Data Model Evolution in the Data Access Protocol

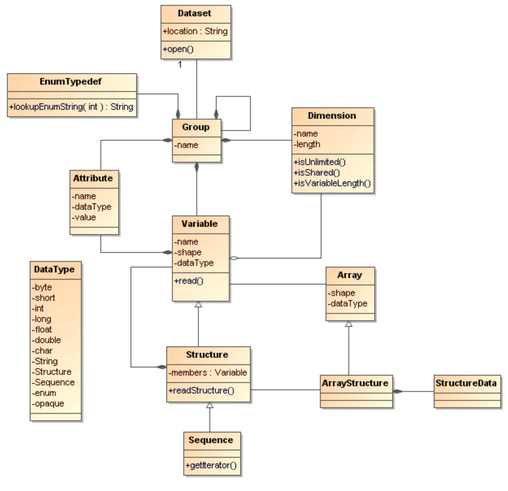
James Gallagher*, John Caron†, Dennis Hiembignert†, Hyo-Kyung Lee**, Rikki McQueary*, Russ Rew†, Muqun Yang**
*OPeNDAP; †Unidata; **The HDF Group

Two Projects to Extend DAP

Serving HDF5 files
• A 'handler' (software module) that enables the Hyrax Data Server to provide access to data stored in HDF5 files
• The initial version of the module encodes data using DAP version 2.0
• Some features of the HDF5 data model do not fit well in DAP 2.0 and thus they suggest ways to extend DAP

Enhance the netCDF library so that it can read from DAP servers in addition to local data files
• The initial version, based on software developed by OPeNDAP, assumes servers use DAP 2.0
• Some data types in DAP 2.0 are impossible to completely map into the netCDF API
• NetCDF 4 will be based on Unidata's Common Data Model (CDM) which can access all of the DAP 2.0 types

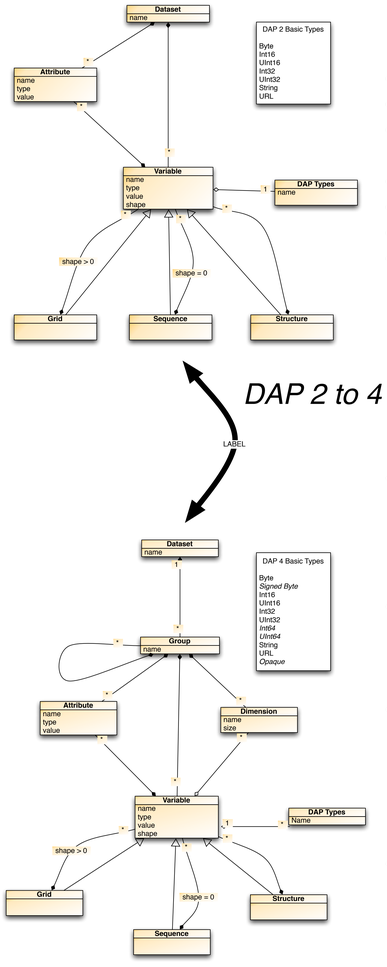
Common ground between projects
• HDF5 data model subsumes the CDM
• However, the CDM is less expressive only in that the Group data type is restricted to a hierarchy while in HDF5 it is a Graph (a rarely used feature). In all other aspects they are equivalent
• Thus, support for the CDM in DAP 4 will suffice to provide support for not only netCDF 4 - both the API and the file format - but also the vast majority of HDF5 files.



The Common Data Model Data Access Layer and DAP 4
Unidata has developed the Common Data Model (CDM) as a way of unifying the DAP and NetCDF data models (as well as the HDF5 data model). The above UML diagram is the CDM which is implemented by NetCDF4.

Guiding Principles

Three ideas have guided the changes to the DAP 2 data model:
• Optional features should be avoided because they tend to decrease interoperability and increase complexity
• Data model features should strive for high fidelity read access at the expense of features needed only for writing
• Attributes, not the data model, should be used to encode information for clients 'in the know.'



DAP 2 Basics

• A Dataset is a collection of variables
• A Variable has a Type, Name and Value(s)
• Variables may be a scalar or an array of any of the Basic Types, or a Structure, Sequence or Grid.
• Arrays of Structures are allowed but not arrays of Sequences or Grids
• Both a Dataset and a Variable may have Attributes
• Attributes are, like variables, Name–Type–Value tuples
• Attributes may be scalar or vector of any of the basic types as well as scalar structures

New Data Types in DAP 4

• **Groups**: Represented in DAP 2 using a naming convention, these are being added to DAP 4 because such conventions are prone to error and because the long names that result require special accommodation by many client applications.

• **Shared Dimensions**: Poorly represented in DAP 2 using extra variables, these will be added so this important semantic information can be made explicit within the data model.

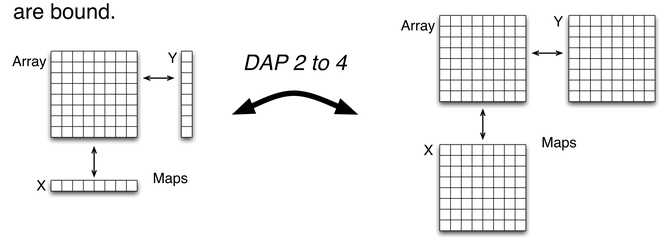
• **Opaque**: Represent a collection of data that must remain whole to have meaning and with a meaning that known to an external entity, not the DAP. These were added because the alternative, to use an array of Byte, etc., obscures the notion that these cannot be sensibly divided.

• **64-bit Integers, Signed Byte**: While including these here might seem incongruous, our experience has shown that not including types to represent integral types of various sizes makes writing servers and clients hard in practice because smaller types must be promoted to larger ones or values too large to represent must be elided. Either case impacts the fidelity of the information accessed.

DAP 4 Changes to Existing Types

• **Strings**: DAP 4 will use the UTF-8 encoding to support multilingual characters.

• **Grid**
  • Several Arrays can share a set of Maps
  • Not every dimension of an Array needs a MAP
  • Maps may be dimensioned up to the rank of the Array to which they are bound.



DAP 2 Grid Basics

A Grid is a simple relational type which binds one more Maps (coordinate variables in CDM's parlance) to an Array. For an N-dimensional array there must be N vector Maps.

A common use is for geospatial data where the vector maps hold latitude and longitude values. The location of a cell Array(x,y) in the array can be found by looking at values of X(i) and Y(j) which are typically vectors of floating point numbers.

DAP 4 Grid Extensions

The change allowing Maps to have up to N-dimensions provides a way to represent curvilinear coordinate systems.

Allowing multiple Arrays (not shown above) provides sharing of Maps (only simulated in DAP 2).

Dropping the requirement that all dimensions must have Maps provides support for model runs, et cetera.

CDM Features not in the DAP 4 Data Model

* **Enumerations**
* **Boolean**
* **Date/Time**
* **Type Definitions**

In all of these cases, these types can easily be encoded using integer (Enumeration, Boolean) or string (Date/Time, Type Definition) types with extra semantic information provided using DAP 4's Attributes. With the exception of Date/Time, these are types most useful in preventing errors during data set write operations.