An Overview and Tutorial in the Use and Creation of CDF Files

Bob McGuire, Tami Kovalick,

Bernie Harris and Bobby Candey

Space Physics Data Facility

Heliophysics Science Division (Code 670)

NASA Goddard Space Flight Center

Presented to the RBSP SOC team at APL, August 18-19, 2010

Our Goals Today

- Meet some of us from SPDF
- Help you understand and use effectively the capabilities and tools of CDF and SPDF
- Help you with any specific problems or questions
- Encourage a dialogue about your needs and your ideas

SPDF Names and Roles

- **Bob McGuire (Project scientist)** robert.e.mcguire@nasa.gov
- **John Cooper (Chief scientist)** john.f.cooper@nasa.gov
- **Bobby Candey (Lead system architect)** robert.m.candey@nasa.gov
- **Tami Kovalick (Lead system s/w developer)** tamara.j.kovalick@nasa.gov
- **Science**
  - Dieter Bilitza (ITM discipline scientist), dieter.bilitza@nasa.gov
  - Natasha Papitashvili (OMNIweb) natalia.e.papitashvili@nasa.gov
  - Len Garcia (SSC lead scientist) leonard.n.garcia@nasa.gov
- **Software**
  - David Han (CDF) david.b.han@nasa.gov
  - Mike Liu (CDF) michael.h.liu@nasa.gov
  - Bernie Harris (webservices), bernard.t.harris@nasa.gov
  - Reine Chimiak (advanced interfaces and services), reine.a.chimiak@nasa.gov
  - Rita Johnson (s/w development), rita.c.johnson@nasa.gov
  - Howard Leckner (s/w development and ingest ops) howard.a.leckner@nasa.gov
  - Nand Lal (guru) nand.lal@nasa.gov

Planned Topics

- **Reminder: What is SPDF**
  - And how are we relevant to RBSP
- **Basic definitions and concepts of CDF**
- **Tools to read and work with CDFs**
- **How to create CDFs**
  - Skeleton CDFs
  - Loading data and writing CDFs using IDLmakecdf procedures
- **Additional details**
- **RBSP-specific Q&A**

REFERENCE URL: http://spdf.gsfc.nasa.gov

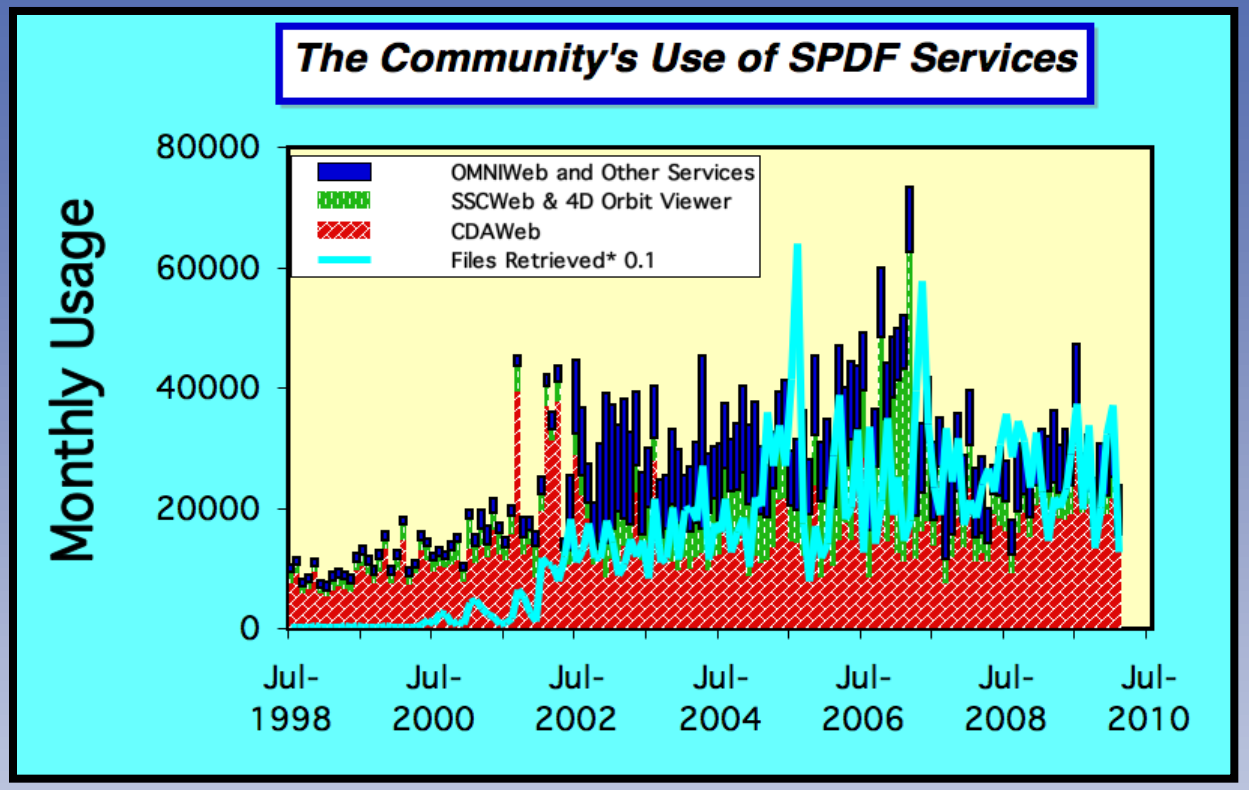
SPDF

What Is SPDF

- **Effective data services enabling NASA heliophysics science**
- **Emphasis on multi-instrument, multi-mission studies**
  (1) Specific mission/instrument data **in context** of other missions/data
  (2) Specific mission/instrument data **as an enriching context** for other data
  (3) Ancillary services and software (orbits, data standards, special products)
- **Unique enabling science data services; e.g.**
  - CDAWeb
    - Recent data includes reprocessed THEMIS data; new data from TWINS, STEREO and CNOFS; GPS TEC; Cluster archival Wide-Band
    - Linked CDAWeb data holdings to Autoplot
    - Current processes keyed to daily (weekday) ingests and updates
  - SSCWeb & 4D Orbit Viewer
  - OMNIweb and upgraded OMNIweb Plus
  - VSPO (heliophysics-wide dataset inventory)
  - CDF (now version 3.3 with 3.3.1 shortly)

SPDF and RBSP

Use of SPDF Services and Data



- **2009 explicit acknowledgements:**
  - 115 JGR Space Physics papers acknowledged SPDF services and/or data
  - This number represents 22% of JGR Space Physics papers published (=2008)
    - > N.B. same % for Space Weather
- **E.g. 200 distinct users made >100 CDAWeb requests Jun08 - May09**
  - 1250 users made >10

SPDF and RBSP

SPDF in the new Heliophysics Science Data Management Policy

- **One of two (active) Final Archives**
  - Ensure the long-term preservation and ongoing (online) access (with appropriate services) to non-solar NASA heliophysics science data
    - Serve and preserve data with metadata / software (with HQs)
    - Understand past / present / future mission data status (with HDMC)
    - May become a highly distributed long-term environment
  - **As Final Archive, allowed formats are NOT restricted to CDF**
- **Critical infrastructure components to Heliophysics Data Environment**
  - Heliophysics-wide dataset inventory (VSPO)
  - APIs (e.g. webservices) to SPDF system capabilities and data
- **Center of Excellence for unique enabling science data services**
  - I.e. CDAWeb, SSCWeb/4D Orbit Viewer, OMNIweb, CDF

SPDF and RBSP

- **SPDF (as a Heliophysics Final Archive) will work closely with RBSP teams and the relevant VxO (ViRBO) to capture an ongoing understanding of planned and actual RBSP data products**
- **SPDF will support RBSP in making and using CDF format**
  - Similar model to our close collaboration with THEMIS
  - Training and support as project and teams request
    - Plus custom support as needed
    - Necessary metadata as joint effort of RBSP, SPDF and ViRBO
- **SPDF plans to load appropriate RBSP planned and definitive ephemeris into SSCWeb's database**
- **SPDF will support the distribution of appropriate RBSP data through CDAWeb (or other SPDF services) and SPDF's webservices APIs**
  - SPDF can support a variety of methods for ingesting RBSP data
  - E.g. RBSP push to SPDF staging disks, SPDF pull from FTP or HTTP

CDF Basics

Basic Definitions: What is CDF?

- **Common Data Format (CDF)**
  - Self-describing data format for the storage and manipulation of scalar and multidimensional data in a platform- and discipline-independent fashion
  - Actual data format which CDF utilizes is intended to be completely transparent to the user and accessible through a consistent set of interface routines
    - Programmers are not burdened with performing low level I/O's to physically format and unformat data files
    - Supports multiple algorithms for compression by variable
    - Library core is pointer logic that maps to/from block data implementation
    - Built-in compression capability and transparent decompression
- **Software distribution includes C, Java, Perl and Fortran APIs**
  - High-level toolkit of utilities for creating, browsing and modifying CDF data to/from a regular text or XML file
- **Additional CDAWlib distribution includes rich set of IDL procedures**

Basic Definitions: Two CDF Concepts

- **Variables carry data**
- **Attributes carry metadata (i.e. information about data)**
  - Two levels of attributes
    - Global (file level) attributes
    - Variable level attributes
- **Some standard attributes are defined in CDF library,**
  - Additional standard attributes defined in the ISTP/IACG guidelines
- **Projects can/have defined additional standard attributes**
  - E.g. Cluster, THEMIS
  - Proposed that RBSP use PRBEM (COSPAR Panel on Radiation Belt Environment and Modeling) standard for some data
    - Constraints on variable naming & additional metadata requirements
    - Will be discussed later by Reiner Friedel
- **Variable attributes can point to other variables and thus carry information about relationships among variables**

Basic Definitions: ISTP/IACG Guidelines and CDAWeb/CDAWlib Software

- **ISTP/IACG Guidelines and subsequent extensions defined a set of implementation standards for CDFs**
  - All data is time ordered; times vary by record
  - Multiple required attributes with specific definitions
- **CDAWeb (IDL) s/w (CDAWlib) and additional tools are keyed to CDFs that follow the ISTP/IACG guidelines**
  - Core CDF and CDF toolkit work with all CDFs
- **CDAWeb service and s/w support the concept of a "Master" CDF to carry supplemental or over-riding metadata for a time-series of data-carrying CDFs**

CDF and SPASE

- **ViRBO presentation will say more about SPASE**
  - SPASE = Space Physics Search and Extract standard
    - http://www.spase-group.org
  - XML based heliophysics data model / data dictionary
    - I.e. Metadata
  - Description standards at overall dataset, parameter and granule level
    - Findability of data, long-term usability, base for new services in a distributed data environment
    - Focus to meaning of data rather than detailed layout of data
    - Used internally by several Heliophysics Virtual discipline Observatories (VxOs) and a defined standard for metadata exchange among Helio VxOs
- **ISTP/IACG guidelines broadly similar to SPASE standard**
  - Codified much earlier and differ in many specifics
  - Some existing code to partially map CDF metadata into SPASE
    - But some manual entry and inspection is unavoidable
  - ViRBO will work closely with SPDF to do most of that work

Using CDFs

I Have A Data File in CDF - Now What?

- **Data from CDAWeb can be readily displayed/subsetted in CDAWeb**
- **Use SPDF/CDF Data Translation Web Services**
- **Use CDF toolkit functions (CDFExport) to make ASCII or XML**
- **Use Autoplot (www.autoplot.org)**
- **In IDL: for CDFs written to the ISTP/IACG/SPDF Guidelines**
  - Value-added CDAWlib functions (that underlie CDAWeb)
  - Value-added display program (CDFX) built on CDAWlib
- **Use IDL or MatLab to read/write and manipulate data**
  - Using CDF supplied functions included in IDL or MatLab distributions
- **Write custom C, Fortran, Perl, C# or Java programs using CDF APIs**

Data is Coming from CDAWeb

- **Standard CDAWeb interfaces and services**
  - FTP, HTTP, Webservices (SOAP and REST), OPeNDAP
- **(New) link to AutoPlot**
  - Interactive data display tool (Jeremy Faden)
- **(New) IDL data selection GUI and command line loader**
  - Restore spdfcdas.sav or (soon) compile CDAWlib
  - IDL> spdfcdawebchooser

Data Translation Web Services

- **Use SPDF/CDF Data Translation Web Services to make ASCII or other output formats**
  - Generic: works with all CDFs
  - Go to http://spdf.gsfc.nasa.gov and link to Data Format Translators
  - Select the input file on your machine
  - Select e.g. ASCII output and submit
  - Retrieve resulting file
- **S/W is available as stand-alone programs**

I Have A Data File in CDF - Now What?

- ✓ **Data from CDAWeb can be readily displayed/subsetted in CDAWeb**
- ✓ **Use SPDF/CDF Data Translation Web Services**
- **Use CDF toolkit functions (CDFExport) to make ASCII or XML**
- **Use Autoplot (www.autoplot.org)**
- **In IDL: for CDFs written to the ISTP/IACG/SPDF Guidelines**
  - Value-added CDAWlib functions (that underlie CDAWeb)
  - Value-added display program (CDFX) built on CDAWlib
- **Use IDL or MatLab to read/write and manipulate data**
  - Using CDF supplied functions included in IDL or MatLab distributions
- **Write custom C, Fortran, Perl or Java programs using CDF APIs**

Simple CDAWlib Example

- **From IDL command line with CDAWlib installed**
  - With IDL_PATH pointing to CDAWlib s/w

```
@compile_cdaweb
vnames=“”
a=strarr(1)
a(0)=‘name_of_cdf_without_cdf_extension’
buf1=read_myCDF(vnames, a, /all, /NODATASTRUCT)
```

  - And use standard IDL commands to explore and use

```
help, /struct, buf1
etc ...
```

CDFX

- **Use IDL and value-added display program (CDFX) built on CDAWlib**
  - From IDL command line with IDL_PATH pointing to CDAWlib s/w

    @compile_cdfx
    cdfx

  - Select cdf file(s) to read
    - Select either all variables or specific variables that you'd like to work with
  - Plots, lists and CDF outputs like CDAWeb
  - Can use “Master” CDFs for additional metadata

Are there any questions so far?

Making CDFs:
Design and the Skeleton CDF

Basic Definitions: Two CDF Concepts

- **Variables carry data**
  - Have properties of Data Type, Dimensionality and dimension Sizes
  - Have properties of Variance (by Record, in a given Dimension)
- **Attributes carry metadata (i.e. information about data)**
  - Two levels of attributes
    - Global (file level) attributes
    - Variable level attributes
  - Have property of Data Type and may have multiple Elements
  - Some standard attributes are defined in CDF library,
    - Additional standard attributes defined in the ISTP/IACG guidelines
  - Variable level attributes can point to other variables and thus carry information about relationships among variables

Steps to Put Data into CDF

1. **Define and create the CDF structure to receive the data**
   - Create skeleton CDF
     - N.B. “Master” CDFs are skeleton CDFs used in a specific way by CDAWeb

2. **Use one of multiple options to add data**
   - In IDL, use IDLmakeCDF procedures
     - Use read_master_cdf to read skeleton and create a structure within IDL,
     - Load that structure using standard IDL calls as necessary,
     - Use write_data_to_cdf to create a CDF with data
   - Use makeCDF tool
   - Direct writes to CDF using CDF library
     - Directly in programs or calls inside IDL

N.B. If data is in another standard self-describing format, may want to use DTWS to convert data files into (some form of) CDF

- E.g. netCDF, HDF, FITS or CDFML description

Creating an ISTP/IACG Skeleton CDF: (1) Understand the Data to be Loaded

- **What are the key data quantities**
  - What is their definition/meaning?
  - What do you want to name them?
- **Understand**
  - Dimensionality,
  - Variance with time and dimension, and
  - Intrinsic dependencies of each of the data quantities
- **General rule is to capture relationships in the structure**
  - Otherwise capture relationships in variable attributes
  - Want relationships to be structured and machine-readable
    - Available for more general-purpose codes to exploit

Creating an ISTP/IACG Skeleton CDF: (1) Understand the Data in CDF Terms

- **A few examples**
  - A simple scalar (e.g. B magnitude) is dimension 0 and record-varying
    - By convention, time dependence is captured as record variance
  - Vector B might be 1 time-dependent/record-varying variable of dimension 1 and size 3
    - OR it could be (not recommended) 3 time-dependent scalars (dimension 0)
  - Flux at 10 energies should be 1 time-dependent variable of dimension 1 and size 10
    - Plus an attribute pointing to another variable with numerical values for these 10 energies (even if they don't vary in time)
    - AND/OR an attribute pointing to another variable with e.g. energy band (time-independent) identifications (for labeling)

More Data Examples

- **More complex cases:**
  - Flux at 10 energies and 15 pitch angles should be 1 time-dependent variable of dimension 2 and sizes (10,15)
  - An image (256x256) at 3 wavelengths might be 3 variables of dimension 2 and sizes (256, 256)
    - OR 1 variable of dimension 3 and sizes (256, 256, 3)
  - A slowly varying scalar might be non-record variant in a given file but allowed to be time-dependent across a dataset
  - Arrays of variable length should generally be sized to the maximum number of values
    - And use compression-by-variable for space efficiency
  - Possible to have variables which point to different time bases in same CDF

SKTEditor

- **SKTEditor is a Java, web-start application**
  - Download and start from web page but application runs locally
- **Using SKTEditor to create a new skeleton CDF**
  - Scalar variables
  - Variable attributes (descriptions, labels, units, display_type)
  - Global attributes and file naming
    - THEMIS file names
  - Checking and validation functions
    - Against ISTP/IACG standards and SPDF/CDAWeb extensions
  - Higher-dimensional variables
    - Pointer attributes
  - Display types and arguments
  - Virtual variables (functions, components)
  - Variations on time: epoch, epoch16,
    - Time as a Virtual Variable
  - Attribute editor

Creating an ISTP/IACG Skeleton CDF: (2) Create the CDF

- **Use SKTEditor to create a new, ISTP/IACG compliant CDF**
- **OR use SKTEditor to modify an existing CDF**
  - To create a new, ISTP/IACG compliant CDF

Creating an ISTP/IACG Skeleton CDF: Other Approaches

- **The ASCII way ...**
  - SkeletonTable (creates a structured ASCII file of CDF information)
  - Your Text Editor
  - SkeletonCDF (creates a CDF from a correctly structured ASCII file)
  - Testing e.g. with CDFX
- **The CDFEdit tool**

Creating an ISTP/IACG Skeleton CDF: Some Other Stuff

- **Data compression**
  - Specified only when data are being or have been loaded
    - Post-generation compression is most efficient for saving space
    - Strongly recommend only compression by variable
- **Leap seconds**
  - Support will be added to next release (3.4) of CDF

Questions?

Any specific RBSP CDF designs we should discuss now?

Making CDFs:
Loading Data into a CDF

Adding Data in IDL to a Skeleton CDF (1 of 4)

; Follow IDL specific instructions for set up of CDAWlib s/w from http://spdf.gsfc.nasa.gov

; Example illustrates how someone who might have data that's stored in IDL save sets,
; can use the routines to setup the "structure" of an output cdf w/ a master/skeleton cdf.
: Then add their data to the cdf variables by simply setting the pointer variables
; and then writing the data out ;to the desired cdf w/ the write_data_to_cdf routine.

;Compile the CDAWlib and IDLmakecdf routines (i.e. **@compile_IDLmakecdf**)

; Define the output cdf file name

out_cdf = 'hk_test.cdf'

; Read the master cdf, copy its contents to our out_cdf and define a structure,
; in this case "buf1" so that I may add the data to the variables.

buf1 = read_master_cdf ('hk_h0_vlf_00000000_v01.cdf',out_cdf)

;buf1 will have a tag for each variable, e.g.:
;EPOCH STRUCT -> <Anonymous> Array[1]
;B_SPD STRUCT -> <Anonymous> Array[1]
;E_SPD STRUCT -> <Anonymous> Array[1]
;etc.

Adding Data in IDL to a Skeleton CDF (2 of 4)

;To reveal the structure for a given variable:

help, /struct, buf1.epoch

;Should show:
; VARNAME STRING 'Epoch'
; DATA POINTER <PtrHeapVar1>

; Restore an IDL save set containing the data values for some variables
; this one puts the values into a structure called 'arec'

restore, 'hk_vlf_74300.sav'

;Type "help, /struct, arec" to see what was read/restored...
;** Structure <400502c8>, 11 tags, length=136, refs=1:
; EPOCH DOUBLE 6.2319283e+13
; SPDE FLOAT Array[16]
; SPDB FLOAT Array[8]
; BAVE FLOAT 58.0411
; RE FLOAT 12.0038
; MLAT FLOAT 55.0457
; MLT FLOAT 6.07035
; XGSM FLOAT -2.80766
; YGSM FLOAT -6.87608
; ZGSM FLOAT 9.43012

Adding Data in IDL to a Skeleton CDF (3 of 4)

```
; Set up the data arrays
   Epoch = double(num_rec)
   espd = fltarr(16,num_rec)
   bspd = fltarr(16,num_rec)
   bave = fltarr(num_rec)
   pos_mag = fltarr(3,num_rec)
   pos_gsm = fltarr(3,num_rec)
   seqno = indgen(num_rec)

; copy values out of idl save set
   Epoch = arec.EPOCH
   espd = arec.spde
   bspd = arec.spdb
   bave = arec.BAVE
   pos_mag(0,*) = arec.RE
   pos_mag(1,*) = arec.MLAT
   pos_mag(2,*) = arec.MLT
   pos_gsm(0,*) = arec.XGSM
   pos_gsm(1,*) = arec.YGSM
   pos_gsm(2,*) = arec.ZGSM
```

Adding Data in IDL to a Skeleton CDF (4 of 4)

```
; check for zeros in espd, bspd, bave and set to fill value
  fillvalue = -1e31
  i = where(espd ge 0)
  if  i[0] ne -1 then espd[i] = fillvalue
  i = where(bspd ge 0)
  if i[0] ne -1 then bspd[i] = fillvalue
  i = where(bave eq 0)
  if i[0] ne -1 then bave[i] = fillvalue

; Now set the buf1 structure data pointers for each variable
;  to the appropriate data arrays/values.

  *buf1.Epoch.data = epoch
  *buf1.E_SPD.data = espd
  *buf1.B_SPD.data = bspd
  *buf1.BAVE.data = bave
  *buf1.pos_mag.data = pos_mag
  *buf1.pos_GSM.data = pos_gsm
  *buf1.activity_index.data = seqno

; write data in the above pointers back out to our new cdf

stat2 = write_data_to_cdf(out_cdf, buf1)
end
```

Optionally Compress the Resulting CDF

- **CDF supports data compression**
  - Whole file or by variable (STRONGLY RECOMMEND “by variable”)
  - Choice of algorithms
- **For most efficient and effective compression**
  - First create and close the data CDF (as preceding)
  - Run cdfconvert (command line version)
    - cdfconvert input/hk_test compressed/hk_test
      -compression 'vars:gzip.9,var:%EPOCH%:none'

The MakeCDF Program

- **Highly relevant to teams with an internal format like csv**
- **Engine+Data+FileFormatDescription+Skeleton CDF = Output CDF**
  - Capable of supporting both ASCII and BINARY files, multiple time formats
  - Can automatically skip header records (or be manually controlled)
  - (Optional) arguments allow handling complex inputs; e.g. sub-records
- **File Format Description (FFD) or Translation File defines**
  - Mapping of data values to named CDF variables
  - Data format of input data
- **Example Input (specifically Geotail CPI plasma data made into csv)**

2007,304,0,1,10,102,0.4400E+03,0.91000E+02,0.18230E+03,-.43958E+03,-.17655E+02,-.76791E+01,0.1050000E+06,0.78900E+01,0.3055E+01

2007,304,0,2,48,842,0.4420E+03,0.90900E+02,0.18180E+03,-.44173E+03,-.13882E+02,-.69426E+01,0.1230000E+06,0.78000E+01,0.3048E+01

2007,304,0,4,24,590,0.4370E+03,0.90600E+02,0.18040E+03,-.43697E+03,-.30506E+01,-.45762E+01,0.1310000E+06,0.58800E+01,0.2246E+01

Example File Format Description (FFD)

```
RUNTIME_PARAMETERS
progress_output              ON        ;(ON/OFF),default=OFF
debug_output                 OFF       ;(ON/OFF),default=OFF
log_to_screen                ON        ;(ON/OFF),default=OFF
logfile_maxrecs              400       ;(positive integer),default=100
log_to_file                  ON        ;(ON/OFF),default=OFF
END_RUNTIME_PARAMETERS
;
INFILE_DESCRIPTION
format                       FREEFORM  ;(FREEFORM/FORMATTED),default=FREEFORM
data_type                    TEXT      ;(TEXT/BINARY),default=TEXT
text_type                    ASCII     ;(ASCII/EBCDIC),default=ASCII
binary_type                  VAX       ;(VAX/SUN/IBMPC) iff data_type=TEXT
delimiter                    ,         ;fields are comma separated
END_INFILE_DESCRIPTION
;
EPOCH_DESCRIPTION
algorithm                    0                    ;algorithm id
operands                     1 0 2 3 4 5 6 0 0 0  ;operands
END_EPOCH_DESCRIPTION
;
VARIABLE_DESCRIPTIONS
;vname       Size form H E br er nreps delta #/r M C aV aS aF Fc fill
Year          1    i   n n 1  0   1      0    n  0 n  n n  n  n   0
day           1    i   n n 0  0   1      0    n  0 n  n n  n  n   0
hour          1    i   n n 0  0   1      0    n  0 n  n n  n  n   0
min           1    i   n n 0  0   1      0    n  0 n  n n  n  n   0
sec           1    i   n n 0  0   1      0    n  0 n  n n  n  n   0
mseconds      1    i   n n 0  0   1      0    n  0 n  n n  n  n   0
SW_V          3    r   n n 0  0   1      0    n  0 n  n n  n  y   0
SW_Vc         3    r   n n 0  0   1      0    n  0 n  n n  n  y   -99999
SW_T          1    r   n n 0  0   1      0    n  0 n  n n  n  y   0
SW_P_Den      1    r   n n 0  0   1      0    n  0 n  n n  n  y   0
SW_Pressure   1    r   n y 0  1   1      0    n  0 n  n n  n  y   0
END_VARIABLE_DESCRIPTIONS
```

Loading Data

Questions on Making a CDF?

Other Things and Details

Subsetting and Supersetting CDFs

- **Use CDFExport**
- **Use CDFx**
- **Use CDAWlib routines within IDL**
  - From the IDL command line
  - w/ IDL_PATH already set to point to CDAWlib s/w

Other Details

Merging CDFs

- **CDFMerge**
  - Command line cdf toolkit program
  - Option to auto-rename variables and pointers values for uniqueness
- **Also possible using IDLmakecdf by**
  - Creating a skeletoncdf for the "merged" dataset and then reading in each of the "component" cdfs.
  - Example of a program that does this is at:
    - http://cdaweb.gsfc.nasa.gov/~kovalick/gen_xyzcomb.pro

Other Details

Overview:
Command line CDF Toolkit programs

- **cdfinquire: displays CDF s/w version of toolkit & default toolkit qualifiers**
- **cdfedit - allows nearly full editing capability to all CDF data files**
  - Often used to quickly edit values for attributes or add attributes, quickly see data values
- **cdfbrowse - same as cdfedit, but does not allow any changes to files**
- **cdfstats - displays statistics about the variables in a CDF.**
  - Including minimum/maximum values and whether or not the variable is monotonic.
  - Quickly gives a user a sense of what the cdf file has in it - variable and variable contents
- **cdfcompare - displays the differences in two (or more) CDFs.**
- **cdfexport - allows the contents of a CDF to be exported to ASCII**
  - We use this regularly to compress variables values in CDFs.
- **cdfconvert - update cdf version, change encoding or compression**
- **skeletontable - produces a skeleton table from a CDF.**
- **skeletoncdf - produces a CDF from a skeleton table.**
- **Also JAVA GUI based interfaces for (most functions) of cdfedit and cdfexport**

Other Details

Overview:
Major CDAWlib routines

- **READ_MYCDF - The function READ_MYCDF reads from one to many variables from one to many CDF files (in one dataset), and returns all data and metadata for these variables in a single structure**
- **PLOTMASTER - This function accepts from 1 to 10 structures of the type returned by READ_MYCDF, determines the plot type for each variable in each of the structures, and plots each (to either an X window or GIF file). Returns a 0 if plotting was successful, and a -1 if unsuccessful.**
- **LIST_MYSTRUCT - Given a "data structure" read with read_mycdf, LIST_mystruct generates an ascii listing of the data**
- **WRITE_MYCDF - This function accepts from 1 to 10 structures of the type returned by READ_MYCDF, produces a cdf file for each structure.**
  - Each have many keywords, please see the code for those

Other Details

Major CDAWlib Routines
read_mycdf

- **READ_MYCDF - The function READ_MYCDF reads from one to many variables from one to many CDF files (in one dataset),**

```
@compile_cdaweb
vnames=“”
a=strarr(1)
a(0)=‘wi_or_pre_20060913_v01’
buf1=read_myCDF(vnames, a, /all, /NODATASTRUCT)
```

- **Returns all data and metadata for these variables in a single structure of the form:**

  structure_name.variable_name.attribute_name.value

- **An example structure:**

```
IDL> help, /struct, buf1
** Structure <8382294>, 8 tags, length=6544, data length=6544, refs=1:
   EPOCH           STRUCT    -> <Anonymous> Array[1]
   GSE_LAT         STRUCT    -> <Anonymous> Array[1]
   GSE_LON         STRUCT    -> <Anonymous> Array[1]
   RADIUS          STRUCT    -> <Anonymous> Array[1]
   XYZ_GSE         STRUCT    -> <Anonymous> Array[1]
   XYZ_GSEO        STRUCT    -> <Anonymous> Array[1]
   CARTESIAN       STRUCT    -> <Anonymous> Array[1]
   XYZ_LBL         STRUCT    -> <Anonymous> Array[1]
```

Other Details

Major CDAWlib Routines and Sample Usage read_mycdf (cont)

Then to see what's in a variable structure:

```
IDL> help, /struct, buf1.epoch
** Structure <83cdc4c>, 44 tags, length=812, data length=812,
      refs=2:
   VARNAME        STRING    'Epoch'
   TITLE          STRING    'SSC ORBIT CDF'
   PROJECT        STRING    'SSC'
   DISCIPLINE     STRING    'Space Physics'
   SOURCE_NAME    STRING    'THEMIS1'
   DATA_VERSION   STRING    '1'
   ADID_REF       STRING    'NSSD'
   LOGICAL_FILE_ID STRING
      'THEMIS1_or_ssc_00000000_v01'
   DATA_TYPE      STRING    'OR>Orbit'
   DESCRIPTOR     STRING    'SSC>Satellite Situation Center
      Ephemeris'
   TEXT           STRING    Array[27]
   MODS           STRING    'Originated Feb. 16, 2006'
   LOGICAL_SOURCE  STRING    'themis1_or_ssc'
   LOGICAL_SOURCE_DESCRIPTION
                  STRING    'THEMIS1 GSE Positions @ 1min. res.'
   PI_NAME        STRING    'SSC/SSCWeb'
   PI_AFFILIATION  STRING    'NASA's GSFC'
   MISSION_GROUP   STRING    'THEMIS'
   INSTRUMENT_TYPE STRING    'Ephemeris'
   FIELDNAM       STRING    'Orbit Epoch Time'
   VALIDMIN       DOUBLE      6.1725370e+13
   VALIDMAX       DOUBLE      6.5733206e+13
   SCALEMIN       DOUBLE      6.1725370e+13
   SCALEMAX       DOUBLE      6.5733206e+13
   LABLAXIS       STRING    'Epoch'
   LABL_PTR_1     STRING    ''
   UNITS          STRING    'ms'
   FORMAT         STRING    ''
   DEPEND_0       STRING    ''
   DEPEND_1       STRING    ''
   MONOTON        STRING    'INCREASE'
   FILLVAL        FLOAT     -1.00000e+31
   VAR_TYPE       STRING    'support_data'
   DICT_KEY       STRING    ''
   CATDESC        STRING    ''
   VAR_NOTES      STRING    ''
   AVG_TYPE       STRING    ''
   VIRTUAL        STRING    ''
   FUNCT          STRING    ''
   COMPONENT_0    STRING    ''
   DISPLAY_TYPE   STRING    ''
   CDFTYPE        STRING    'CDF_EPOCH'
   CDFRECVARY     STRING    'VARY'
   IDLSIZE        LONG      Array[4]
   HANDLE         LONG            11
```

Note: The values for the variable are stored in an IDL handle (like a c pointer), called HANDLE, if you do not use the /NODATASTRUCT keyword, the data values will be stored in a structure member call "DAT"

Other Details

Subsetting and Supersetting CDFs: Using read_myCDF in IDL

- **Example 1(subset):**

```
;Given a cdf file covering a days worth of Cluster data
;Want to retain all variables and reduce the time range
@compile_cdaweb
;Specify whatever time range you'd like in to be read from old and written to new cdf)
start='2005/07/26 00:00:00'
stop='2005/07/26 01:59:59'
vnames=''
a=strarr(1)
a(0)='/home/cdaweb/data/cluster/cl/sp/fgm/2005/cl_sp_fgm_20050726_v01.cdf'
buf1 = read_myCDF([vnames, a, /all, /NODATASTRUCT, TSTART=start, TSTOP=stop)
s = write_mycdf(buf1,filename="cl_fgm_subsetbytime.cdf")
```

- **Example 2 (subset):**

```
;Given a cdf file covering a days worth of Cluster data
;Want to retain just variables needed for one specific variable and reduce the time range
@compile_cdaweb
;Specify whatever time range you'd like in to be read from old and written to new cdf)
start='2005/07/26 00:00:00'
stop='2005/07/26 01:59:59'
a=strarr(1)
a(0)='/home/cdaweb/data/cluster/cl/sp/fgm/2005/cl_sp_fgm_20050726_v01.cdf'
buf1 = read_myCDF([['B_xyz_gse__CL_SP_FGM'], a, /NODATASTRUCT, TSTART=start, TSTOP=stop)
s = write_mycdf(buf1,filename="cl_fgm_subsetbytimeandvar.cdf")
```

Sample Usage: PlotMaster

- **Once IDL_PATH has been set up according to the directions run IDL.**
- **This example will read all variables 3 days of data from one of the large (2 year) themis1 orbit files and produce orbit and time series plots for the appropriate variables.**

```
idl
@compile_cdaweb
start='2008/09/08 00:00:00'
stop='2008/09/11 00:00:00'
a=strarr(2)
a(0)='/home/cdaweb/data/0MASTERS/tha_or_ssc_00000000_v01.cdf'
a(1)='/home/cdaweb/data/themis/tha/ssc/2008/tha_or_ssc_20080901_v01.cdf'
vars=''
buf1 = read_myCDF(vars, a, /all, /NODATASTRUCT, TSTART=start, TSTOP=stop);to
    see the structure type help, /struct, buf1;help, /struct, buf1
s = plotmaster(buf1, /AUTO, /CDAWEB, PID='A0811', TSTART=start, TSTOP=stop,
    $ /GIF, OUTDIR='/home/kovalick/public_html/themis/', /SMOOTH, /SLOW)
exit
```

Other Details

Sample Usage (cont)

If we change this slightly to also read the corresponding THEMIS B orbit files, we'll get three gif files, one for each datatset's time-series plots and then combined orbit plots (one for each coordinate system). Note, do not run these examples back to back.

```
start='2008/09/08 00:00:00'
stop='2008/09/11 00:00:00'
a=strarr(2)
;THEMIS A - data read into buf1
a(0)='/home/cdaweb/data/0MASTERS/tha_or_ssc_00000000_v01.cdf'
a(1)='/home/cdaweb/data/themis/tha/ssc/2008/tha_or_ssc_20080901_v01.cdf'
vars=''
buf1 = read_myCDF(vars, a, /all, /NODATASTRUCT, TSTART=start, TSTOP=stop)
;THEMIS B - data read into buf2
a(0)='/home/cdaweb/data/0MASTERS/thb_or_ssc_00000000_v01.cdf'
a(1)='/home/cdaweb/data/themis/thb/ssc/2008/thb_or_ssc_20080901_v01.cdf'
vars=''
buf2 = read_myCDF(vars, a, /all, /NODATASTRUCT, TSTART=start, TSTOP=stop)
s = plotmaster(buf1,buf2, /AUTO, /CDAWEB, PID='AB0811', TSTART=start, TSTOP=stop,
    $ /GIF, OUTDIR='/home/kovalick/public_html/themis/', /SMOOTH, /SLOW)
exit
```

Other Details

Sample Usage (cont)

- **GIFS for first example can be viewed at:**
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/THA_OR_SSC_A0811_000.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_A0811_002.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_A0811_003.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_A0811_004.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_A0811_005.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_A0811_006.gif

- **GIFS for second example can be viewed at:**
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/THA_OR_SSC_AB0811_000.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/THB_OR_SSC_AB0811_001.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_AB0811_003.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_AB0811_004.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_AB0811_005.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_AB0811_006.gif
  - http://cdaweb.gsfc.nasa.gov/~kovalick/themis/ORBIT_AB0811_007.gif

Other Details

Sample Usage: LIST_MYSTRUCT and WRITE_MYCDF

- **Example of LIST_MYSTRUCT - reads the same three days, but just for two variables, creates an ascii listing with heading containing all global attribute information and data is formatted in columns according to the "format" variable attribute.**

```
start='2008/09/08 00:00:00'
stop='2008/09/11 00:00:00'
a=strarr(2)
a(0)='/home/cdaweb/data/0MASTERS/themis1_or_ssc_00000000_v01.cdf'
a(1)='/home/cdaweb/data/themis/themis1/or_ssc/themis1_or_ssc_20061019_v01.cdf'
buf = read_myCDF(['GSE_LAT','GSE_LON'], a, /NODATASTRUCT, TSTART=start,
    TSTOP=stop, /DEBUG)
s = LIST_mystruct(buf, TSTART=start, TSTOP=stop, /NOVATT, /NORV,$
FILE='/home/kovalick/public_html/THEMIS1_OR_SSC_5636.txt')
```

listing file can be seen at:
http://cdaweb.gsfc.nasa.gov/~kovalick/themis/THEMIS1_OR_SSC_5636.txt

- **Example of WRITE_MYCDF:**

From the above example, call write_mycdf instead of list_mystruct, e.g.

```
s = write_mycdf(buf,OUTDIR='/home/kovalick/public_html/',/autoname,$
/longtime,/bothtimes,/lowercase)
```

A Few References

- **The s/w distribution information and distribution site:**
  - http://spdf.gsfc.nasa.gov/CDAWlib.html
- **Relevant URLs and contact people...**
  - http://spdf.gsfc.nasa.gov/ - Space Physics Data Facility
  - http://cdf.gsfc.nasa.gov/ - CDF home page - CDF s/w distribution site
  - http://translators.gsfc.nasa.gov/ - CDF data format translation site
  - http://cdaweb.gsfc.nasa.gov/
    - Coordinated Data Analysis web site; produces plots, listing, cdfs, etc.
  - http://cdaweb.gsfc.nasa.gov/sp_test
    - Site containing test datasets in cdaweb such as the "splitvc" THEMIS datasets.
  - ftp://cdaweb.gsfc.nasa.gov/pub/CDAWlib/0MASTERS
    - Site containing all of our CDAWeb related "master" skeleton cdfs
- **SPDF contacts:**
  - Dr. Bob McGuire - head of everything
  - David Han, Mike Liu - CDF support
  - Tami Kovalick, Rita Johnson - CDAWeb/CDAWlib, CDFx support

What Else Can We Do for You Today?

- **Questions?**
- **Specific problem situations or concerns?**
- **Other topics for Discussion?**
  - Other CDF capabilities or tools that would be useful?
  - How should RBSP data be displayed through CDAWlib/CDAWeb?
  - SSCWeb capabilities and outputs
  - Demo: The 4-D Interactive Orbit Viewer (TIPSOD)

Added Bonus Demo:
The 4-D Interactive Orbit Viewer (TIPSOD)

Questions and Topics for Conversation

- **What is the present range of computers and operating systems of primary concern to the RBSP team?**

Backup and Preliminary

Software Installation Summary

	URL	Windows	Mac OSX	UNIX/ LINUX
CDF 3.3	http://cdf.gsfc.nasa.gov	Self-installation file	Apple Installer	tar/gzip
CDAWlib/ CDFX	ftp://cdaweb.gsfc.nasa.gov/pub/CDAWlib/	FTP file copies or tar/gzip to IDL directory (use UNIX subdir)	tar/gzip to IDL directory (use UNIX subdir)	tar/gzip to IDL directory
SKTEditor	http://sscweb.gsfc.nasa.gov/skteditor/	WebStart	WebStart	WebStart
makeCDF	ftp://nssdcftp.gsfc.nasa.gov/selected_software/makecdf/	Source and make file for local compile	Source and make file for local compile	Source and make file for local compile

Open Issues for Session

- **What are the issues in adding SPASE keywording to SKTEditor?**
  - Many keywords and complex hierarchy for files and variables
- **What's the working status of CDFX?**
- **Set aside time (key a viewgraph) to talk about displays that THEMIS would like to see enabled**
  - Need this time and other times for THEMIS to talk to us
- **Crosscheck list from Vassilis**
  - Availability of multiple paths/utilities to same goal.
  - CDF utilities (XP, MacOS, Unix flavors) cdfwalk, cdfread etc.
  - How to create a CDF with simple IDL data/structure
  - How to read a CDF pre-written by someone
  - How to access CDF data files
  - read_myCDF, write_myCDF and other utilities
  - How to modify attributes to make them more accurate to my instrument, particular data I have.
  - How do I write a time-section (10min section) of a CDF into a different file, with all attributes preserved?
  - CDF versions and IDL compatibility
  - Plotting (IDL) tools available in SPDF for CDFs, standard formats, how do you interface with new tools we can provide.

Other Major CDAWlib Routines

- **PLOTMASTER - This function accepts from 1 to 10 structures of the type returned by READ_MYCDF, determines the plot type for each variable in each of the structures, and plots it (to either an X window or GIF file). Returns a 0 if plotting was successful, and a -1 if unsuccessful.**
- **LIST_MYSTRUCT - Given a "data structure" read with read_mycdf, LIST_mystruct generates an ascii listing of the data.**
- **WRITE_MYCDF - This function accepts from 1 to 10 structures of the type returned by READ_MYCDF, produces a cdf file for each structure**

SPDF and VxOs (ViRBO) and RBSP

- XXX

CDFExport

- **Use CDF toolkit (CDFExport) to make ASCII or XML**
  - Generic: works for all CDFs
- **Example**
  - Start cdfexport for a given data cdf file
    - E.g. cdfexport wi_or_pre_20060913_v01
  - Use the <tab> key to see all available selection windows.
  - De-select variables not needed in output
  - Use <Alt> <A> to b
    - Select output to, another CDF.
    - Hit the return key and enter the output file name

Recheck and Shorten

  - Specific instructions for subsetting:

    Go to the window that has "Minimum" and "Maximum" fields displayed.
    - Move the curser over to the Min or Max field for Epoch variable and
    - Hit enter key to allow changing their initial values to the range to extract.
    - "Filter" field will change from "no" to "Yes" once you change the Min/Max field.