	[Error]
BAD_DIM_INDEX	One or more dimension index is out of range. A valid value must be specified regardless of the dimension variance. Note also that the combination of dimension index, count, and interval must not specify an element beyond the end of the dimension. [Error]
BAD_DIM_INTERVAL	Illegal dimension interval specified. Dimension intervals must be at least one (1). [Error]
BAD_DIM_SIZE	Illegal dimension size specified. A dimension size must be at least one (1). [Error]
BAD_ENCODING	Unknown data encoding specified. The CDF encodings are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
BAD_ENTRY_NUM	Illegal attribute entry number specified. Entry numbers must be at least zero (0) for C applications and at least one (1) for Fortran applications. [Error]
BAD_FNC_OR_ITEM	The specified function or item is illegal. Check that the proper number of arguments are specified for each operation being performed. Also make sure that `NULL_` is specified as the last operation. [Error]
BAD_FORMAT	Unknown format specified. The CDF formats are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
BAD_INITIAL_RECS	An illegal number of records to initially write has been specified. The number of initial records must be at least one (1). [Error]
BAD_MAJORITY	Unknown variable majority specified. The CDF variable majorities are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
BAD_MALLOC	Unable to allocate dynamic memory — system limit reached. Contact CDF User Support if this error occurs. [Error]
BAD_NEGtoPOSfp0_MODE	An illegal `-0.0` to `0.0` mode was specified. The `-0.0` to `0.0` modes are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
BAD_NUM_DIMS	The number of dimensions specified is out of the allowed range. Zero (0) through `CDF_MAX_DIMS` dimensions are allowed. If more are needed, contact CDF User Support. [Error]
BAD_NUM_ELEMS	The number of elements of the data type is illegal. The number of elements must be at least one (1). For variables with a non-character data type, the number of elements must always be one (1). [Error]
BAD_NUM_VARS	Illegal number of variables in a record access operation. [Error]

BAD_READONLY_MODE	Illegal read-only mode specified. The CDF read-only modes are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
BAD_REC_COUNT	Illegal record count specified. A record count must be at least one (1). [Error]
BAD_REC_INTERVAL	Illegal record interval specified. A record interval must be at least one (1). [Error]
BAD_REC_NUM	Record number is out of range. Record numbers must be at least zero (0) for C applications and at least one (1) for Fortran applications. Note that a valid value must be specified regardless of the record variance. [Error]
BAD_SCOPE	Unknown attribute scope specified. The attribute scopes are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
BAD_SCRATCH_DIR	An illegal scratch directory was specified. The scratch directory must be writeable and accessable (if a relative path was specified) from the directory in which the application has been executed. [Error]
BAD_SPARSEARRAYS_PARM	An illegal sparse arrays parameter was specified. [Error]
BAD_VAR_NAME	Illegal variable name specified. Variable names must contain at least one character and each character must be printable. [Error]
BAD_VAR_NUM	Illegal variable number specified. Variable numbers must be zero (0) or greater for C applications and one (1) or greater for Fortran applications. [Error]
BAD_zMODE	Illegal zMode specified. The CDF zModes are defined in `cdf.h` for C applications and in `cdf.inc` for Fortran applications. [Error]
CANNOT_ALLOCATE_RECORDS	Records cannot be allocated for the given type of variable (e.g., a compressed variable). [Error]
CANNOT_CHANGE	Because of dependencies on the value, it cannot be changed. Some possible causes of this error follow: 1. Changing a CDF's data encoding after a variable value (including a pad value) or an attribute entry has been written. 2. Changing a CDF's format after a variable has been created or if a compressed single-file CDF. 3. Changing a CDF's variable majority after a variable value (excluding a pad value) has been written. 4. Changing a variable's data specification after a value (including the pad value) has been written to

that variable or after records have been allocated for that variable.

5. Changing a variable's record variance after a value (excluding the pad value) has been written to that variable or after records have been allocated for that variable.
6. Changing a variable's dimension variances after a value (excluding the pad value) has been written to that variable or after records have been allocated for that variable.
7. Writing "initial" records to a variable after a value (excluding the pad value) has already been written to that variable.
8. Changing a variable's blocking factor when a compressed variable and a value (excluding the pad value) has been written or when a variable with sparse records and a value has been accessed.
9. Changing an attribute entry's data specification where the new specification is not equivalent to the old specification.

`CANNOT_COMPRESS` The CDF or variable cannot be compressed. For CDFs, this occurs if the CDF has the multi-file format. For variables, this occurs if the variable is in a multi-file CDF, values have been written to the variable, or if sparse arrays have already been specified for the variable. [Error]

`CANNOT_SPARSEARRAYS` Sparse arrays cannot be specified for the variable. This occurs if the variable is in a multi-file CDF, values have been written to the variable, records have been allocated for the variable, or if compression has already been specified for the variable. [Error]

`CANNOT_SPARSERECORDS` Sparse records cannot be specified for the variable. This occurs if the variable is in a multi-file CDF, values have been written to the variable, or records have been allocated for the variable. [Error]

`CDF_CLOSE_ERROR` Error detected while trying to close CDF. Check that sufficient disk space exists for the dotCDF file and that it has not been corrupted. [Error]

`CDF_CREATE_ERROR` Cannot create the CDF specified — error from file system. Make sure sure that sufficient privilege exists to create the dotCDF file in the disk/directory location specified and that an open file quota has not already been reached. [Error]

`CDF_DELETE_ERROR` Cannot delete the CDF specified — error from file system. Unsufficient privileges exist the delete the CDF file(s). [Error]

`CDF_EXISTS`	The CDF named already exists — cannot create it. The CDF library will not overwrite an existing CDF. [Error]
`CDF_INTERNAL_ERROR`	An unexpected condition has occurred in the CDF library. Report this error to CDFsupport. [Error]
`CDF_NAME_TRUNC`	CDF pathname truncated to `CDF_PATHNAME_LEN` characters. The CDF was created but with a truncated name. [Warning]
`CDF_OK`	Function completed successfully.
`CDF_OPEN_ERROR`	Cannot open the CDF specified — error from file system. Check that the dotCDF file is not corrupted and that sufficient privilege exists to open it. Also check that an open file quota has not already been reached. [Error]
`CDF_READ_ERROR`	Failed to read the CDF file — error from file system. Check that the dotCDF file is not corrupted. [Error]
`CDF_WRITE_ERROR`	Failed to write the CDF file — error from file system. Check that the dotCDF file is not corrupted. [Error]
`COMPRESSION_ERROR`	An error occured while compressing a CDF or block of variable records. This is an internal error in the CDF library. Contact CDF User Support. [Error]
`CORRUPTED_V2_CDF`	This Version 2 CDF is corrupted. An error has been detected in the CDF's control information. If the CDF file(s) are known to be valid, please contact CDF User Support. [Error]
`DECOMPRESSION_ERROR`	An error occured while decompressing a CDF or block of variable records. The most likely cause is a corrupted dotCDF file. [Error]
`DID_NOT_COMPRESS`	For a compressed variable, a block of records did not compress to smaller than their uncompressed size. They have been stored uncompressed. This can result if the blocking factor is set too low or if the characteristics of the data are such that the compression algorithm choosen is unsuitable. [Informational]
`EMPTY_COMPRESSED_CDF`	The compressed CDF being opened is empty. This will result if a program which was creating/modifying the CDF abnormally terminated. [Error]
`END_OF_VAR`	The sequential access current value is at the end of the variable. Reading beyond the end of the last physical value for a variable is not allowed (when performing sequential access). [Error]
`FORCED_PARAMETER`	A specified parameter was forced to an acceptable value (rather than an error being returned). [Warning]
`PC_OVERFLOW`	An operation involving a buffer greater than 64k bytes in size has been specified. [Error]
`ILLEGAL_FOR_SCOPE`	The operation is illegal for the attribute's scope. For example, only gEntries may be written for gAttributes — not rEntries or zEntries. [Error]

`ILLEGAL_IN_zMODE`	The attempted operation is illegal while in zMode. Most operations involving rVariables or rEntries will be illegal. [Error]
`ILLEGAL_ON_V1_CDF`	The specified operation (i.e., opening) is not allowed on Version 1 CDFs. [Error]
`MULTI_FILE_FORMAT`	The specified operation is not applicable to CDFs with the multi-file format. For example, it does not make sense to inquire indexing statistics for a variable in a multi-file CDF (indexing is only used in single-file CDFs). [Informational]
`NA_FOR_VARIABLE`	The attempted operation is not applicable to the given variable. [Warning]
`NEGATIVE_FP_ZERO`	One or more of the values read/written are `-0.0` (an illegal value on VAXes and DEC Alphas running OpenVMS). [Warning]
`NO_ATTR_SELECTED`	An attribute has not yet been selected. First select the attribute on which to perform the operation. [Error]
`NO_CDF_SELECTED`	A CDF has not yet been selected. First select the CDF on which to perform the operation. [Error]
`NO_DELETE_ACCESS`	Deleting is not allowed (read-only access). Make sure that delete access is allowed on the CDF file(s). [Error]
`NO_ENTRY_SELECTED`	An attribute entry has not yet been selected. First select the entry number on which to perform the operation. [Error]
`NO_MORE_ACCESS`	Further access to the CDF is not allowed because of a severe error. If the CDF was being modified, an attempt was made to save the changes made prior to the severe error. In any event, the CDF should still be closed. [Error]
`NO_PADVALUE_SPECIFIED`	A pad value has not yet been specified. The default pad value is currently being used for the variable. The default pad value was returned. [Informational]
`NO_STATUS_SELECTED`	A CDF status code has not yet been selected. First select the status code on which to perform the operation. [Error]
`NO_SUCH_ATTR`	The named attribute was not found. Note that attribute names are case-sensitive. [Error]
`NO_SUCH_CDF`	The specified CDF does not exist. Check that the pathname specified is correct. [Error]
`NO_SUCH_ENTRY`	No such entry for specified attribute. [Error]
`NO_SUCH_RECORD`	The specified record does not exist for the given variable. [Error]
`NO_SUCH_VAR`	The named variable was not found. Note that variable names are case-sensitive. [Error]
`NO_VAR_SELECTED`	A variable has not yet been selected. First select the variable on which to perform the operation. [Error]

NO_VARS_IN_CDF	This CDF contains no rVariables. The operation performed is not applicable to a CDF with no rVariables. [Informational]
NO_WRITE_ACCESS	Write access is not allowed on the CDF file(s). Make sure that the CDF file(s) have the proper file system privileges and ownership. [Error]
NOT_A_CDF	Named CDF is corrupted or not actually a CDF. This can also occur if an older CDF distribution is being used to read a CDF created by a more recent CDF distribution. Contact CDF User Support if you are sure that the specified file is a CDF that should be readable by the CDF distribution being used. CDF is backward compatible but not forward compatible. [Error]
PRECEEDING_RECORDS_ALLOCATED	Because of the type of variable, records preceeding the range of records being allocated were automatically allocated as well. [Informational]
READ_ONLY_DISTRIBUTION	Your CDF distribution has been built to allow only read access to CDFs. Check with your system manager if you require write access. [Error]
READ_ONLY_MODE	The CDF is in read-only mode — modifications are not allowed. [Error]
SCRATCH_CREATE_ERROR	Cannot create a scratch file — error from file system. If a scratch directory has been specified, ensure that it is writable. [Error]
SCRATCH_DELETE_ERROR	Cannot delete a scratch file — error from file system. [Error]
SCRATCH_READ_ERROR	Cannot read from a scratch file — error from file system. [Error]
SCRATCH_WRITE_ERROR	Cannot write to a scratch file — error from file system. [Error]
SINGLE_FILE_FORMAT	The specified operation is not applicable to CDFs with the single-file format. For example, it does not make sense to close a variable in a single-file CDF. [Informational]
SOME_ALREADY_ALLOCATED	Some of the records being allocated were already allocated. [Informational]
TOO_MANY_PARMS	A type of sparse arrays or compression was encountered having too many parameters. This could be causes by a corrupted CDF or if the CDF was created/modified by a CDF distribution more recent than the one being used. [Error]
TOO_MANY_VARS	A multi-file CDF on a PC may contain only a limited number of variables because of the 8.3 file naming convention of MS-DOS. This consists of 100 rVariables and 100 zVariables. [Error]
UNKNOWN_COMPRESSION	An unknown type of compression was specified or encountered. [Error]
UNKNOWN_SPARSENESS	An unknown type of sparseness was specified or encountered. [Error]

`UNSUPPORTED_OPERATION`	The attempted operation is not supported at this time. [Error]
`VAR_ALREADY_CLOSED`	The specified variable is already closed. [Informational]
`VAR_CLOSE_ERROR`	Error detected while trying to close variable file. Check that sufficient disk space exists for the variable file and that it has not been corrupted. [Error]
`VAR_CREATE_ERROR`	An error occurred while creating a variable file in a multi-file CDF. Check that a file quota has not been reached. [Error]
`VAR_DELETE_ERROR`	An error occurred while deleting a variable file in a multi-file CDF. Check that sufficient privilege exist to delete the CDF files. [Error]
`VAR_EXISTS`	Named variable already exists - cannot create or rename. Each variable in a CDF must have a unique name (rVariables and zVariables can not share names). Note that trailing blanks are ignored by the CDF library when comparing variable names. [Error]
`VAR_NAME_TRUNC`	Variable name truncated to `CDF_VAR_NAME_LEN` characters. The variable was created but with a truncated name. [Warning]
`VAR_OPEN_ERROR`	An error occurred while opening variable file. Check that sufficient privilege exists to open the variable file. Also make sure that the associated variable file exists. [Error]
`VAR_READ_ERROR`	Failed to read variable as requested — error from file system. Check that the associated file is not corrupted. [Error]
`VAR_WRITE_ERROR`	Failed to write variable as requested — error from file system. Check that the associated file is not corrupted. [Error]
`VIRTUAL_RECORD_DATA`	One or more of the records are virtual (never actually written to the CDF). Virtual records do not physically exist in the CDF file(s) but are part of the conceptual view of the data provided by the CDF library. Virtual records are described in the Concepts chapter in the CDF User's Guide. [Informational]

Appendix D

Release Notes

D.1 Introduction

This appendix should provide you with all of the information you need to know to begin using CDF V2.6 assuming that you are already familar with CDF V2.5. For additional information contact CDF User Support.

D.2 Supported Systems

CDF V2.6 is currently supported on the following computers/operating systems.

1. VAX (OpenVMS & POSIX shell)
2. Sun (SunOS & SOLARIS)
3. DECstation (ULTRIX)
4. Silicon Graphics Iris & Power Series (IRIX)
5. IBM RS6000 series (AIX)
6. HP 9000 series (HP-UX)
7. PC (MS-DOS, Windows 3.x, Windows NT/95, Linux, & QNX)
8. NeXT (Mach)
9. DEC Alpha (OSF/1 & OpenVMS)
10. Macintosh (MacOS 7.0)

D.3 Changes for CDF V2.6

A number of changes have been made for the CDF V2.6 distribution...

1. Compression support has been added to the CDF library. Several compression algorithms are available and may be applied to the entire CDF or to individual variables in a CDF. The available compression algorithms include run-length encoding of zeros, Huffman, adaptive Huffman, and Gnu's ZIP.
2. Support for sparse records has been added for variables. When a variable is specified as having sparse records, only those records actually written to the variable will be physically stored in the CDF. This allows gaps of missing records to exist rather than physically writing those records with some sort of pad value.
3. A consequence of the compression and sparse records capability of CDF V2.6 is that previous CDF releases will not be able to read CDFs created or modified by CDF V2.6.
4. Individual records may now be deleted from a variable in a CDF.
5. Individual blocks of records may now be allocated as needed for uncompressed variables. Previously, records could only be allocated once and had to start at the first record for a variable.
6. A more efficient hierarchical indexing scheme has been implemented in an effort to optimize the access to sparse and/or compressed records of a variable. This indexing scheme is transparent to an application.
7. The "cache" qualifier used with many of the CDF toolkit programs now allows the specification of the cache sizes to be used for the scratch files maintained by the CDF library. The CDF library uses scratch files in support of compression and sparse records. The directory in which these scratch files are created may be specified by an application program via the Internal Interface or with the logical name `CDF$TMP` (on VMS systems) and environment variable `CDF_TMP` (on UNIX and MS-DOS systems).
8. The functionality of CDFlist and CDFwalk has been added to CDFexport. CDFlist and CDFwalk will remain in the CDF distribution but are no longer supported. The use of CDFexport is strongly recommended.
9. Compression and sparseness qualifiers have been added to CDFconvert which allow the compression of a CDF or its variables and the sparseness of its variables to be changed during a conversion.
10. Compression and sparseness display/selection has been added to the CDFbrowse/CDFedit and CDF-export toolkit programs.
11. The Internal Interface can be used to inquire the number of records written to a variable having sparse records (in addition to the maximum record number written). The existence of individual records may also be determined.
12. A zMode qualifier has been added to SkeletonCDF which allows zVariables to be created from the rVariables section of a skeleton table.
13. An option has been added to CDFbrowse/CDFedit with allows the character string entries of a gAttribute to be displayed/edited as if they were a text file.
14. Skeleton tables can now contain values for record-variant (RV) variables.
15. 4-byte integers are now the default when using Symantec's Think C on the Macintosh. This was necessary to support some of the compression routines in the CDF library.

16. For CDF's IDL interface, a new way to include the needed CDF parameter constants has been added. Entering `@cdf0x.pro` on the IDL command line (or in a function/procedure) causes a set of structure variables to be created and initialized with the CDF parameter constants.
17. The toolkit programs are now installed with their full names on UNIX systems.
18. CDFskeleton was renamed SkeletonCDF. SkeletonTable creates a skeleton table. SkeletonCDF creates a (skeleton) CDF.
19. CDFedit in browse-only mode (CDFbrowse) was modified to look more like a browser.
20. A variable's "extend records" now referred to as it's "blocking factor."
21. The "definitions" files used on UNIX systems are now installed in the `<path>/bin` directory. Also, the online help files are now installed in the `<path>/lib/cdf/help` directory. The location of `<path>` is specified by the system administrator who installs the CDF distribution.
22. An "about" qualifier (or selection) was added to all CDF toolkit programs which allows the inquiry of the CDF distribution from which the program originated.
23. Routines were added to the CDF library which allow `CDF_EPOCH` values to be encoded for display using custom formats specified by the user.
24. The CDF V2.6 distribution was ported to QNX (a flavor of UNIX for the PC).
25. The CDF V2.6 distribution was ported to Windows NT/95.

Appendix E

Glossary

`AHUFF`	The Adaptive Huffman compression algorithm.
allocated records	For uncompressed variables in a single-file CDF it is possible for an application to allocate records before they are written. This has the advantage of reducing the indexing overhead in the dotCDF file which will improve performance when accessing a variable. An application would generally then write to the records that were allocated.
attribute	A CDF object with which entries of metadata are associated.
big-endian	The byte ordering in which the most significant byte (MSB) is stored in the lowest memory location.
blocking factor	For a standard variable (in a single-file CDF), the blocking factor is the minimum number of records actually allocated when a new record is written. More records may be allocated than are actually needed in order to keep the variable's records as contiguous as possible (with the assumption that the records will eventually be written). For a compressed variable in a single-file CDF, the blocking factor is the maximum number of records per compressed block. For an uncompressed variable having sparse records in a single-file CDF, the blocking factor is the number of records allocated in the staging scratch file. For this type of variable the staging scratch file is used to optimize the indexing in the dotCDF file by storing sequential records contiguously when possible. Blocking factors are not applicable to variables in multi-file CDFs.
caching	The method used by the CDF library to improve performance when accessing a file. An attempt is made to keep commonly accessed blocks of the file in memory rather than repeatedly reading them from or writing them to disk.

CDF	This term is used in more than one way... 1. The actual files that contain your data/metadata. For example: The CDF library must be used to create a "CDF." 2. The software distribution containing the CDF library, include files, and toolkit. For example: We like using "CDF" to store our data.
CDF base name	The pathname of a CDF minus the extension (or extensions if a multi-file CDF).
CDF distribution	The directory of software consisting of the CDF library, include files, and toolkit.
CDF library	The software library that is used to access a CDF.
CDF toolkit	A set of utility programs which ease the creation, modification, and verification of CDFs.
CDFbrowse	A read-only version of CDFedit.
CDFedit	A CDF toolkit program that allows the display and modification of a CDF's contents.
CDFexport	A CDF toolkit program that allows the (possibly filtered) contents of a CDF to be exported to the terminal screen, a text file, or another CDF.
CDFstats	A CDF toolkit program that generates a report containing various statistics about a CDF's variables.
CDFcompare	A CDF toolkit program that reports any differences between two CDFs.
CDFconvert	A CDF toolkit program that allows various overall properties of a CDF to be changed (in a newly created CDF).
CDFinquire	A CDF toolkit program that displays the version of the CDF distribution being used, many of the configurable parameters, and the default CDF toolkit qualifiers/options.
`CDF_OK`	A completion status code indicating unqualified success.
`cdf.h`	An include file used in C applications.
`cdf.inc`	An include file used in Fortran applications.
`cdfmsf.inc`	An include file used in Microsoft Fortran applications.
column-major	The variable majority where the first index of a multidimensional array of values increments the fastest.
compression	The process of encoding a group of bytes into a smaller group of bytes, storing the smaller group of bytes, and then decoding the smaller group of bytes back to the original group of bytes. CDF allows both a CDF and/or individual variables to

be compressed when stored.

conceptual view — The way that values along a dimension having a variance of `NOVARY` are made to appear as if they do actually exist (only one value is actually physically stored). This also applies to records beyond the last record actually stored. The conceptual view of a variable consists of "virtual" records and values (in addition to the physical records and values actually stored).

current — When the Internal Interface is used, current objects/states are those items affected when an operation is performed. For example, a current CDF is selected and then any operation performed involving a CDF is performed on that CDF (until a different current CDF is selected).

data specification — For a variable or attribute entry the data type and number of elements of that data.

data type — For a variable or attribute entry, the type of data being stored (e.g., integer, floating-point, character).

decoding — The integer/floating-point representation of data values passed to an application by the CDF library as they are read from a CDF. This is independent of the way the data values are physically stored in the CDF.

dimension variance — The property of a variable that specifies whether or not the values along a dimension change or stay the same.

dimensionality — The number of dimensions and the dimension sizes for the rVariables or a zVariable.

dotCDF file — A file having an extension of `.cdf` (or `.CDF` if the operating system being used prefers uppercase). For a single-file CDF this will be the only file. For a multi-file CDF this file will exist along with zero or more variable files (depending on the number of variables in the CDF).

encoding — The integer/floating-point representation of the data values physically stored in a CDF.

entry — A CDF object in which metadata is stored. An entry is associated with an attribute.

error code — A status code indicating that a fatal condition was encountered. The operation was aborted.

format — In reference to a CDF, the way in which files are used to store the CDF's control/data/metadata. This may be single-file or multi-file.

full-physical record — A variable record consisting of values exactly as physically stored in the CDF.

gAttribute — A global scoped attribute.

gEntry — An entry for a gAttribute.

global scope	Global scope indicates that an attribute describes some property of the entire CDF.
`GZIP`	The Gnu ZIP compression algorithm.
host decoding	The decoding of the computer currently being used.
host encoding	The encoding of the computer currently being used.
`HUFF`	The Huffman compression algorithm.
hyper access	A variable access method in which multiple records/values are read/written for a variable.
IDL Interface	A set of functions callable from within IDL (Interactive Data Language) that allow access to CDFs. The CDF distribution contains an IDL interface in addition to the CDF interface built into IDL by Research Systems, Inc. (RSI — the distributors of IDL).
IEEE 754	The floating-point representation of XDR.
include file	A file, included by a C or Fortran application, that contains constants recognized by the CDF library pertaining to various aspects of CDF objects/states.
indexing	The method used in a single-file CDF to keep track of where each variable's records are located.
informational code	A status code indicating success but providing some additional information that may be of interest.
Internal Interface	A set of routines in the CDF library callable from C and Fortran applications that provide all types of access to CDFs.
item	When the Internal Interface is used, an object or state on which a function is performed.
`libcdf.a`	The CDF library on UNIX systems.
`LIBCDF.LIB`	The CDF library on MS-DOS systems.
`LIBCDF.OLB`	The CDF library on VMS and OpenVMS systems.
little-endian	The byte ordering in which the least significant byte (LSB) is stored in the lowest memory location.
majority	The order in which the values of a multidimensional array are stored. This may be either row-major or column-major.
metadata	Data about data. A CDF stores metadata using attributes and attribute entries.
monotonicity	The property of a variable that specifies whether or not that variable's values increment or decrement (or neither) along a dimension or from record to record.
multi-file	A CDF format. Multi-file CDFs consist of one file for control/metadata and one file per variable of data.

multiple variable access	A variable access method in which one full-physical record is read/written for each of one or more variables.
network encoding	The encoding that uses the XDR representation.
`NOVARY`	A record/dimension variance indicating that the values do not change from record to record or along a dimension.
NRV variable	Non-record variant variable. A variable whose values do not change from record to record (a record variance of `NOVARY`).
NSSDC	National Space Science Data Center.
number of elements	For a variable the number of instances of the data type at each value. For an attribute entry the number of instances of the data type for that entry.
object	When the Internal Interface is used, an item that exists and may be accessed/manipulated (e.g., a CDF or variable).
operation	When the Internal Interface is used, a function performed on an item (e.g., creating or writing).
pad value	A value written to a variable by the CDF library in those cases where a physical record must be written but not all of its values have been specified by an application. For example, when a single value is written to a new record, all of the other values are written using the pad value.
physical record	A variable record actually stored in a CDF.
physical value	A variable value actually stored in a CDF.
read-only	A mode of the CDF library in which modifications to a CDF are not allowed.
record variance	The property of a variable that specifies whether or not its values change from record to record.
reserve percentage	For a compressed variable, the reserve percentage specifies how much additional space to allocate in the dotCDF file when a compressed block of records is initially written. A value of `0` (zero) causes no reserve space to be allocated. Values from `1` to `100` cause at least that percentage of the uncompressed size to be allocated. Values greater than `100` cause that percentage of the compressed size to be allocated (but not exceeding the uncompressed size).
rEntry	An entry for a vAttribute corresponding to an rVariable.
`RLE`	A run-length encoding compression algorithm. Currently, the only type of `RLE` compression supported is the run-length encoding of bytes containing zero.
row-major	The variable majority where the last index of a multidimensional array of values increments the fastest.
RV variable	Record variant variable. A variable whose values change from

	record to record (a record variance of `VARY`).
rVariable	"R" variable. A CDF object in which data values are stored. All rVariables have the same dimensionality.
scratch directory	The directory in which the CDF library creates scratch files. This directory may be specified by a user or an application.
scratch files	Temporary files used by the CDF library to minimize core memory usage.
scope	The intended use for an attribute. This may be global scope or variable scope.
sequential access	A variable access method in which values are read/written in the physical order in which they are stored in the CDF.
single-file	A CDF format. Single-file CDFs are entirely contained within one file.
single value access	A variable access method in which exactly one value is read/written for a variable.
skeleton CDF	A CDF consisting of only control, metadata, and NRV variable values.
skeleton table	A text file containing the control, metadata, and traditionally only the NRV variable values of a CDF. RV variable values may now also be included in a skeleton table. A skeleton table is read by the SkeletonCDF toolkit program which then creates the corresponding skeleton CDF (or complete CDF if the RV variable values also existed in the skeleton table). The SkeletonTable toolkit program can be used to create a skeleton table from a CDF.
SkeletonCDF	A CDF toolkit program which creates a skeleton CDF based on a skeleton table. A complete CDF may also be created if the skeleton table contained RV variable values in addition to NRV variable values.
SkeletonTable	A CDF toolkit program which creates a skeleton table from a CDF.
sparse arrays	A property assigned to a variable indicating that only those values written to a record should be stored. Because the values of a variable record can be written in any order this allows gaps of missing values to occur.
sparse records	A property assigned to a variable indicating that only those records written to the variable should be stored. Because the records of a variable can be written in any order this allows gaps of missing records to occur.
Standard Interface	A set of routines in the CDF library callable from C and Fortran applications that provide access to a commonly used subset of the capabilities of the Internal Interface. This interface

	was defined with the release of CDF V2.0 and has not changed since. New features since that time are available only through the Internal Interface (e.g., zVariables and zMode).
standard variable	A variable in a single-file CDF that is not compressed nor has sparse records or arrays.
state	When the Internal Interface is used, a property pertaining to an object (e.g., a CDF's format or variable's data specification).
status code	The result of a CDF function/subroutine call. `CDF_OK` indicates unqualified success.
status handler	A function/subroutine that acts upon a status code received from the CDF library.
variable file	In a multi-file CDF, these are the files containing the data values for each variable (in one file per variable). These files are named using the CDF's base name with extensions of '`.v0`', '`.v1`', and so on for rVariables and '`.z0`', '`.z1`', and so on for zVariables.
variable scope	Variable scope indicates that an attribute describes some property of each variable.
variance (dimension)	The property of a variable that specifies whether or not the values along a dimension change or stay the same.
variance (record)	The property of a variable that specifies whether or not its values change from record to record.
`VARY`	A record/dimension variance indicating that the values change from record to record or along a dimension.
vAttribute	A variable scoped attribute.
virtual record	A variable record that is not actually stored in a CDF but does appear in the conceptual view of the CDF. Virtual records would be those records beyond the first record of an NRV variable and those records beyond the last record actually written to an RV variable.
virtual value	A variable value this is not actually stored in a CDF but does appear in the conceptual view of the CDF. Virtual values would be those values beyond the first value of a dimension whose variance is `NOVARY`.
warning code	A status code indicating that the operation did complete but probably not as expected.
XDR	External Data Representation. An integer/floating-point representation using big-endian byte ordering and the IEEE 754 floating-point representation.
zEntry	An entry for a vAttribute corresponding to a zVariable.
zMode	A mode of the CDF library in which rVariables are made to

appear as zVariables (and rEntries appear as zEntries).

zVariable	"Z" variable. A CDF object in which data values are stored. zVariables can have dimensionalities that are different than those of the rVariables (and each other).

Index