by the CalDB structures defined therein are three-fold:

- To separate CalDB data upgrades from software upgrades, so that software patches are not necessary for every CalDB upgrade.
- To allow multi-mission use of analysis software for missions with a compliant CalDB.
- To facilitate the use of multiple software packages for the same data.

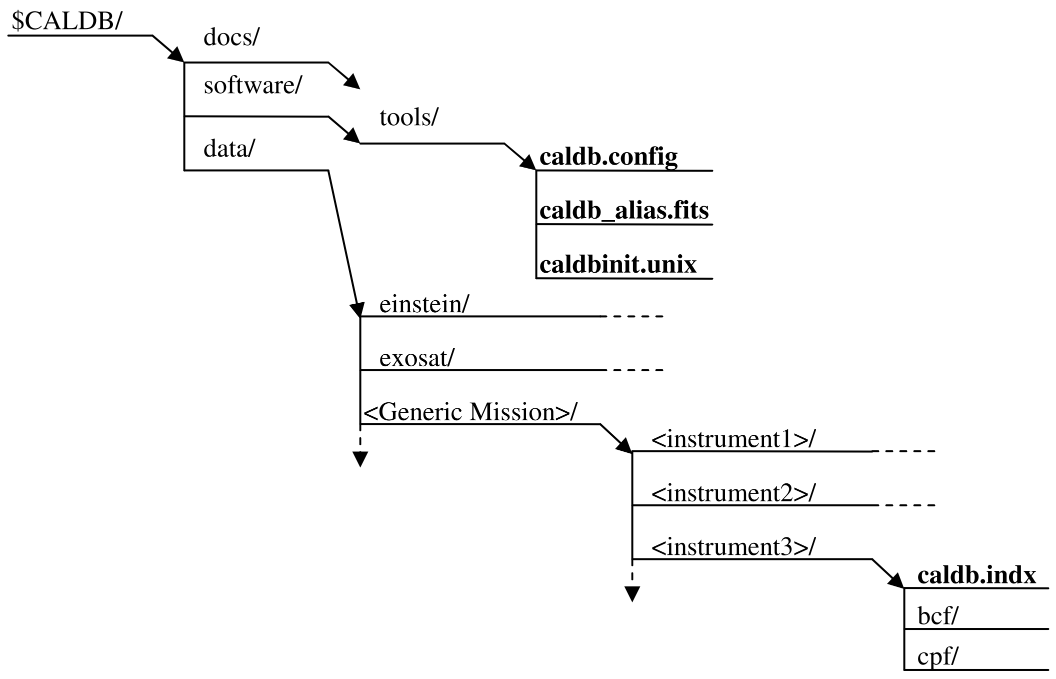


Fig. 2: Generic directory structure for the HEASARC CalDB. The file *caldb.indx* contains the index listings for its respective index branch.

Table 1: Chandra portion of the *caldb.config* control file, which gives the locations of index files for each mission branch of CalDB.

TELESCOP	INSTRUME	INDEX DEVICE	INDEX DIR	INDEX FILE	CAL_DEV	CAL_DIR
CHANDRA	ACIS	CALDB	data/chandra/acis	caldb.indx	CALDB	data/chandra/acis
CHANDRA	EPHIN	CALDB	data/chandra/ephin	caldb.indx	CALDB	data/chandra/ephin
CHANDRA	HRC	CALDB	data/chandra/hrc	caldb.indx	CALDB	data/chandra/hrc
CHANDRA	PCAD	CALDB	data/chandra/pcad	caldb.indx	CALDB	data/chandra/pcad
CHANDRA	SIM	CALDB	data/chandra/sim	caldb.indx	CALDB	data/chandra/sim
CHANDRA	TEL	CALDB	data/chandra/tel	caldb.indx	CALDB	data/chandra/tel

The HEASARC CalDB directory tree is illustrated in Fig. 2. Note that the "…/data/" branch subdivides into the individual missions. The missions then divide up into the respective instruments and spacecraft subsystems onboard, and these subdirectories include their own index files. This structure is actually somewhat flexible below the mission level, because the index file locations are set up in an ASCII text file called *caldb.config*, which dwells under the

"…/software/tools/" directory. Table 1 includes a portion of the *caldb.config* file, specifically with components for Chandra, as an example. It would be possible to set up *caldb.config* so that all instruments are included in the same calibration index file. However, that would cause considerable difficulty in establishing the uniqueness of index listings for specific.

In Table 2 are listed the mandatory CalDB keywords[9], that are included in a FITS file header for a typical calibration file in the CalDB. Table 3 lists the columns of the Calibration Index File (CIF)[10] for CalDB version 1.1, the currently active version of the CalDB at HEASARC. The keyword configuration for each FITS Header-Data Unit (HDU) that is to be listed in the CIF determines two things:

- The particular CalDB directory in which the file belongs.
- The particular index file in which the HDU is to be listed.

Table 2: Mandatory CalDB keywords version 1.1, as they must appear in the headers of CalDB data files. Those parameters marked with an asterisk* at the bottom of the table are full-fledged keywords for Chandra data files, but cannot be included as columns in CIF version 1.1. Hence they must be relegated to the Calibration Boundary (CBD) block.

Keyword Name	Allowed Values with CHANDRA	Index Column Name	Description
TELESCOP	CHANDRA	TELESCOP	Name of the mission.
INSTRUME	ACIS, EPHIN, HRC, PCAD, SIM, TEL	INSTRUME	The science instrument to which the data correspond.
DETNAM	ACIS-n (n=0-9), HRC-I, HRC-S, ACA-P, ACA-S, HRMA, PIXLIB, GRATING	DETNAM	The subset of the science instrument in question, (detector name).
FILTER	NONE	FILTER	The intervening absorbing filter used.
CALCLASS	PCF, BCF, CPF	CAL_CLAS	The calibration file class, PCF = primary calibration file, BCF=basic calibration file, CPF=calibration product file.
CALDTYP	DATA, FEF, TASK	CAL_DTYP	Cal file type, such as actual data, FITS-embedded function, or virtual data.
CCNM0001	See Table 4.	CAL_CNAM	CODENAME, the name of the type of calibration data. Used to identify desired dataset
CVSD0001	<yyyy-dd-mmThh:mm:ss>	CAL_VSD	Calibration validity start date.
CVST0001	<hh:mm:ss>	CAL_VST	Calibration validity start time.
CBDn0001, n=1-9	Various	CAL_CBD	Calibration Boundary Conditions.
CDES0001	Any ASCII text	CAL_DESC	Text description of the dataset.
GRATING*	HETG, LETG	Must be in CAL_CBD	CHANDRA: Name of the grating used.
GRATTYPE*	HEG, MEG, LEG	Must be in CAL_CBD	CHANDRA: Grating type, HEG, MEG, LEG.
CTI_CORR*	BOOLEAN	Must be in CAL_CBD	CHANDRA: CTI correction: TRUE or FALSE.
READMODE*	TIMED, CONTINUOUS	Must be in CAL_CBD	CHANDRA: ACIS event read setting.
DATAMODE*	FAINT, VFAINT, GRADED	Must be in CAL_CBD	CHANDRA: ACIS data collection mode.
OBS_MODE*	'SECONDARY', 'POINTING', 'RASTER', 'SCAN', 'SLEW'	Must be in CAL_CBD	CHANDRA: Observation mode setting.

Table 3: CalDB version 1.1 CIF columns. The CIF in CalDB version 1.1 has a fixed file definition.

Column Name	Data Type	Header Keyword	Description
TELESCOP	String[10]	TELESCOP	Same as TELESCOP in Table 2.
INSTRUME	String[10]	INSTRUME	Same as INSTRUME in Table 2.
DETNAM	String[20]	DETNAM	Same as DETNAM in Table 2.
FILTER	String[10]	FILTER	Same as FILTER in Table 2.
CAL_DEV	String[20]	-	Device specification for the CalDB file, “ONLINE” for live in the directory, “OFFLINE” for absent, <device name> for other location.
CAL_DIR	String[70]	-	Path to the directory containing the CalDB file.
CAL_FILE	String[40]	-	Name of the calibration file.
CAL_CLAS	String[3]	CCLS0001	Same as CCLS0001 above.
CAL_DTYP	String[4]	CDTP0001	Same as CDTP0001 above.
CAL_CNAM	String[20]	CCNM0001	Same as CCNM0001 above.
CAL_CBD[9]	String[70](9)	CBDn0001, n=(1-9)	The CBD block.
CAL_XNO	Int2	-	The FITS extension number containing the calibration data. May be zero if the primary HDU (e.g. an image) is used.
CAL_VSD	String[10]	CVSD0001	Same as CVSD0001 above.
CAL_VST	String[8]	CVST0001	Same as CVST0001 above.
REF_TIME	Real8	-	The MJD corresponding to CAL_VSD/CAL_VST.
CAL_QUAL	Int2	-	The “quality” of the calibration data. Only 0 (=GOOD) and 5 (=BAD) have been implemented in CalDB/CIF version 1.1.
CAL_DATE	String[10]	-	The UTC when this entry was added to the index file, yyyy-dd-mm.
CAL_DESC	String[70]	CDES0001	Text description of the dataset.

A certain subset of FTOOLS, namely CALTOOLS[11], includes several routines that will perform the various functions needed to build and maintain a CalDB for a generic mission. These are the following:

- *crcif:* Creates an empty CIF version 1.1 file on the default directory, named *caldb.indx*.
- *udcif:* Updates a CIF one file at a time, listing all HDU’s that contain a mandatory set of keywords.
- *caldbflag:* Allows the CalDB manager to change or update individual column values in a particular index listing, or group of listings.

In order to read the index file, either interactively or from an Application Programming Interface (API), one may use the CALTOOL routines

- *quzcif:* Searches the appropriate index file for matches to a set of required calibration data specifications. This routine is the heart of the CalDB 1.1 software interface. *quzcif* takes input parameters in the following order:
  - § *mission:* Actually, the value of TELESCOP applicable, required
  - § *instrument:* The value of the INSTRUME keyword applicable, required
  - § *detnam:* The value of DETNAM that is relevant, if any, otherwise “-“
  - § *filter:* The applicable filter designation, if any, otherwise “-“
  - § *codename:* The value of CCNM0001 i.e. the desired calibration data set type, required
  - § *date:* The starting date of the observation in format “yyyy-mm-dd”, required
  - § *time:* The hour-minute-second of the start of the observation, in “hh:mm:ss.sss”
  - § *expr:* A text expression enclosed in single quotes, which gives values of specific boundary conditions as may exist in the desired indexed HDU. If none known or applicable, then use “-“
- *lstgood:* Lists the good entries in a given index file for a given mission. Interactive use only.

Note that the user or the API must “know” the relevant parameter values in order to get an actual and unique index entry. There is no automatic means to determine such values for a given TELESCOP, INSTRUME, and CCNM0001. Therefore, particularly in the case of an API, the information to input to *quzcif* must be known in advance and hard-coded into the software.

The selection hierarchy works in the following order:

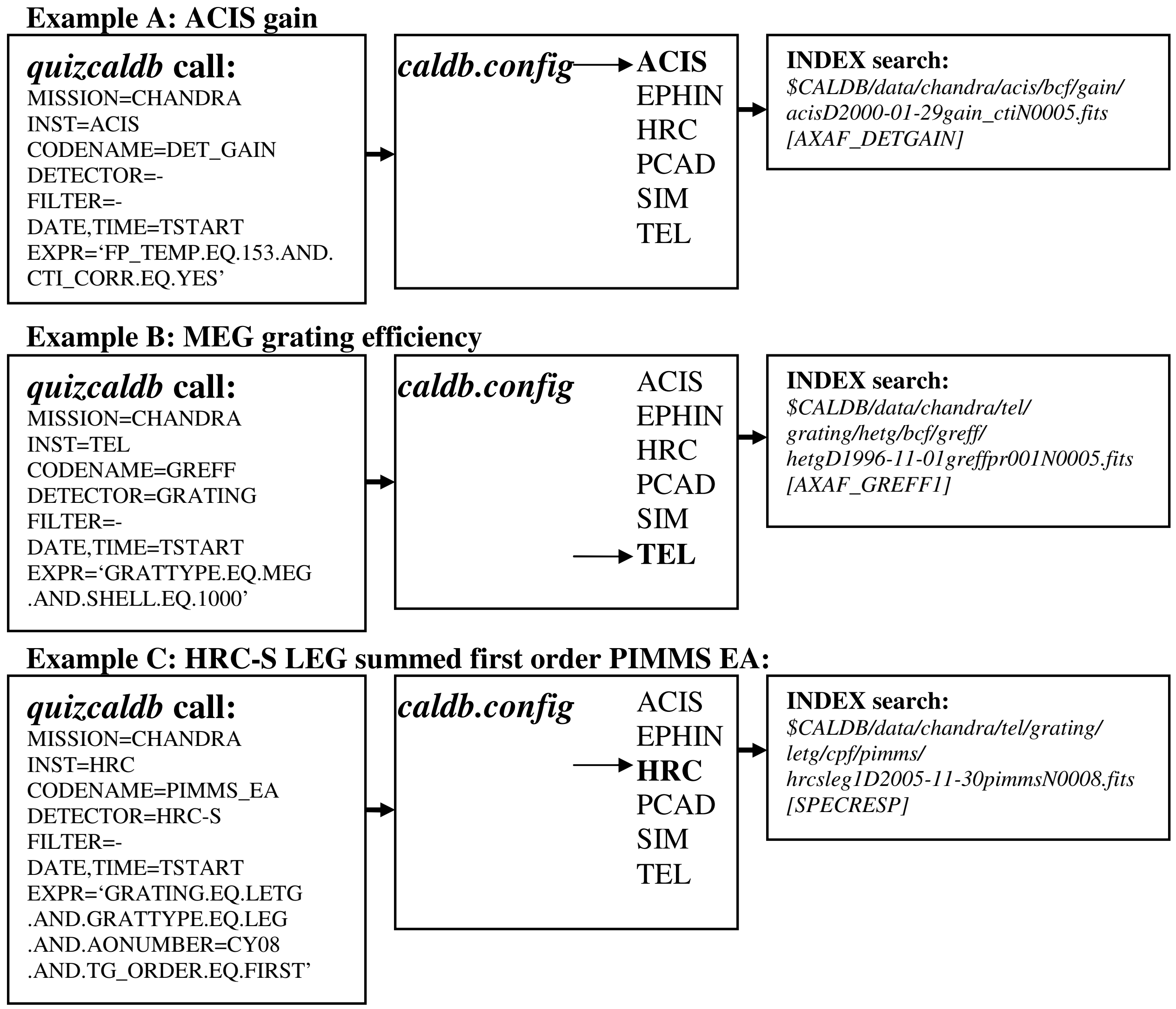
- For parameters *instrument*, *codename*: An exact match is required.
- For parameters *detnam* and *filter:* As these are optional FITS keywords, if a value of NONE is entered in the INDEX file, then any value entered in the calling routine will be ignored. Otherwise an exact match is required.

- For parameters *date* and *time*: These values are converted into the REF_TIME value in MJD, a double-precision numerical. For a given observation starting time, also converted into MJD, the INDEX listing with the latest REF_TIME before the starting time will be selected.
- For elements specified in *expr:* These are the boundary conditions, which are compared successively with the CAL_CBD column elements. If there is a parameter match, then values are compared with the indexed value or range of values. If the values do not match, the selection is excluded. If a parameter name doesn't match any of the parameters stored in the boundary column, then that parameter specification is *ignored* in the search.

2.2. The current Chandra CalDB implementation

Also included in Table 2 are the specific values for the keywords permitted by the Chandra branch of the HEASARC CalDB. This branch is the one built and maintained at the Chandra X-ray Center (CXC) by the author (DEG), with the support and/or approval of coauthors INE, JDE, GF (supervisor), and RJB (CXC program manager). In addition, Table 2 includes some additional keywords that Chandra CalDB files must address in some way to determine their applicability, including the GRATING and GRATTYPE keywords, which may be applicable to any INSTRUME and DETNAM configuration with *Chandra*. "GRATING" is a required keyword for *Chandra* FITS files in its own right; it is not a FILTER or INSTRUME value, and is not primarily associated with either ACIS or HRC. For the current Chandra CalDB setup, we have employed (or rather "usurped") the spacecraft subsystem "TEL" for certain spacecraft geometry libraries (PIXLIB), the HRMA calibration files, and the GRATING calibrations. However, "TEL" is not an instrument proper, and no science data are ever associated with it. By CXC FITS convention, INSTRUME = TEL admits of DETNAM settings of "PIXLIB", "HRMA", and "GRATING". Hence we have elected to store PIXLIB, HRMA, and GRATING CalDB files in the directory branch "$CALDB/data/tel/". Codenames such as GREFF, AXEFFA, VIGNET, AIMPTS, and SKY are indexed under the "tel" branch, and may only be selected by specifying ISNTRUME = TEL in *quzcif* or *quizcaldb*.

In Figure 3 below, we have illustrated three calls to the CalDB index using the CXCDS/CIAO tool *quizcaldb*. This tool is a C-wrapped version of the CALTOOL *quzcif,* described in Section 2.1 above. For *quizcaldb*, the parameter *inst* is the same as INSTRUME, and *detector* corresponds to DETNAM. *Mission* is used for TELESCOP.



Example A: ACIS gain

***quizcaldb* call:**
MISSION=CHANDRA
INST=ACIS
CODENAME=DET_GAIN
DETECTOR=-
FILTER=-
DATE,TIME=TSTART
EXPR='FP_TEMP.EQ.153.AND.
CTI_CORR.EQ.YES'

caldb.config → **ACIS**
EPHIN
HRC
PCAD
SIM
TEL

INDEX search:
$CALDB/data/chandra/acis/bcf/gain/
acisD2000-01-29gain_ctiN0005.fits
[AXAF_DETGAIN]

Example B: MEG grating efficiency

***quizcaldb* call:**
MISSION=CHANDRA
INST=TEL
CODENAME=GREFF
DETECTOR=GRATING
FILTER=-
DATE,TIME=TSTART
EXPR='GRATTYPE.EQ.MEG
.AND.SHELL.EQ.1000'

caldb.config
ACIS
EPHIN
HRC
PCAD
SIM
→ **TEL**

INDEX search:
$CALDB/data/chandra/tel/
grating/hetg/bcf/greff/
hetgD1996-11-01greffpr001N0005.fits
[AXAF_GREFF1]

Example C: HRC-S LEG summed first order PIMMS EA:

***quizcaldb* call:**
MISSION=CHANDRA
INST=HRC
CODENAME=PIMMS_EA
DETECTOR=HRC-S
FILTER=-
DATE,TIME=TSTART
EXPR='GRATING.EQ.LETG
.AND.GRATTYPE.EQ.LEG
.AND.AONUMBER=CY08
.AND.TG_ORDER.EQ.FIRST'

caldb.config
ACIS
EPHIN
→ **HRC**
PCAD
SIM
TEL

INDEX search:
$CALDB/data/chandra/tel/grating/
letg/cpf/pimms/
hrcsleg1D2005-11-30pimmsN0008.fits
[SPECRESP]

Fig. 3: Example of Chandra CalDB calls with quizcaldb, for ACIS *gain*, HETG (MEG) *grating efficiency*, and HRC-S/LETG summed first order *PIMMS effective area*. Specific knowledge of the CalDB index listings is necessary to obtain a unique filespec.

Table 4 lists all codenames relevant to the Chandra CalDB, and their associated instruments and text descriptions, with a list of parameters whose values must be specified to call a unique file and extension number from the CalDB. In several cases with grating files that are associated with the instruments ACIS or HRC, the file is stored in the …/tel/gratings/<GRATING>/ subdirectory, but is indexed in the ACIS or HRC instrument index branch. This is because the INSTRUME values for these files are set to ACIS or HRC, and the *quizcaldb* command will not find the files unless they are indexed under those branches, no matter where they are stored. We thought this was the best way to make the files intuitively accessible to the users, but still to have *quzcif* serve its function properly.

Table 4: List of codenames, or data set types, applicable to Chandra pipeline processing (SDP) and analysis software (CIAO).

Codename (CAL_CNAM)	**Applicable Instruments / Gratings**	***quizcaldb* parameter specifications required for unique selection (excluding mission, codename, date, and time)**
2D_PSF	ACIS, HRC	inst, detector, grid size
AIMPTS	TEL (PIXLIB)	inst
ALIGN/FID_POS	PCAD	inst
AMP_SF_COR	HRC	inst, detector
BADPIX	ACIS, HRC, PCAD	inst, detector, expr: FP_TEMP
BKGRND	ACIS	inst, detector, expr: FP_TEMP, CCD_ID
CCD_CHAR	PCAD	inst, detector
CCD_RESP	PCAD	inst, detector
CTI	ACIS, PCAD	inst, detector, expr: FP_TEMP, CCD_ID
DARK_CURR	PCAD	inst, detector
DEGAP	HRC	inst, detector
DET_GAIN	ACIS	inst, expr: FP_TEMP, CTI_CORR
DET_POS	SIM	inst
DET_POSCORR	SIM	inst
EFTEST	HRC	inst, detector
EVTSPLT	HRC	inst, expr: READMODE
FEF_PHA	ACIS	inst, expr: FP_TEMP, CTI_CORR
FDC	PCAD	inst
FPTEST	HRC	inst, detector
GAPLOOKUP	HRC	inst, detector
GEOM	EPHIN, TEL (PIXLIB)	inst
GMAP	HRC	inst, detector
GRADE	ACIS	inst, expr: DATAMODE
GTI_LIM	ACIS, HRC	inst, detector, expr: OBS_MODE
IRMF	ACIS	inst
IRU	PCAD	inst
MATRIX	ACIS, HRC	inst, detector
OBI_TOLS	TEL	inst, detector, expr: OBS_MODE
OSIP	ACIS (with grating)	inst, detector
P2_RESP	ACIS	inst, expr: FP_TEMP, CTI_CORR
QE	ACIS, HRC	inst, expr: CCD_ID, FP_TEMP
QEU	ACIS, HRC	inst, expr: CCD_ID, FP_TEMP
RWS	PCAD	inst
SATTEST	HRC	inst, detector
SFMA	PCAD	inst
SGEOM	TEL (PIXLIB)	inst
SKY	TEL (PIXLIB)	inst
TAPRINGTEST	HRC	inst, detector
TDET	TEL (PIXLIB)	inst
T_GAIN	ACIS	inst, expr: FP_TEMP, CTI_CORR
TGMASK2	HRC	inst, detector
TGPIMASK2	HRC	inst, detector
AXEFFA	TEL (HRMA)	inst, expr: SHELL
VIGNET	TEL (HRMA)	inst, expr: SHELL
WPSF	TEL (HRMA)	inst
GREFF	TEL (GRATING)	inst, expr: GRATING, GRATTYPE, SHELL
LSFPARM	ACIS, HRC	inst, detector
PIMMS_EA	ACIS, HRC	inst, detector, expr: AONUMBER, GRATING, GRATTYPE, TG_ORDER

3. SOME DIFFICULTIES WITH THE CURRENT IMPLEMENTATION

As indicated above, HEASARC CalDB version 1.1 has certain fundamental assumptions and paradigms that may not apply to a particular mission. The difficulties we have encountered in this implementation may be enumerated as follows.

- Index specification is rigid, depends on INSTRUME.
- Configurational keywords such as GRATING, GRATTYPE, OBS_MODE, must be specified as boundary conditions, even though they are better specified as CalDB keywords.
- FILTER keyword is unnecessary for Chandra, optional for FITS, but required in CIF. DETNAM is required in CIF, but not useful for ACIS.
- Calibration parameters such as boundary conditions must be hard-coded into the tools, which makes them mission-specific.
- Some features of the CalDB interface are undocumented; some documented features have not been fully implemented. We have identified these by experimentation.
- Updates to CALTOOLS software require a release of FTOOLS, which requires significant lead time.
- Other new missions' calibration database teams find CalDB index specification 1.1 inappropriate for their configurations. Some have avoided building a CalDB for HEASARC.

To be sure, the Chandra CalDB may now be built and fully maintained with the CALTOOLS software, and the addition of a CXCDS-built boundary block editor. The Chandra CalDB has been upgraded 36 times since the installation of CalDB 1.0 in March of 2000. Currently, the Chandra CalDB index includes 589 listings. The CalDB directory contains 376 files. Each index listing may be selected uniquely by a particular specification of parameters to *quizcaldb*, and the index has been built and maintained almost entirely using the CALTOOLS. However, we find that certain improvements to the interface would greatly enhance the applicability of CalDB to new configurations and missions, and hence the CXCDS is now moving toward implementation of a new CalDB system, with the support of the HEASARC.

4. A NEW CALDB INITIATIVE

As part of the recent HEASARC Senior Review Proposal and Contract (HEASARC Software Archives grant), the CXCDS has proposed the following initiatives for the *Chandra* CalDB, in three directives:

- Modify the CalDB interface to allow for improved mission-independent software usage.
- Implement the selection of related data files for downstream processing to be consistent with upstream processing. This would be helpful for higher-level pipeline processing as well as analysis of archival data without reprocessing by the users.
- Provide certain design scars in these modifications to allow further software additions. Specifically we want to add features for better ease-of-use in implementing a new HEASARC-style CalDB for a new mission.

We have translated the above initiatives into the following current activities specific to the Chandra CalDB interface:

- Generalize the CIF to be adjustable to the configuration and calibration requirements of specific missions.
- Provide a sufficiently flexible interface to read the generalized CIF.
- Allow for the automatic handling of calibration parameters such as boundary conditions and configuration keywords, so that the API does not require hard-coding of these parameters in each call to the CalDB.
- Provide backward compatibility with the existing missions represented in the HEASARC CalDB, through the addition of one or more default configuration control files for those missions.

There are actually two levels of software needed to separate mission-specific calibration parameters *and* data from the API of the analysis tools. First, there must be an intermediate library to determine which specific data types (here, *codenames*) will be required for a given mission for the task at hand, and to manipulate the calibration data once located into the correct units or interpolation grids for the API. For Chandra's data analysis system, *CIAO*, this is done mainly by the *ARDLIB*[12]. Second, below the ARDLIB level, the CalDB interface must handle calibration parameters and the index requirements for unique file and extension selection. The calibration parameters must generally be supplied from the headers of the observation data files being used in the desired analysis. (For cases in CIAO where the ARDLIB is not used, this interaction must be done at the tool level; hence in this case the tool must "know" what is needed and in what format it will be found. Such a tool is not mission-independent.) In any case, there must be some means of hand-shaking between the analysis process and the CalDB interface, so that calibration parameters may be passed back and forth to satisfy an automatic CalDB query. In the next section, we present an overview of how that may be done in the new Chandra CalDB interface design. For now we shall designate the upgrade as HEASARC CalDB/CIF version 2.0.

5. CURRENT PROGRESS

Our current activities were begun in August 2005 with the drafting of a requirements document for the new CIF definitions and the query interface software. Our intention is to document the upgrades thoroughly, including those aspects that apply to backward compatibility with existing CalDB branches. While the interface document continues to be modified and updated with additional specifications and clarifications, some progress has been made constructing the interface software, and the first phase of this is nearly ready for testing with a *live* CalDB and a new CIF.

5.1. The new CIF definition and *key.config*

We have dispensed with the INSTRUME subdirectory paradigm, so that now this keyword is no longer mandatory, but may be specified as a query keyword for a specific value of the codename. The new CIF may be more easily defined in terms of the types of keywords now to be included as columns in the file. There are three types of keywords to be specified: *mandatory*, *query*, and *optional*. Tables 5, 6, and 7 give an example of a subset of keywords which would appear as columns in a generalized CIF. Given the keywords specified in these tables, we require a systematic means to control the CIF structure a priori in the process of building and populating it. We have elected to include this information in a human-readable text file called *key.config*, similar to *caldb.config* discussed earlier.

The specification for *key.config* for the example in question is given in Table 8. The index-building routine reads this information and uses it to specify the index file or files for the given CalDB. The index-reading tool uses the *key.config* to set up the I/O for reading the index file, to identify necessary parameters for which values need to be supplied, and finally to select a unique index entry if essential parameters can be assigned appropriate values.

5.2. The CalDB query interface

The new CalDB query interface works in two query levels. The first-level query takes only the mission name and the codename for the desired data type, uses the *key.config* module to determine how to read the CIF, then queries the appropriate CIF for all occurrences of the given codename, and determines a minimum subset of the query and mandatory keywords that must be specified in order to obtain a unique index listing. This information is passed back to the API to provide the required parameter specifications, most frequently from the headers of observation data files (that must be available to do the desired analysis).

Subsequently, a second-level query may be submitted to the index routine, which in turn calls upon the *key.config* module for CIF information, and then produces the listing dictated by the available parameters. What is passed back to the API is simply the path, the filename, and the extension where the data are found. If an ARDLIB is employed, it will use further mission-specific information regarding the available calibration data to condition those data for use by the active analysis tool or process. Otherwise, the tool itself must know the mission, codename, and anticipated data format in the specific CalDB listing returned by the query interface. The two-level query is illustrated in Fig. 4, with Examples A and B shown for the corresponding cases shown in Fig. 3.

Alternatively, the first level query may be skipped, if the API or the user already knows which parameters to specify, and how to specify them. We refer to Example C in Fig. 4, the case for a PIMMS effective area file. In this case, only a second-level query is required, and a unique index listing, or if desired a series of index listings, may be obtained. A user might actually want to review multiple listings to verify the data structure of the CalDB for a given codename, for example. In any case, the interface may be used either in the two-level (or "hand-shaking") mode, or in the single-query mode similar to using *quizcaldb* with the old interface. Hence, older analysis systems for earlier missions could be refitted (with some software modification) to use the new interface for any existing CalDB, provided the appropriate default “key.config” file is built for that mission’s CalDB.

Table 5: Mandatory INDEX columns in the new CIF (version 2.0) specification. Any CIF must have these keywords, regardless of key.config specification.

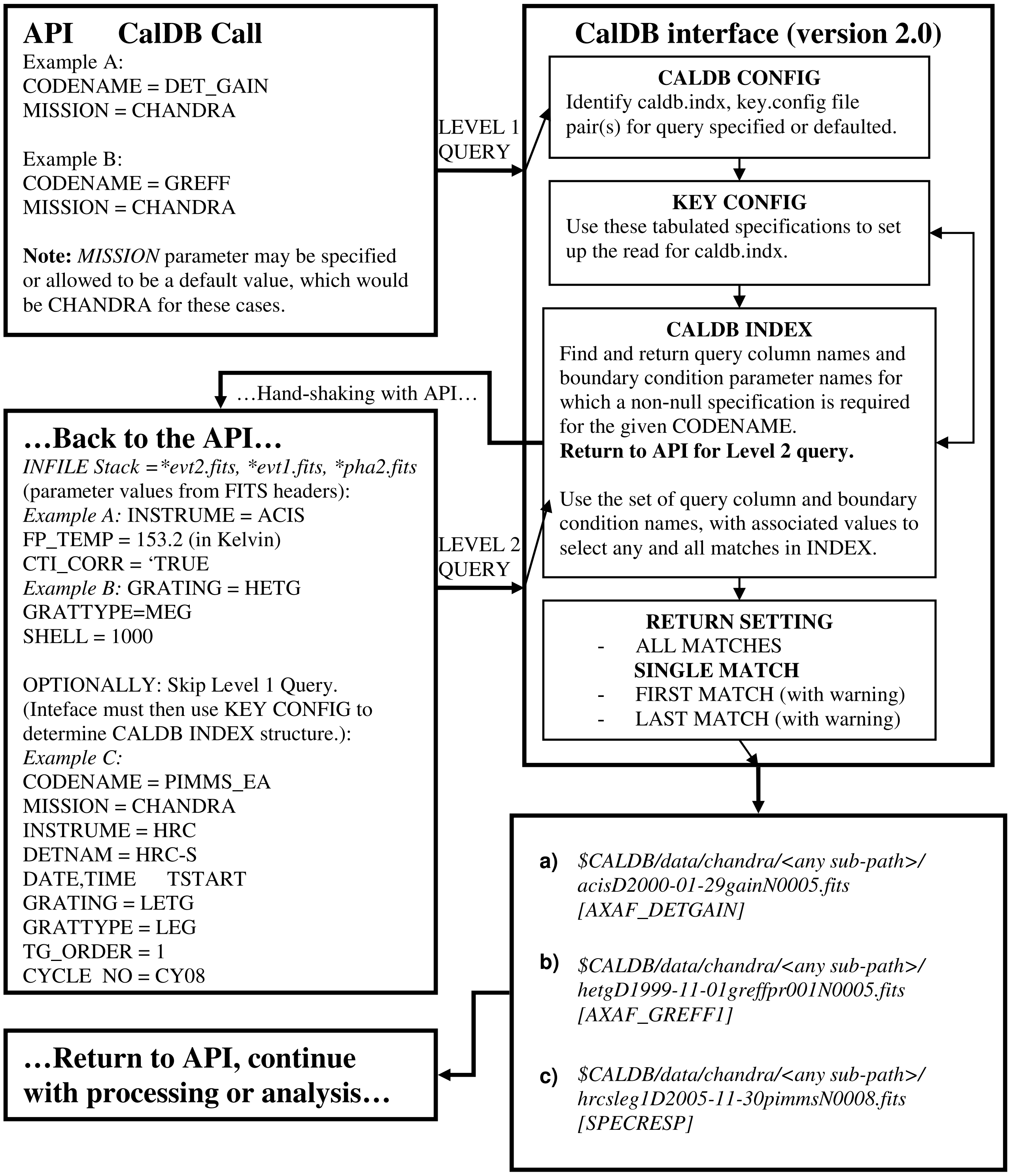
Index File Column Name	Data Type	Format	Equivalent Header Keyword	Description
CAL_DEV	string	char20		“ONLINE” for calibration files present in the CalDB tree. “OFFLINE” if not. “<device name>” for files located in another area.
CAL_DIR		char70		The path to the directory containing the calibration file specified by CAL_FILE.
CAL_FILE		char40		The name of the calibration file.
CAL_CNAM		char20	CCNM0001	The codename for the kind of calibration data the interface is asking for from the API or user.
CAL_CBD		char70[9]	CBDn0001	Calibration boundary conditions
CAL_XNO	int	int		The FITS extension number for the requested dataset
CAL_QUAL	int	int		The “quality” or usability of the calibration data.
CAL_DATE	string	char10		The UTC date when this entry was added to the index file, in yyyy-mm-dd format.

Table 6: Specified query columns for CIF version 2.0, from *key.config*, for the examples to be used in three CalDB calls in Fig. 4.

TELESCOP	string	char10	TELESCOP	Name of the mission, used to select CalDB tree.
INSTRUME	string	char10	INSTRUME	Science instrument system or full detector array.
DETNAM	string	char20	DETNAM	Sub-element(s) of INSTRUME pertinent to the calibration.
GRATING	string	char3	GRATING	Chandra Grating, one of HETG or LETG.
GRATTYPE	string	char4	GRATTYPE	Grating type, one of “LEG”, “HEG”, or “MEG”
CYCLE_NO	string	char70	AONUMBER	Name for a proposal cycle, e.g. ‘AO-1’, ‘CY08’.

Table 7: Optional columns, which *may* be specified in key.config. Some of these allow backward compatibility with existing mission CalDB trees.

CAL_CLAS	string	char3	CCLS0001	The calibration file class, one of “BCF” or “CPF”. Not interpreted by software.
CAL_DTYP	string	char4	CDTP0001	The type of the calibration file, one of “DATA”, “FEF”, or “TASK” (virtual data). Not interpreted by software.
CAL_DESC	string	char70	CDES0001	Short string description of the calibration data. Not interpreted by software.
CAL_VSD	string	char10	CVSD0001	Calibration validity start date, UTC, yyyy-mm-dd
CAL_VST	string	char8	CVST0001	Calibration validity start time, UTC, hh:mm:ss
REF_TIME	double	double		The MJD corresponding to CAL_VSD / CAL_VST. Used to compare with the observation date, to determine that calibration data are valid for that date.
CAL_VED	string	char10	CVED0001	Calibration validity END date, UTC, yyyy-mm-dd
CAL_VET	string	char8	CVET0001	Calibration validity END time, UTC, hh:mm:ss
END_TIME	double	double		The MJD corresponding to CAL_VED/CAL_VET.
FIDELITY	double	double	FDLT0001	Numeric “fidelity” of calibration data; may be used to select amongst different datasets with overlapping query requirements.



API CalDB Call

Example A:
CODENAME = DET_GAIN
MISSION = CHANDRA

Example B:
CODENAME = GREFF
MISSION = CHANDRA

Note: *MISSION* parameter may be specified or allowed to be a default value, which would be CHANDRA for these cases.

LEVEL 1 QUERY

CalDB interface (version 2.0)

CALDB CONFIG
Identify caldb.indx, key.config file pair(s) for query specified or defaulted.

KEY CONFIG
Use these tabulated specifications to set up the read for caldb.indx.

CALDB INDEX
Find and return query column names and boundary condition parameter names for which a non-null specification is required for the given CODENAME.
Return to API for Level 2 query.

Use the set of query column and boundary condition names, with associated values to select any and all matches in INDEX.

RETURN SETTING
- ALL MATCHES

SINGLE MATCH
- FIRST MATCH (with warning)
- LAST MATCH (with warning)

…Hand-shaking with API…

…Back to the API…
*INFILE Stack =*evt2.fits, *evt1.fits, *pha2.fits*
(parameter values from FITS headers):
Example A: INSTRUME = ACIS
FP_TEMP = 153.2 (in Kelvin)
CTI_CORR = 'TRUE
Example B: GRATING = HETG
GRATTYPE=MEG
SHELL = 1000

OPTIONALLY: Skip Level 1 Query.
(Inteface must then use KEY CONFIG to determine CALDB INDEX structure.):
Example C:
CODENAME = PIMMS_EA
MISSION = CHANDRA
INSTRUME = HRC
DETNAM = HRC-S
DATE,TIME TSTART
GRATING = LETG
GRATTYPE = LEG
TG_ORDER = 1
CYCLE NO = CY08

LEVEL 2 QUERY

a) *$CALDB/data/chandra/<any sub-path>/ acisD2000-01-29gainN0005.fits [AXAF_DETGAIN]*

b) *$CALDB/data/chandra/<any sub-path>/ hetgD1999-11-01greffpr001N0005.fits [AXAF_GREFF1]*

c) *$CALDB/data/chandra/<any sub-path>/ hrcsleg1D2005-11-30pimmsN0008.fits [SPECRESP]*

…Return to API, continue with processing or analysis…

Fig. 4: Illustration of the two-level CalDB/CIF version 2.0 query with hand-shaking, using three example codenames and configurations. These are the same examples as used in Figure 3.

Table 8: The key.config table that would be necessary for our examples given in Figure 4.

colName	dataType	Format	hdrKey	queryCol	nullVal
CAL_CBD	String	char70[9]	CBD*	no	“NONE”
CAL_CLAS	String	char3	CCLS*	no	“BCF”
CAL_DTYP	String	char4	CDTP*	no	“DATA”
CAL_DESC	String	char70	CDES*	no	“”
FIDELITY	Double	Double	“”	no	0.0
TELESCOP	String	char10	TELESCOP	yes	“”
INSTRUME	String	char10	INSTRUME	yes	“NONE”
DETNAM	String	char20	DETNAM	yes	“NONE”
GRATING	String	char20	GRATING	yes	“NONE”
GRATTYPE	String	char10	GRATTYPE	yes	“”
CYCLE_NO	String	char4	AONUMBER	yes	“”

6. ACKNOWLEDGMENTS

The authors acknowledge helpful discussions with HEASARC CalDB representative Dr. Michael C. Corcoran, NASA-GSFC, Greenbelt, MD. We also acknowledge the creator of the HEASARC CalDB system, Ian M. George. This work is supported by NASA under Chandra X-ray Center Contract Number NAS8-03060, and by the HEASARC Software Archives under grant number NNG05GH50G.

REFERENCES

1. D.A. Schwartz, L.P. David, R.H. Donnelly, R.J. Edgar, T.J. Gaetz, D.E. Graessle, D. Jerius, M. Juda, E.M. Kellogg, B.R. McNamara, P.P. Plucinsky, L.P. Van Speybroeck, B.J. Wargelin, S. Wolk, P. Zhao, D. Dewey, H.L. Marshall, N.S. Schulz, R.F. Elsner, J.J. Kolodzeijczak, S.L. O’Dell, D.A. Swartz, A.F. Tennant, M.C. Weisskopf, “Absolute effective area of the Chandra High-Resolution Mirror Assembly (HRMA)”, in *Proc. SPIE,* **4012,** pp. 28-40, 2000.
2. HEASARC = High Energy Astrophysics Archive Research Center, at the Goddard Space Flight Center, Greenbelt, MD. See http://heasarc.gsfc.nasa.gov/.
3. See NASA’s “HEASARC Calibration Database” at http://heasarc.gsfc.nasa.gov/docs/heasarc/caldb/caldb_intro.html
4. See "NASA's HEASARC Software" at web link http://heasarc.gsfc.nasa.gov/docs/software/ftools/ftools_menu.html.
5. Janet D. Evans, M. Cresitello-Dittmar, S. Doe, I. Evans, G. Fabbiano, G. Germain, K. Glotfelty, D. Hall, D. Plummer, P. Zografou, “The Chandra X-ray Center Data System: supporting the mission of the Chandra X-ray Observatory”, *Proc. SPIE*, **6270**, (these proceedings).
6. Ian N. Evans, M. Cresitello-Dittmar, S. Doe, J. Evans, G. Fabbiano, G. Germain, K. Glotfelty, D. Plummer, and P. Zografou, “The Chandra X-ray Observatory data processing system”, *Proc. SPIE*, **6270**, (these proceedings).
7. Antonella Fruscione, J.C. McDowell, G. Allen, N. Brickhouse, D.J. Burke, J. Davis, N. Durham, M. Elvis, E.C. Galle, D.P. Huenemoerder, J. Houck, B. Ishibashi, M. Karovska, M. Nowak, F.A. Primini, A. Siemiginowska, “CIAO: Chandra’s data analysis system”, *Proc. SPIE*, **6270**, (these proceedings).
8. Michael C. McCollough, Arnold H. Rots, and Sherry L. Winkelman, “Chandra Data Archive Operations: Lessons Learned”, *Proc. SPIE*, **6270**, (these proceedings).
9. “Mandatory Calibration File Keywords”, http://heasarc.gsfc.nasa.gov/docs/heasarc/caldb/caldb_keywords.html.
10. Ian M. George, Bill Pence, “Calibration Index Files”, http://heasarc.gsfc.nasa.gov/docs/heasarc/caldb/docs/memos/cal_gen_92_008/cal_gen_92_008.html.
11. Ian M. George, Ron S. Zellar, Rehana Yousaf, “Summary of CALTOOLS Tasks”, http://heasarc.gsfc.nasa.gov/docs/heasarc/caldb/docs/memos/cal_sw_93_004/cal_sw_93_004.html
12. John E. Davis, “A Framework for the Development of Multi-Mission Software”, in *Astronomical Data Analysis Software and Systems IX,* eds. N. Manset, C. Veillet, and D. Crabtree, 1999.