# HCR Verification Utility Specification

**September 25, 1996**

Prepared Under RFP# ECS-URD-RFP-009

(U. of Illinois Ref. No. 96-NASA SBC-F-0162)

Albert Cheng

Michael Folk

William Whitehouse

**National Center for Supercomputing Applications**

**University of Illinois at Urbana-Champaign**

# 1.0 Purpose of the Utility

As part of the requirements of the HCR project a means of verifying the contents of an HDF-EOS file with an HCR description file must be provided. This is to be accomplished by a standalone software utility that may be bundled with the HCR suite of utilities.

# 2.0 Development/Testing Tools Needed

One of the primary development goals of the HCR verification software utility is to, as much as possible, utilize existing parser and development tools, and in particular, the same parser and development tools used by the HCR conversion and input utilities.

The required development and testing tools will be:

1. The current release of the ODL parser.
2. Version 4.0 release 2 of HDF.
3. Version 2.7.2.1 of the Gnu C compiler.
4. Version 2.5 of the Sun Solaris OS.

# 3.0 Prototype Functionality

1. The HCR definition file and the HDF-EOS file will contain only one object.
2. The utility will perform the verification according to the description in Section 7.0, Items 1a - 3a of this document.
3. The utility will return a value of 0 when both of the above conditions are met, a value of 1 when they aren't and a value of 2 when a functional error occurs.

### 3.0.1 Command-Line Syntax

The command line syntax of the prototype will be:

**hcrhdfdiff <HCR input file name> <HDF-EOS input file name>**

# 4.0 Final Product Functionality

1. The HCR description file and the HDF-EOS file will contain at least one object.

2. The functionality described in Items 2 through 3 in Section 3.0 will included in the final product, but expanded to accommodate more than one object and corresponding object description in the input files.
3. The utility will provide the ability to search on a user-specified object name and object type in the two input files, according to the description of the process in Section 7.0, Item 1b of this document.

It should be noted that this utility will only verify HDF-EOS objects in the data file - all other contents of the data file will be ignored by the verifier.

### 4.0.1 Command-Line Syntax

The command line syntax of the final version will be:

**hcrhdfdiff [-t <target object definition name> <ntarget HDF-EOS object name>] <HCR input file name> <HDF-EOS input file name>**

## 5.0 Utility Inputs

The proposed required inputs of this utility will be:

1a. One file containing HCR statements.

2a. One file containing HDF-EOS objects.

In addition, there are certain criteria that must be met by the two input files. If these criteria aren't met an error condition will result and the verification process will not be completed:

1b All HCR object definitions must follow the semantic guidelines described in the "HCR Configuration Record Requirements" document. It must at least contain one "END" statement - which is referred to as a "trivial" HCR file.

2b The order of occurrence of object definitions in the HCR input file need not be reflected in the order of occurrence of the defined objects in the HDF-EOS input file.

3b. The HDF-EOS input file can be any HDF file. In the case of either a file containing no HDF-EOS objects, the verifier will only report a match if the HCR file is a "trivial" one.

## 6.0 Utility Outputs

The proposed required outputs of the utility will be:

1. Terminal output consisting of a description of how the object definitions in the HCR input file and the objects in the HDF-EOS input file differ, but only if they do differ.
2. A returned status code values of 0 when the two files match, 1 when they don't, and 2 on a functional error.
3. All system-related error information (for example, an error description string returned by perror() on a malloc() error) will be written to the standard error stream.

## 7.0 Description of the Verification Processes

There are two methods by which an HCR file and an HDF-EOS file will be compared. In the default case the verifier will compare all object definitions in the HCR input file with all HDF-EOS objects in the HDF-EOS input file. The following criteria will determine the condition where the two input files are declared as "matching" by the verifier:

1a. Each object definition in the HCR input file must fully describe (according to the semantic guidelines described in the "HCR Configuration Record Requirements" document) a corresponding object in the HDF-EOS input file, AND

2a. Each object in the HDF-EOS input file must be fully described by a corresponding object definition in the HCR input file, AND

3a. Each component of said objects in the HDF-EOS input file must be described by a corresponding object component description in the HCR input file.

If any of these criteria are not met, the two input files will be identified by the verifier as non-matching, a description of the specific instances where the above criteria are not met will be printed to the terminal and the verifier will exit with a return code of 1.

The second means of verification occurs when a target HCR definition and HDF-EOS object is specified by the user. In this case, the following criteria determines a "match" condition:

1b. Each target object component definition in the HCR file fully describes the components of the target object in the HDF-EOS file.

If this criteria isn't met, the two input files will be identified by the verifier as non-matching, a description of the specific instances where the above criteria are not met will be printed to the terminal and the verifier will exit with a return code of 1.

## 8.0 Utility Test Methodology

The verification utility will be tested by running it in a shell script with a set of pre-prepared HCR and HDF-EOS input files. The contents of these files will include specific inaccuracies in syntax or incompatibilities between the object definitions and the objects themselves that will result in specific test outcomes and errors. To automate the evaluation of the results of these test phases, the output of the verifier may be redirected to a file and the contents of that file compared with a previously generated file containing the expected verifier output information. Also, the code returned by the verification utility will be evaluated and the script will contain instances of invalid calls to the verifier to determine if the utility correctly enforces its command-line syntax. Finally, each test that involves processing more than one object definition and/or object in the input files will not be included in the prototype tests.

Specifically, these test phases will be conducted as follows:

1. One HCR/HDF-EOS input file pair will test the verification process described in Section 7.0, Items 1a-3a. Therefore, there will be HCR objects defined in the HCR input file that will not exist in the HDF-EOS input file and vice versa. The verifier should accurately identify each inconsistency between the HCR input file contents and the contents of the HDF-EOS input file in its redirected terminal output and exit with a return code of 1.
2. Two HCR file/HDF-EOS file pairs will test the verification process described in Section 7.0, Item 1b. The first pair will test for a positive outcome and will contain identical object component information in both the HCR object definition and the HDF-EOS object. In this case, the verifier should exit with a return code of 0. The second pair will test for a negative outcome and will contain non-identical object component information in the HCR object definition and the HDF-EOS object. In this case, the verifier should write as description of the dissimilarities and exit with a return code of 1.
3. Two HCR/HDF-EOS input file pairs will test the verifier's ability to enforce the utility input requirements described in Section 5.0, Item 1b. The first description file will contain no object definitions, the corresponding HDF-EOS input file will contain one HDF-EOS object, and it is expected that the verifier will write a message describing the error to the standard error stream and exit with a return code of 2. The second description file will contain only the "END" statement, the HDF-EOS input file will contain no HDF-EOS objects, and it is expected that the verifier will produce no terminal output and exit with a return code of 0.
4. One HCR/HDF-EOS input file pair will test the verifier's ability to enforce the utility input requirement described in Section 5.0, Item 3b. The HCR input file will contain one object definition, the HDF-EOS input file will contain no HDF-EOS objects, and it is expected that the verifier will write a message describing the error to the standard error stream and exit with a return code of 2.
5. Three HCR/HDF-EOS input file pairs will test the verifier's ability to enforce some of the requirements specified in Section 5.0, Items 1b and 2b. These items refers to the specifications described in the "HCR Configuration Record Definition" document. These test phases will evaluate three of the requirements defined in this document:

a. Object namespace within the HCR input file. Two object definitions in this file will share the same name. It is expected that the verifier will exit on an error condition.

b. Correct inclusion of object components within an object description. In the prototype test, an HCR object definition with no object component descriptions will comprise the contents of the input HCR definition file. In the final version test, HCR definitions of each of the three HDF-EOS object types (i.e., swath, grid and point) will have necessary object component definitions missing in the HCR input file - for example, a swath datafield definitions will not be included in a swath object definition. It is expected that, in each case, the verifier will exit on an error condition.

c. Position-independence of object component definitions within an object definition. The HCR input files for this test will contain one object definition of the same type in two separate HCR input files. Each object definition will contain a list of object component definitions that occur in a different order from the other object definitions - for example, one swath definition will list its swath geofield component definition before the swath datafield definition, another will list the datafield component definition before the geofield definition. These HCR input files will be compared with a corresponding object of the same type in the HDF-EOS file. It is expected that the verifier will recognize the position independence of each of object component definitions and produce the same result for each HCR input file; specifically, no terminal output and a return code of 0.

6. One HCR/HDF-EOS file pair will test the verifier's ability to test for invalid object component definitions. The HCR input file will contain a grid object definition with an invalid data type value in the grid datafield definition, and an invalid projection type in a grid parameter definition. The verifier should recognize this as a input-file format error, write the appropriate error message to the standard error stream and end execution, returning the error code of 2.

7. Three HCR/HDF-EOS file pairs will be provided to test the verifier's ability to verify the three currently-supported HDF-EOS object types. One pair will have a swath definition in the HCR input file and a swath object in the HDF-EOS input file, the second will have a grid definition and a grid object in the appropriate files, etc.. It is expected that the verifier will process these objects correctly and produce no terminal output and a return code of 0.

8. In the calling script, the verification utility will be called with no command-line arguments. The verifier should recognize this as a command-line syntax error, write the appropriate error message to the standard error stream and end execution, returning the error code of 2.

9. In the calling script, the verification utility will be called with one command-line argument. The verifier should recognize this as a command-line syntax error, write the appropriate error message to the standard error stream and end execution, returning the error code of 2.

10. In the calling script, the verification utility will be called with the name of the HDF-EOS data file as the first argument and the HCR description file as the second. The verifier should recognize this as a file input error (violating the criteria listed in Section 5.0, Items 1b and 3b), write the appropriate error message to the standard error stream and end execution, returning the error code of 2.