# HDF Configuration Record Requirements

Technical Document

April 1996

Prepared Under RFP# ECS-URD-RFP-009

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

Albert Cheng

Michael Folk

Paul Harten

William Whitehouse

National Center for Supercomputing Applications

University of Illinois at Urbana-Champaign

# 1. Introduction

## *1.1 Purpose*

This document defines the requirements of the HDF Configuration Record (HCR). It describes only the components of objects that HCR is required to support. The syntax of the HCR script language is described in another document being drafted.

The rest of this chapter describes the HDF-EOS library and the needs of the Configuration Record. The next four chapters describe the support requirements of the HDF-EOS objects. The last chapter is a collection of related documents.

## *1.2 Introduction of the HDF-EOS Library*

HDF-EOS files consist of a collection of data objects. The various types of data objects (datatypes) supported are listed in the following table.

| HDF-EOS Datatypes | HDF Interfaces |
|---|---|
| ASCII Text | Annotation |
| Science Data Table | Vdata |
| Image | Raster |
| N-Dimensional Array | Scientific Dataset |
| Structure | Vgroup |
| Swath | |
| Grid | |
| Point | |
| P=V Metadata | |
| Data Dictionary | |

As the above shows, the HDF interfaces [HDF96] have already supported some object types but not the Swath, Grid, Point, Metadata and Data Dictionary objects. HDF-EOS library [EOS96-1] is created to provide interfaces to support those *HDF-EOS objects* in a fashion similar to the HDF interfaces.

## *1.3 The Need for a Configuration Record*

End-users may wish to create an HDF-EOS file using the HDF-EOS library interfaces according to the various rules governing different objects. In order to simplify this task, the HDF Configuration Record (HCR) is designed to provide a high-level description of the configuration of objects in an HDF-EOS file and the conceptual relationships between them. Additional software tools can then be used to automatically create a *structure* file based on the contents of the HCR.

In addition, the HCR should also support descriptions of the actual, low-level storage organization used by HDF to store HDF-EOS objects. These descriptions should be sufficient to enable users to write HDF code that will read the corresponding data in a meaningful way.

### 1.4 Objects Supported by the HCR

The HCR is required to support the configuration of the following HDF-EOS objects,

Swath
Grid
Point
Metadata

Appendix A shows a pictorial representation of objects supported by HCR.

## 2. The Swath Object

### 2.1 What is a Swath

The concept of a Swath is based on a typical satellite swath, where remote-sensing apparatus in a satellite performs a series of scans of the Earth's surface perpendicular to the ground track of the satellite as it travels in its orbit. (see the "Swath Concept" paper[EOS96-2] for detail.) The swath data the satellite transmits back to its data receiving stations contain two primary kinds of data, the geolocation data and the scanned data. HCR is required to support the definition of a Swath, the Swath's geolocation and scanned data, and the relation between the two constituent kinds of Swath data.

### 2.2 Defining Swath Dimensions

HCR must support the definition of Swath dimensions. A Swath dimension consists of two components - the dimension name and the size of the dimension. A Swath object's geolocation and scanned data are defined with respect to the Swath dimensions.

## *2.3 Defining Scanned Data*

HCR must support the definition of Scanned Data. The components of Scanned Data are Name, Datatype, Track-Dimension, Cross-Track-Dimension, and any other dimensions that comprise the Scanned Data.

## *2.4 Defining Geolocation Data*

HCR must support the definition of Geolocation Data. Geolocation Data consists of the components Name, Datatype, Track Dimension, Cross-Track Dimension, and any other dimensions that comprise the Geolocation Data.

## *2.5 Defining a Mapping Between Scanned Data and Geolocation Data*

HCR must support mappings between the Scanned Data and the Geolocation Data. This is done by defining relationships between dimensions of the Geolocation Data and dimensions of the Scanned Data. If a Scanned Data shares a dimension with a Geolocation Data, a direct mapping of that dimension between the Scanned and Geolocation Data is implied.

Dimension mappings can be categorized as either *regular* or *index*.

### 2.5.1 Regular Dimension Mapping

Regular Dimension MappingA regular dimension mapping consists of a Name, Scanned Data Dimension, Geolocation Data Dimension, Offset and Increment.

### 2.5.2 Index Dimension Mapping

An index dimension mapping consists of a Name, Scanned Data Dimension, Geolocation Data Dimension, and Index Array.

## *2.6 Defining Storage Schemes*

Sometimes it is more efficient to store several data arrays as one larger composite array. This allows for faster I/O on certain access patterns and reduces the number of physical objects in the HDF file.

HCR must support the specification of combining multiple data arrays into a single array provided the data arrays share the same dimensions and are of the same datatype.

# 3. The Grid Object

## *3.1 What is a Grid*

In HDF-EOS, a *Grid* object is one where the data is stored in a 2D or 3D array and this data is defined by a projection method supported by the HDF-EOS library.

The library supports those projections for which a known mathematical procedure exists. The procedure gives the geolocation (or latitude and longitude) for any data array location, and conversely gives the data array location for a given geolocation. One difference between a Grid object and a Swath is that the geolocation of Swath data values are not in any known projection while a Grid object may contain multiple data arrays but they all must share one and only one projection. The HDF-EOS library uses the General Cartographic Transformation Package (GCTP) for projections calculation. (See the "Grid Concept" paper [EOS96-3] for details.)

The HCR must support the definitions of the data storage structures for Grid objects and the projection methods applicable to Grid data.

## *3.2 Defining the Grid Parameters*

The GCTP software [GCTP96] specifies several parameters within Grid data that provide the geo-reference of the Grid. HCR must support the specification of the parameters as follows.

| **Parameter** | **Data Type** |
|---|---|
| Upper-left-point Metric | FLOAT64 |
| Lower-right-point Metric | FLOAT64 |
| X-dimension Size | INT32 |
| Y-dimension Size | INT32 |
| Pixel-Registration | TBD[1] |

[1] Details of Pixel-Registration to be determined.

### 3.3 Defining the Projection

Each projection contains a *name* and a list of *parameters* as specified in GCTP. Projections supported by the current release of the HDF-EOS library are:

Space Oblique Mercator,
Polar Stereographic,
Universal Transverse Mercator (or UTM),
Interrupted Goodes Homolosine, and
Geographic.

The GCTP software specifies projection parameters as a list of 15 FLOAT64 numbers with two exceptions. Projection **UTM** uses *ZoneCode* as its only parameter and projection **Geographic** does not use the *parameters* or the *ZoneCode* at all. All projections may have an optional parameter called *Spherecode*.

HCR must support the specification of the listed projections and their parameters.

### 3.4 Defining Grid Dimensions

HCR must support the definition of Grid dimensions. As with a Swath dimension, a Grid dimension consists of two components - the *dimension name* and the *dimension size*. Grid data is defined with respect to the Grid dimensions.

### 3.5 Defining Data Array

HCR must support the definition of Data Arrays. A Grid Data Array consists of three components - the *Array name*, *Array datatype* and a *list of dimensions*. This dimension list is comprised of the *Grid Y dimension*, the *Grid X dimension* and *the names of the other dimensions* of the Array.

### 3.6 Defining Storage Schemes

HCR must support the combination of several Grid data arrays into a single array, provided the data arrays are defined with respect to the same dimensions and are of the same datatype.

# 4. The Point Object

## 4.1 What is a Point

In HDF-EOS, a *point* object is one where data with associated geolocation that is not spatially or temporally ordered is stored in a science data table. Typically, a point object consists of a large table, with columns of data and other columns containing geolocation information (see the "Point Concepts" paper [EOS96-4] for details.)

A *point* object consists of one or more tables, also known as *levels*, and depending on how the *tables (levels)* are linked together, the *object* can be classified as *Simple Point*, *Linked Field Point*, or *Index Point*. HCR must support the definition of these three kinds of Point objects.

## 4.2 Defining a Table

HCR must support the definition of a table which consists of fields. A field consists of components of Name, Datatype and Order[2].

### 4.2.1 Reserved Field Names

HDF-EOS library version 1 reserves the following field names with predefined data types.

| **Keyword** | **Datatype** | **Comments** |
|---|---|---|
| Latitude | FLOAT64 | floating point latitude |
| Longitude | FLOAT64 | floating point longitude |
| CoLatitude | FLOAT64 | floating point colatitude |
| Time | FLOAT64 | TAI93 time in float |
| BEGIN | INT32 | used to point to tables |
| EXTENT | INT32 | ditto |

## 4.3 Defining Linking Relations

There are three kinds of linking relations corresponding to the three kinds of point objects.

### 4.3.1 Simple Point Object

A *Simple Point* object contains one or more tables with no linking between the tables. Each table should contain both *geolocation fields* and *data fields.* The absence of linking information implicitly makes the object a Simple Point.

[2] *Order* as defined in the HDF Vdata.

### 4.3.2 Linked Field Point Object

A *Linked Field Point* object contains two or more tables. Each table contains multiple fields and any two tables are linked to each other by the declaration of a common field name - referred to as a *LinkField*. One table is declared the Parent and the other the Child. The values of the LinkField in the Parent table must be unique and must contain all values that exist in the LinkField of the Child table.

HCR must support the definition of LinkField relationship between two tables. The relationship is composed of Parent Table Name, Child Table Name and the LinkField Name.

*Note that HCR does not verify the validity of the values of the Link Field. It only defines the relationship.*

### 4.3.3 Index Point Object

An *Index Point* object contains two or more tables. Each table contains several fields. These tables are conceptually related as Linked Field Point objects, but instead of being related by a LinkField, the Parent table stores row indices pointing at the Child table.

The Parent table contains, in addition to its other defined fields, two fields called *BEGIN* and *EXTENT* which are used to index into the Child table. The *BEGIN* field refers to the row number of the Child table, and the *EXTENT* field refers to the number of rows.
The
*BEGIN* field points to the first row number indexed in
the Child table and the *EXTENT* field points to the
total number of rows indexed.

HCR must support the definition of the Index relation between two tables. This relation is comprised of the Parent Table Name, Child Table Name and the special Link Field name POINTER.

# 5. Metadata/Data Dictionary

## *5.1 Types of Metadata*

Three types of metadata exist in an HDF-EOS file to provide general services such as listing the components, subsetting and subsampling the components by parameters and geolocations, and so on (see "Metadata" paper [EOS96-5] for details.) The three types are:

- **Core Metadata**
  Information about data product. Fields are common across data products.

- **Product-Specific Metadata**
  Information about data product. Fields are specific to that data product.

- **Structural Metadata**
  Information about the structure and contents of a particular granule.

### *5.2 Forms of Metadata*

All metadata are specified in the form of Object Description Language (ODL) [PDS95-12]. All metadata must be stored as part of the HDF-EOS file in text format.

### *5.3 HCR Support for Metadata*

HCR must support the specification of the *Structural Metadata* of the *Swath, Grid,* and *Point* objects in the form of ODL and store the configuration input as a text block in the HDF-EOS file as an HDF attribute.

### *5.4 HCR Support for the Data Dictionary*

This version of HCR does not currently support the Data Dictionary.

## 6. Related Documents

### *6.1 HDF-EOS Documents*

For a detailed description of the HDF-EOS library, refer to the following documents.

[EOS96-1]
"Draft Design Document for Proposed HDF-EOS Library",
http://edhs1.gsfc.nasa.gov/ftp/hdf_eos/doc/HDFEOSLib/

[EOS96-2]

"The HDF-EOS Swath Concept",
http://edhs1.gsfc.nasa.gov/ftp/hdf_eos/doc/SwathPaper/

[EOS96-3]
"The HDF-EOS Grid Concept",
http://edhs1.gsfc.nasa.gov/ftp/hdf_eos/doc/GridPaper/

[EOS96-4]
"The HDF-EOS Point Concept",
http://edhs1.gsfc.nasa.gov/ftp/hdf_eos/doc/PointPaper/

[EOS96-5]
"Thoughts on HDF-EOS Metadata",
http://edhs1.gsfc.nasa.gov/ftp/hdf_eos/doc/MetaThought/

### *6.2 HDF Documents*

For a detailed description of the Hierarchical Data Format (HDF) software, refer to the following documents.

[HDF]
"HDF Information Server",
http://hdf.ncsa.uiuc.edu/

[HDF96]
"HDF Users Guide", version 4.0,
ftp://hdf.ncsa.uiuc.edu/pub/dist/HDF/Documentation/HDF4.0/Users_Guide

### *6.3 GCTP Document*

For a detailed description of the GCTP software, refer to the following document.

[GCTP96]
"General Cartographic Transformation Package",
ftp://edcftp.cr.usgs.gov/pub/software/gctpc/getpc.tar.Z

### *6.4 Object Description Language Documents*

The Metadata are stored in the form of Object Description Language (ODL) as defined in the Planetary Data System (PDS) of the Jet Propulsion Laboratory. The following are related ODL and PDS documents.

[PDS95]
"Planetary Data System Standards Reference", Version 3.2, http://stardust.jpl.nasa.gov/stdref/stdref.htm

[PDS95-12]
“Object Description Language (ODL) Specification and Usage”, http://stardust.jpl.nasa.gov/stdref/chap12.htm

### *6.5 Parameter Value Language Documents*

ODL is related to the Parameter Value Language (PVL) as defined in the Standard Formatted Data Units (SFDU). The following are related PVL and SFDU documents.

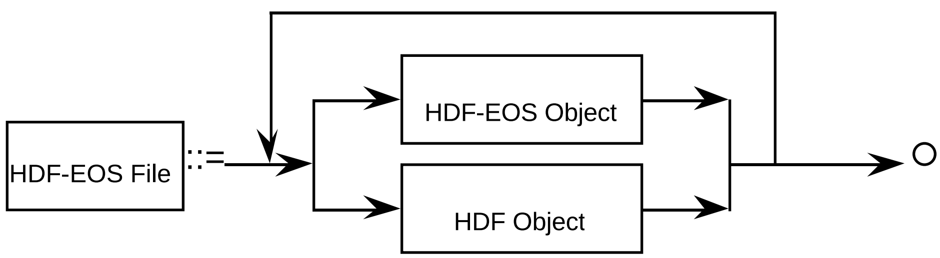
[SFDU92]
"Recommendation for Space Data System Standards, Standard Formatted Data Units -- Structure and Construction Rules", ftp://nssdc.gsfc.nasa.gov/pub/sfdu/p2docs/postscript/ccsds-641-0-b-1.ps

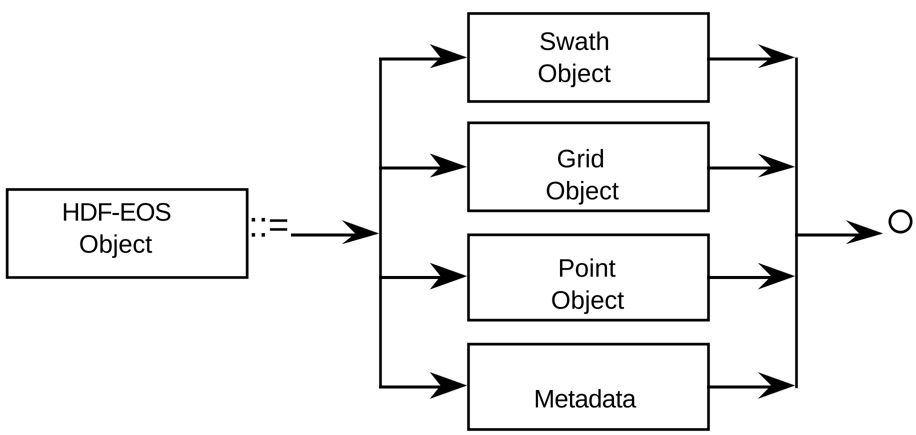
[PVL92]
"Recommendation for Space Data System Standards, Parameter Value Language Specification", ftp://nssdc.gsfc.nasa.gov/pub/sfdu/p2docs/postscript/ccsds-641-0-b-1.ps

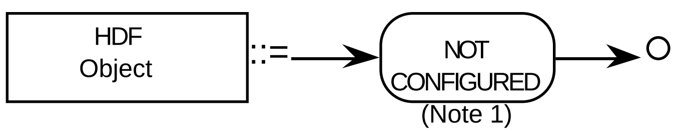
# Appendix A:

# HDF-EOS Objects Pictorial Representation

# HDF-EOS Objects

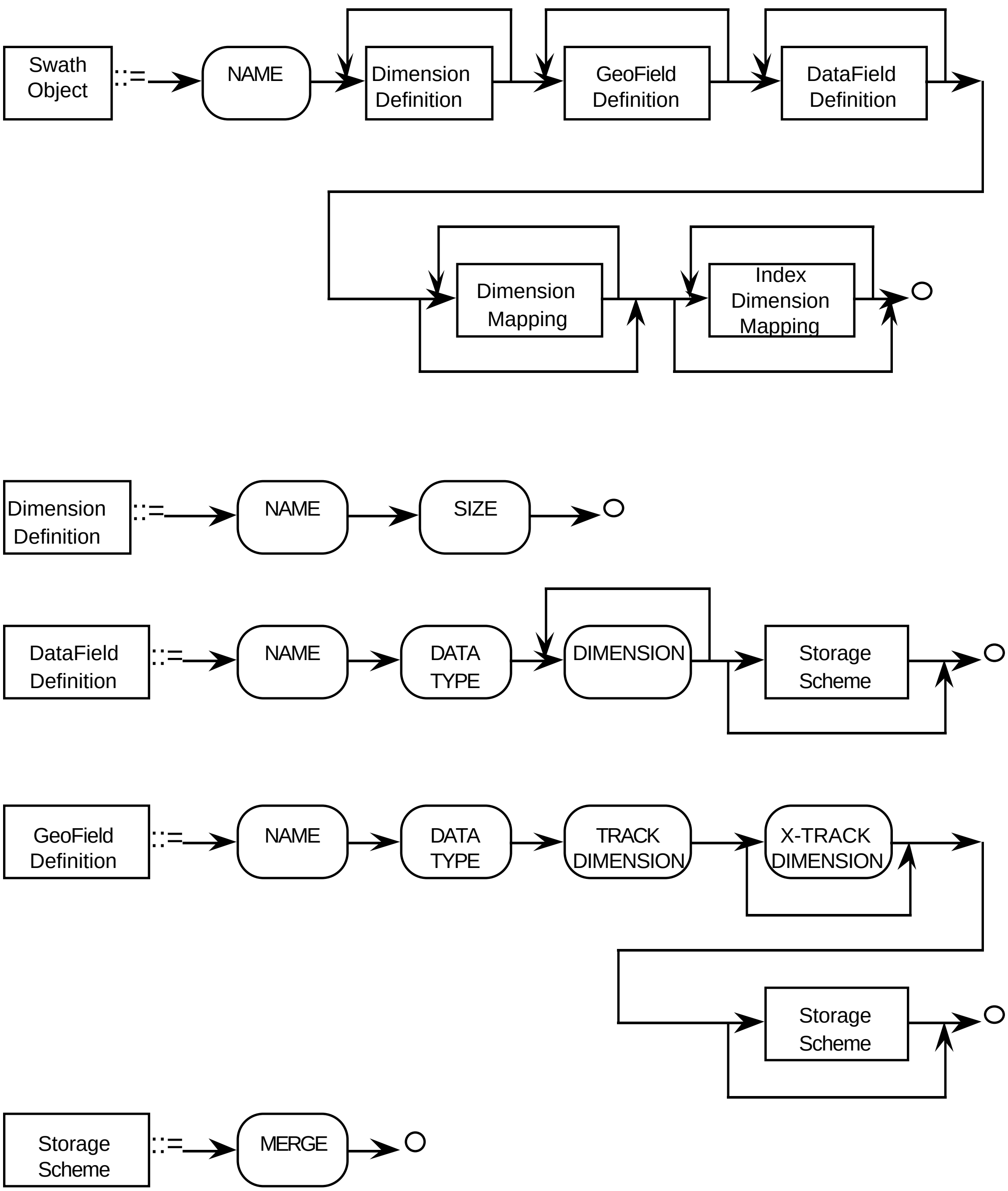


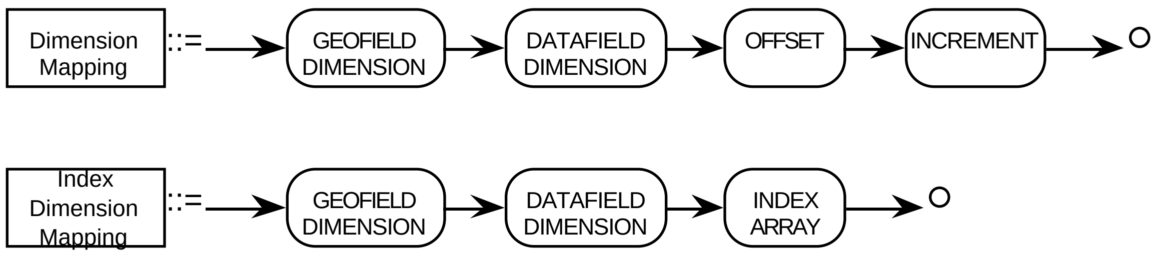




Note 1: Details of HDF Objects
to be determined later.

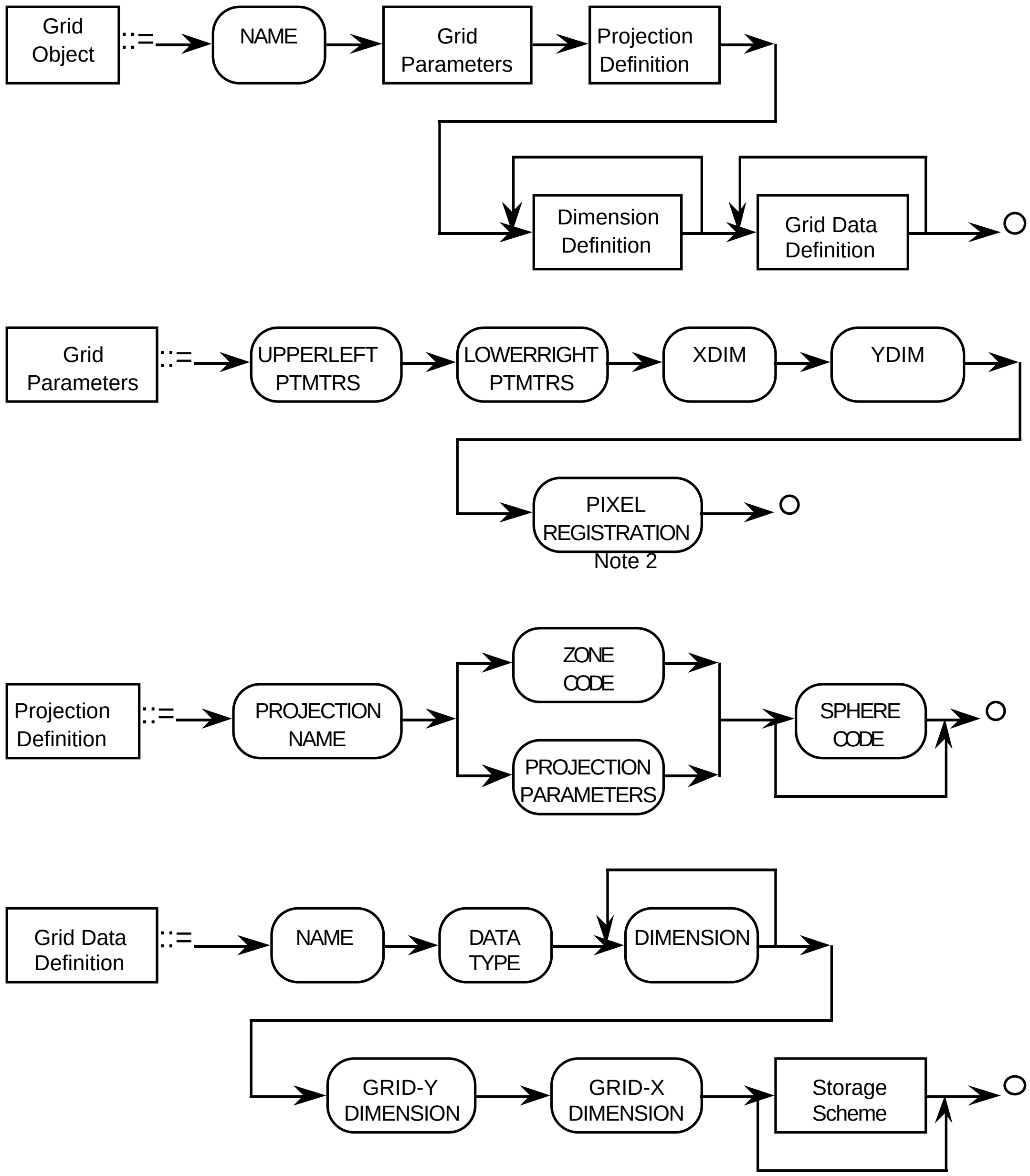
# Swath Object



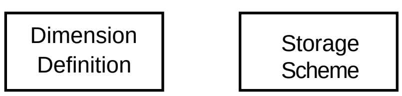
# Swath Object (Cont.)



# Grid Object




See Swath Structure for:




Note 2: Details about Pixel Registration to be determined later.

# Point Object