# Upgrading the Chandra/HEASARC CalDB Index and Software Interface

DE Graessle, IN Evans, K Glotfelty, XH He, AJ Rots, JD Evans, G Fabbiano

Chandra X-ray Center Data Systems, SAO, Cambridge, MA

**HEASARC CalDB Index and Interface Version 2.0 is defined and under construction by the CXC Data Systems Group.**

· Library, index builder, and search tool have been implemented.

· Testing is in progress.

## ABSTRACT

The HEASARC Calibration Database (CalDB) standard has been implemented for numerous X-ray missions, including EINSTEIN, EXOSAT, ROSAT, ASCA, Chandra, and SWIFT, among others. As mission configurations have become more complicated, it has become clear that a more flexible index definition and interface would be extremely helpful in adapting the CalDB standard to newer missions. Chandra has by far the most complete and complex calibration data structure of any mission to date. Hence, Chandra's need for an upgrade to the index definition and associated interface software is most pronounced. We have reported previously (SPIE Conference 6270, May, 2006) a plan for such an upgrade and herein report progress in implementing that plan. We have finalized a set of requirements for this effort. We have constructed the basic libraries, an index builder, and an index search tool. We have designed a very flexible index, which may include optional columns for calibration parameters, and have implemented an expandable boundary conditions block. We have also implemented a two-level index query system, with the goal of eliminating the hard-coding of calibration-specific parameters in the analysis tools. The ultimate goal is to facilitate the development of mission-independent analysis software.

## OUTLINE

· Introduction
· Requirements for HEASARC CalDB v2.0
· CXC implementation status
· Testing plan
· Summary

---

# I. Introduction

### Purpose of the HEASARC CalDB

· 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.

**NOTE**: Current HEASARC Index version is "1.1".

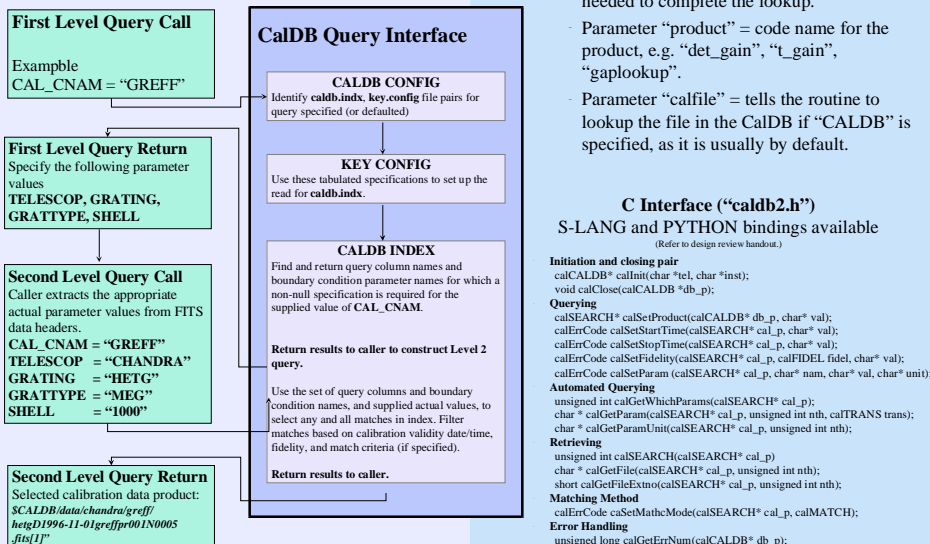### Issues to be addressed by Version 2.0 upgrade

· New mission configurations with intervening elements (e.g. gratings) and interchangeable instruments.
· Generalized indexing to accommodate a wider variety of calibration-related configuration and boundary parameters.
· Mission-independent tooling: Remove mission-dependent calibration parameters from tool coding.
· Backward compatibility with existing and archival missions in the HEASARC CalDB.

# II. Requirements

### CalDB Index v2.0: The Generalized Index

· Mandatory, optional, and query columns
  · Mandatory columns (Table 1) must be present in any CalDB index file
  · Optional columns
  · Query columns
· Optional and Query columns for an index file are defined in a "key configuration" file.
  · An appropriate key.config file may be used to construct and read a CalDB v1.1 index file.

**Fig. 1: Example of a multi-level or hand-shaking query to the CalDB as specified in the Requirements document for CalDB v2.0.**



**First Level Query Call**

Example
CAL_CNAM = "GREFF"

**First Level Query Return**
Specify the following parameter values
TELESCOP, GRATING, GRATTYPE, SHELL

**Second Level Query Call**
Caller extracts the appropriate actual parameter values from FITS data headers.
CAL_CNAM = "GREFF"
TELESCOP = "CHANDRA"
GRATING = "HETG"
GRATTYPE = "MEG"
SHELL = "1000"

**Second Level Query Return**
Selected calibration data product:
*$CALDB/data/chandra/greff/ hetgD1996-11-01greffpr001N0005 .fits[1]"*

**CalDB Query Interface**

**CALDB CONFIG**
Identify **caldb.indx**, **key.config** file pairs 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 supplied value of **CAL_CNAM**.

**Return results to caller to construct Level 2 query.**

Use the set of query columns and boundary condition names, and supplied actual values, to select any and all matches in index. Filter matches based on calibration validity date/time, fidelity, and match criteria (if specified).

**Return results to caller.**

---

# III. Implementation Status

### CalDB v2.0 Baselines: Primitives

· **Platforms** – Linux, Unix (no VMS)
· **System** – C++ compiler, STL, cstd
· **I/O Lib** – cfitsio for FITS file reader and writer
· **Modules** – input stacking, error handling, string manipulator, time analyzer, etc.
· **Defined types** – calSEARCH, calCALDB, calMATCH, calFIDEL, calTRANS

### Primary Input Files

· **caldb index** – Indices of caldb DATA in FITS format, mandatory + optional columns (including query columns)
· **key config** – plain text data table for user to specify attributes
· **caldb config** – plain text file to indicate locations of above files
· **caldb alias** – FITS file to record INSTRUME aliases

### INDEX BUILDER TOOL

· **CALINDEX**
  · Command syntax:
    calindex index keyconf calfile [clobber] [verbose]
  · Parameter "index" = path and name of the output index file, default=./caldb.indx
  · Parameter "keyconf" = key configuration file which defines optional columns.
  · Parameter "calfile" = filename, list of filenames, stack, directory, "CALDB", or none (blank index file).

### CALDB SEARCH TOOL

· **QUIZ**
  · Command syntax:
    quiz infile product calfile
  · Parameter "infile" = input file with metadata needed to complete the lookup.
  · Parameter "product" = code name for the product, e.g. "det_gain", "t_gain", "gaplookup".
  · Parameter "calfile" = tells the routine to lookup the file in the CalDB if "CALDB" is specified, as it is usually by default.

### C Interface ("caldb2.h")
S-LANG and PYTHON bindings available
(Refer to design review handout.)

**Initiation and closing pair**
calCALDB* calInit(char *tel, char *inst);
void calClose(calCALDB *db_p);

**Querying**
calSEARCH* calSetProduct(calCALDB* db_p, char* val);
calErrCode calSetStartTime(calSEARCH* cal_p, char* val);
calErrCode calSetStopTime(calSEARCH* cal_p, char* val);
calErrCode calSetFidelity(calSEARCH* cal_p, calFIDEL fidel, char* val);
calErrCode calSetParam (calSEARCH* cal_p, char* nam, char* val, char* unit);

**Automated Querying**
unsigned int calGetWhichParams(calSEARCH* cal_p);
char * calGetParam(calSEARCH* cal_p, unsigned int nth, calTRANS trans);
char * calGetParamUnit(calSEARCH* cal_p, unsigned int nth);

**Retrieving**
unsigned int calSEARCH(calSEARCH* cal_p)
char * calGetFile(calSEARCH* cal_p, unsigned int nth);
short calGetFileExtno(calSEARCH* cal_p, unsigned int nth);

**Matching Method**
calErrCode caSetMathcMode(calSEARCH* cal_p, calMATCH);

**Error Handling**
unsigned long calGetErrNum(calCALDB* db_p);
void calPrintErrors(calCALDB* db_p);
**Utility** void calFree(char *);

---

# IV. Testing Plan

### CalDB v2.0 3-Tier Testing Plan

· Elemental tests (against requirements)
  · Verification of elements
    · Index building
    · CalDB Searches
  · Error, warning checks
    · Index building
    · CalDB Searches
· Field test Chandra CalDB
  · Version 4.0.0.0
    · Index building
    · Second-level queries – all entries
    · First-level queries – all SDP and CIAO automatic searches
  · Version 3.x.x (backward compatibility)
    · Index building
    · Second-level queries – all entries
· Field test HEASARC missions (second-level queries)
  · ROSAT (archival)
  · ASCA (archival)
  · SWIFT (current)

### EXAMPLE 1:
**Building Chandra DEFAULT index with "calindex"**

· Includes files for which no INSTRUME may be specified:
  · GREFF (LETG, HETG), grating efficiencies
    · Per mirror shell
  · HRMA files
    · AXEFFA (axial effective areas, total, and per shell)
    · VIGNET (off-axis effective areas)
  · PIXLIB files (current implementation)
    · AIMPTS, GEOM, SGEOM, SKY, TDET
  · OBI_TOL, observation interval tolerances

### EXAMPLE 2:
**Search the above index for GREFF, AXEFFA, GEOM**

· quiz input_file.fits geom
· *<$CALDB>/data/chandra/default/pixlib/pixD1999-07-22geom N0005.fits[INSTRUMENTS]*
· quiz input_file.fits axeffa calfile=CALDB(SHELL=0100)
· *<$CALDB>/data/chandra/default/hrma/hrmaD1996-12-20axe ffaN0007.fits[AXAF_AXEFFA1]*
· quiz infile=input_file.fits product=greff calfile=CALDB(GRATTYPE=HEG; SHELL=0010)
· *<$CALDB>/data/chandra/default/grating/hetg/hetgD1996-11-01greffpr001N0005.fits[AXAF_GREFF3]*
· quiz input_file.fits greff calfile=CALDB(GRATTYPE=MEG; SHELL=1000)
· *<$CALDB>/data/chandra/default/grating/hetgD1996-11-01gr effpr001N0005.fits[AXAF_GREFF1]*
· quiz input_file.fits greff calfile=CALDB(GRATTYPE=LEG; SHELL=0100)
· *<$CALDB>/data/chandra/default/grating/letgD1996-11-01gre ffpr001N0005.fits[AXAF_GREFF2]*

# V. Summary

· Requirements have been completed and delivered for development of the Chandra/HEASARC CalDB Index v2.0, and associated interface software.
· CXCDS has implemented most of the requirements, building a significant library, and two tools: calindex (builder) and quiz (search and return CalDB datasets).
· CXCDS CalDB manager is now testing the above tools against the requirements document specifications.
· We anticipate the new CalDB interface will be ready for installation and testing in CIAO and the DS software during 2008, with the release of Chandra CalDB version 4.0.0.0.