Chapter 1

Introduction to HDF

1.1 Chapter Overview

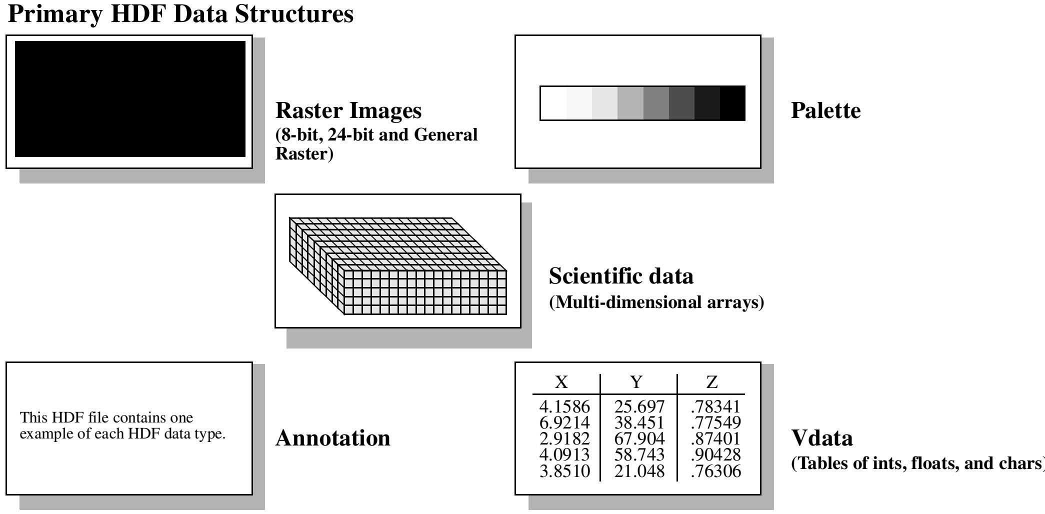
This chapter provides a general description of HDF including its indigenous object structures, application programming interface and accompanying command-line utilities. It also provides a short discussion of HDF's original purpose and philosophy and concludes with a list of the platforms HDF has been ported to.

1.2 What is HDF?

The ***Hierarchical Data Format***, or ***HDF***, is a multi-object file format for sharing scientific data in a distributed environment. HDF was created at the National Center for Supercomputing Applications to serve the needs of diverse groups of scientists working on projects in many fields. HDF was designed to address many requirements for storing scientific data, including:

- Support for the types of data and metadata commonly used by scientists.
- Efficient storage of and access to large data sets.
- Platform independence.
- Extensibility for future enhancements and compatibility with other standard formats.

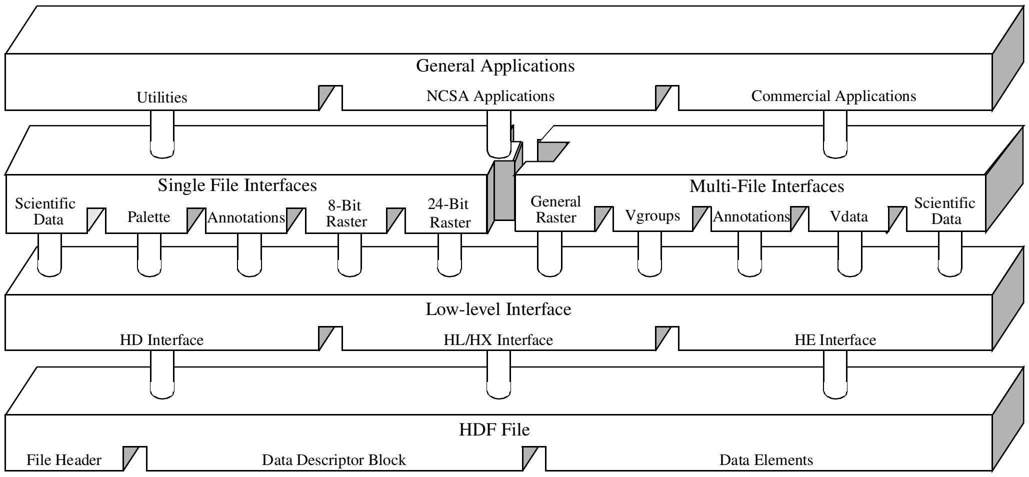
FIGURE 1a **Primary HDF Data Structures**



Five of the primary data objects supported by HDF are illustrated in Figure 1a. A sixth, the vgroup object, does not contain data and is designed for the purpose of grouping the other five primary data objects within an HDF file.

HDF is more than a file format. It also consists of supporting software that make it easy to store, retrieve, visualize, analyze, and manage data in HDF files. HDF can be viewed as several interactive levels as illustrated in the following figure.

FIGURE 1b **The Three Levels of Interaction with the HDF File Format**



At its lowest level, HDF is a physical file format for storing scientific data. At its highest level, HDF is a collection of utilities and applications for manipulating, viewing, and analyzing data in HDF files. Between these levels, HDF is a software library that provides high-level APIs and a low-level data interface.

The basic interface layer, or the ***low-level interface***, is reserved for software developers. It was designed for direct file I/O of data streams, error handling, memory management, and physical storage. It is essentially a software toolkit for skilled programmers who wish to make HDF do something more than what is currently available through the higher-level interfaces. Low-level routines are only available in C.

The HDF ***application programming interfaces,*** or ***APIs,*** include several independent sets of routines, with each set specifically designed to simplify the process of storing and accessing one type of data. These APIs are represented in Figure 1b as the second layer from the top. Although each interface requires programming, all the low-level details can be ignored. In most cases, all one must do is make the correct function call at the correct time, and the interface will take care of the rest. Most HDF API routines are available in both Fortran-77 and C.

The routines that make up the low-level interface and the APIs are available in the NCSA HDF libraries. Source code for the HDF libraries, as well as binaries for some platforms, is in the public domain and is on the NCSA ftp server at `hdf.ncsa.uiuc.edu`.

On the highest "general applications" level, HDF includes ***command-line utilities*** for managing and viewing HDF files, ***NCSA applications*** that support data visualization and analysis, and a variety of ***third-party applications***. HDF utilities are included in the NCSA HDF distribution. Applications supported by NCSA, as well as applications contributed by members of the worldwide HDF user community are freely available on the NCSA ftp server.

1.3 Why Was HDF Created?

Scientists commonly generate and process data files on several different machines, use various software packages to process files and share data files with others who use different machines and software. Also, they may include different kinds of information within one particular file, or within a group of files, and the mixture of these different kinds of information may vary from one file to another. Files may be conceptually related but physically separated. For example, some data may be dispersed among different files and some in program code. It is also possible that data may be related only in the scientist's conception of the data; no physical relationship may exist.

HDF addresses these problems by providing a general-purpose file structure that does the following:

- Provides the mechanism for programs to obtain information about the data in a file from the file itself, rather than from another source.
- Lets you store mixtures of different data types and related information in more than one file, even when the files are processed by the same application program.
- Standardizes the formats and descriptions of many types of commonly-used datasets, such as raster images and multidimensional arrays.
- Encourages the use of a common data format by all machines and programs that produce files containing a specific dataset.
- Can be adapted to accommodate virtually any kind of data by defining new tags or new combinations of tags.

HDF files are ***self-describing***. For each data object in an HDF file, there is information about the type of data, the amount of data, its dimensions, and its location in the file. "Self-description" means that many types of data can be included within an HDF file. For example, it is possible to have symbolic, numerical and graphical data within one HDF file.

1.4 High-Level HDF APIs

The HDF I/O library consists of C and Fortran-77 routines for accessing objects and associated information. Although there is some overlap among object types, in most cases an API operates on data of only one type. Therefore, you need only familiarize yourself with the APIs specific to your needs in order to access data in an HDF file. The names of the HDF APIs designed to access the seven object types are:

- **DFR8 API** — Stores, manages and retrieves 8-bit raster images, their dimensions and palettes in one file. It is described in Chapter 6, titled *8-bit Raster Images (DFR8 API).*
- **DFP API** — Stores and retrieves 8-bit palettes in one file. It is described in Chapter 9, titled *Palettes (DFP API).*
- **DF24 API** — Stores, manages and retrieves 24-bit images and their dimensions in one file. It is described in Chapter 7, titled *24-bit Raster Images (DF24 API).*
- **SD API** — Stores, manages and retrieves multi-dimensional arrays of integer or floating-point data, along with their dimensions and attributes, in more than one file. It is described in Chapter 3, titled *Scientific Data Sets (SD API).*
- **DFSD API** — Stores, manages and retrieves multi-dimensional arrays of integer or floating-point data, along with their dimensions and

attributes, in one file. It is described in Chapter 11, titled *Scientific Data Sets (DFSD API)*.

- **AN API** — Stores, manages and retrieves text strings used to describe a file or any of the data elements it contains. It is described in Chapter 10, titled *Annotations (AN API)*.
- **VS API** — Stores, manages and retrieves multivariate data stored as records in a table. It is described in Chapter 4, titled *Vdatas (VS API)*.
- **V API** — Creates groups of any primary HDF object type. It is described in Chapter 5, titled *Vgroups (V API)*.
- **GR API** — Stores, manages and retrieves raster images of several bit lengths, their dimensions and palettes in more than one file. It can also manipulate unattached palettes in more than one file. It is described in Chapter 8, titled *General Raster Images (GR API)*.

As these interfaces are the tools used to read and write HDF files they are the primary focus of this manual.

1.4.1 User's Guide Code Examples

Copies of the C and Fortran-77 User's Guide examples, in ASCII format, are located on the HDF ftp server at `hdf.ncsa.uiuc.edu` in the "/pub/HDF/examples/Users_Guide" directory.

1.5 NCSA HDF Command-Line Utilities and Visualization Tools

HDF application software fall within the following three categories:

1. The Fortran-77 and C APIs described above.
2. Scientific visualization and analysis tools that read and write HDF files.
3. Command-line utilities that operate directly on HDF files.

Scientific visualization and analysis software that can read and write HDF files is available. This software includes NCSA tools such as Collage, user-developed software, and commercial packages. The use of HDF files guarantees the interoperability of the such tools. Some tools operate on raster images, others on color palettes. Some use images, others color palettes, still others data and annotations, and so forth. HDF provides the range of data types that these tools need, in a format that allows different tools with different data requirements to operate on the same files without confusion.

The HDF ***command-line utilities*** are application programs that can be executed by entering them at the command prompt, like UNIX commands. They make it possible for you to perform common operations on HDF files for which you would normally have to write your own program. The HDF utilities are described in detail in Chapter 13, titled *HDF Command-Line Utilities*.

1.6 Current HDF Platforms

The HDF library and utilities are maintained on a number of different machines and operating systems. Refer to the 'INSTALL' file in the top-level directory of the HDF distribution for this list.