RFC: 0006

Title

DataMap Self-Describing Format

Author

R.J.Barnes

Summary

A Description of the DataMap self-describing format

Description

1. Introduction

The DataMap format was developed as a self-describing format to replace the existing SuperDARN binary formats. Although many self-describing formats already existed, none was suitable for use in the operational environment of the radar sites. The format was designed to be the simplest possible implementation of a self-describing format that placed the minimum number of restrictions on the user and developer. Although DataMap was originally developed as a simple file format, the same encoding method can also be applied to any stream of data. In fact, the DataMap format is currently used to encode the real-time data stream from the radar sites.

2. The Format

DataMap files or streams are comprised of blocks or records. Each block represents a single packet of information. A block is comprised of two types of variables that store the data, scalars and arrays. A scalar variable contains a single discrete value while an array variable contains multiple values with multiple dimensions. There is no restriction on the contents of a block, and different blocks can contain different scalars and arrays, although in most cases the same scalars and arrays will appear in each block.

2.1 Block Format

A block is comprised of a header followed by the scalar and array variables. The block header consists of an encoding identifier and the total block size. The encoding identifier is a unique 32-bit integer that indicates how the block was constructed. This value is used to differentiate between possible future changes to the DataMap format, currently only one encoding exists. The second part of the header is the block size, also stored as a 32-bit integer; this size represents the total size of the block including both the header and the subsequent data elements.

Byte Offset	Size (Bytes)	Type	Content
0	4	32-bit integer	Encoding identifier
4	4	32-bit integer	Block size
8	4	32-bit integer	Number of scalar variables
12	x	multiple	Scalar data
12+x	4	32-bit integer	Number of array variables
16+x	y	multiple	Array data

2.2 Names

Each variable has an associated name that uniquely identifies it. The name can contain any mixture of characters and are case-sensitive.As array and scalar data are treated independently, a scalar variable can have the same name as an array, however this is discouraged to avoid confusion. A name can contain whitespace characters, but should not as they may cause problems when converting to and from other data formats.

2.3 Data Types

Each item of data has an associated type. The DataMap format currently defines the following types:

Name	Data Size (Bytes)	Content
DATACHAR	1	Single character
DATASHORT	2	16 bit integer
DATAINT	4	32 bit integer
DATAFLOAT	4	Single precision floating point number
DATADOUBLE	8	Double precision floating point number
DATASTRING	x	String of characters terminated with a zero byte

2.4 Scalar Data

Scalar data consists of single discrete values. To store a scalar value, the DataMap format only requires enough bytes to store the name, the type of data and the actual scalar value:

Byte Offset	Size (Bytes)	Type	Value
0	x	Zero terminated String	Name
x	4	32-bit integer	Type
x+4	y	multiple	Value

2.5 Array Data

An array is defined by the number of dimensions it posesses and the ranges of those dimensions. The simplest form of an array is a vector, which consists of a one-dimensional array:

$v=(10.0\ 5.0\ 3.0)$	A one-dimensional array
$v=\begin{pmatrix} 10.0 & 5.0 & 6.0 \\ 3.0 & 4.0 & 2.0 \end{pmatrix}$	A two-dimensional array

Every dimension in an array has a range or extent which represents that maximum value an index along that dimension can have. Array ranges start at zero and can never have a negative value. For the two examples above, the range for the one-dimensional array is 3 and for the two dimensional array the ranges are 3 and 2. Arrays are exactly analogous to arrays in a programming language

An array variable has a name, type, dimension, a set of ranges and the data values contained in the array:

Byte Offset	Size (Bytes)	Type	Value

0	x	Zero terminated String	Name
x	4	32-bit integer	Type
x+4	4	32-bit integer	Dimension N
x+8	4*N	N 32-bit integers	Dimension ranges
x+8+4*N	y	multiple	Array values

The array values are stored so that the first dimension is the fastest varying. For a simple two dimension array with the range of the first dimension being 3 and the second being 2 the array data is ordered as follows:

v(x,y)	Index into Array Values
v(0,0)	0
v(1,0)	1
v(2,0)	2
v(0,1)	3
v(1,1)	4
v(2,1)	5

References

None

History

2004/06/22 Initial Revision.