Astronomical Data Analysis Software and Systems XIII
ASP Conference Series, Vol. 314, 2004
F. Ochsenbein, M. Allen, and D. Egret, eds.

Designing a Data Model for the Virtual Observatory

Mark Cresitello-Dittmar, Janet DePonte Evans, Ian Evans, Michael Harris, Stephen Lowe, Jonathan McDowell, Arnold Rots

Harvard-Smithsonian Center for Astrophysics

Abstract. The goal of the Virtual Observatory is to make astronomical data more accessible and to provide the means to more easily analyze that data. Currently, archives hold analogous data in a variety of different representations, which impedes interoperability at the data analysis stage.

An important element of the VO is a data model that can unambiguously represent the relationships between data values and physical properties. At the CfA we are developing a data model design that can support the representation, analysis and display of data collected on different types of instruments. This model is a common, high-level framework of general-purpose components for fusion of heterogeneous data sources. From this framework, we have focused on a subset of components required to meet selected science objectives on spectral and image data.

1. Dataset

Here we present elements of an observation data model for the VO. Figure 1 shows the *Dataset*, the major object for managing data either from empirical observations or from simulations. The shaded boxes indicate the focus of our current modeling efforts at CfA. Starting with section 2, this paper concentrates on the *Data Container* object which provides access to the data values. The remaining components are described briefly here.

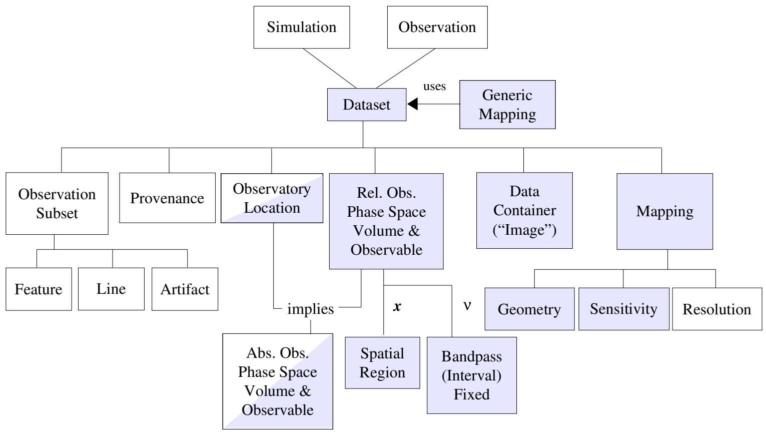


Figure 1. Dataset Model

The *Relative Observational Phase Space Volume & Observable* component specifies the region of physical space being observed (*Phase Space*, which may have dimensions of space, time, wavelength, etc.) and the quantity being measured (*Observable*) relative to the observatory location. These values can be translated to an *Absolute* reference by using data in the *Observatory Location*.

The *Mapping* component provides the translation from pixel elements to volumes in the phase space. It also specifies the relationship between the pixel values and physical values.

The *Generic Mapping* component provides a framework for organizing standard data transformations. It can be thought of as a library of transformations that may be used to define the specific mappings needed in a dataset. This library includes the usual astronomical spherical projections as well as mappings between units, between coordinate systems and between data values that are denoted using interchangeable properties such as frequency and wavelength.

2. Data Container

The *Data Container* (Fig. 2) addresses the conflicting requirements of permitting arbitrarily irregular instrument structures to be represented while maintaining efficiency for the many common datasets that are highly regular. It provides access to the measurement data and a logical view of its organization. (This may differ from the in-memory layout.) This logical organization is framed by the *Index Set*, which specifies the indexes or labels that identify the individual data cells. For the many data sets which are naturally laid out as a simple data (hyper) cube, the Index Set would be the usual n-tuples, e.g., $(1,1),\ldots,(m,n)$. The key to handling the conflicting needs is to provide multiple views, at least two access patterns for the data.

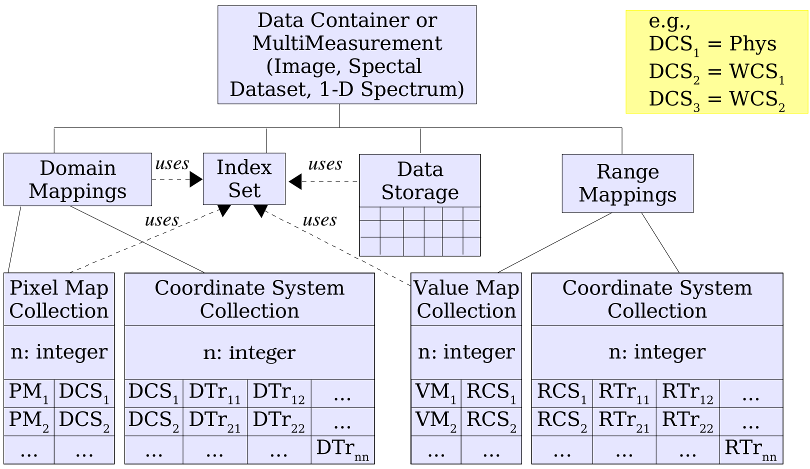


Figure 2. Data Container Model

To support generality, the *Data Container* methods always allow the list of indexes to be obtained and used to iterate through the data cells. The data value and/or metadata can be obtained for each cell, in essence using heavyweight objects for each data item. A data consumer (i.e., application software) can fall back on this form to process the data if it does not recognize the *Index Set*'s structure.

To support efficiency, the *Index Set* conforms to one of a small set of archetypal structures such as array, array with bad cell mask, sparse array, or event list. Application software can then be designed to take advantage of the structure to organize processing.

Metadata describing the correspondence between the data cells and locations in detector or observational space is represented as a collection of pixel mappings PM_1, PM_2, ... into coordinate spaces DCS_1, DCS_2, ... Similarly, interpretation of the data cell values is handled by a value mappings VM_1, VM_2, ... into coordinate spaces RCS_1, RCS_2, ... Depending on the need, the VMs may depend on the cell location as specified by its index.

These mappings are not simply computable functions, but also have the type and parameters of the transformation encoded, such as constant, linear, piecewise linear or tangent projection. Thus, the application program can inspect this information to best organize its processing.

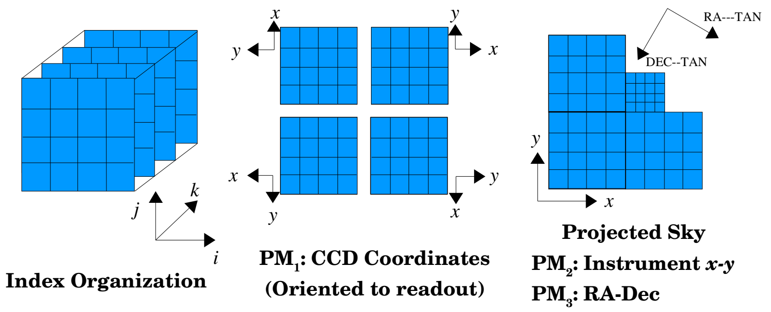
3. Three Ways to Data

In our model, there are three ways of accessing the data:

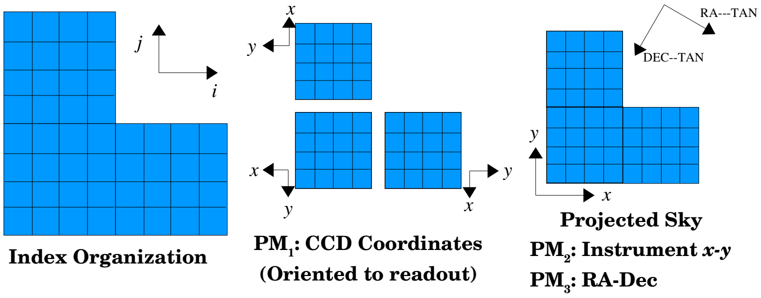
- As a list of pixels with no assumptions about contiguity of pixels in physical space or in memory.
- By the logical structure which is defined by the Index Set, such as an n-dimensional array, which might not be fully rectangular due to missing or invalid cells. The data provider determines this structure.
- As chunks of pixels which are rectangular, regular, filled arrays addressable by pixel offsets into contiguous memory. This supports highly efficient access. A simple FITS image would require only a single chunk, mosaics a few chunks, and sparse arrays many chunks.

4. Example: Hubble WFPC2

In the diagram below we show how these elements might be used to represent the data from the Hubble Wide Field and Planetary Camera. The data from the four CCDs can be organized as an 800x800x4 block. Mappings $\mathrm{PM}_{1,2,3}$ describe the layout of the detector panels and the sky layout in two coordinate systems.

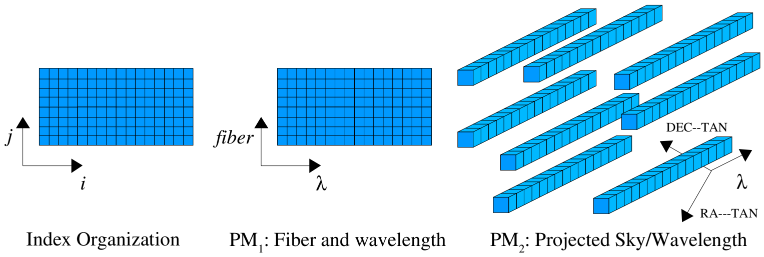


The *Index Set* is not constrained to be rectangular. Using this feature, another *Data Container* can be defined describing just the Wide Field Camera, as shown in the accompanying figure below. This object uses a different *Index Set* and correspondingly different mappings to access the same data. The data provider (i.e., archive) defines the *Data Container(s)* and *Index Set(s).* This gives the provider the flexibility to create an organization natural for its data, while at the same time define alternate views for different audiences or purposes.



5. Example: Fiberoptic Spectrometer

In a fiberoptic spectrometer, 1-D spectra are measured at a number of irregularly-arranged sky positions. As seen in the next figure, the data may be stored as a 2-D array, each row holding the spectrum for a single position. Consequently, each array element maps to a location in the 3-D domain *sky* $\times$ *wavelength.*



6. Continuing Effort

Our next steps in moving the data model development forward are:

- Complete definition of components for 1-D spectra and for images.
- Define XML format for data model components.
- Develop software to render data from several archives into XML.

We are in the process of developing a prototype system. In addition to data model components, the system includes a network interface module that manages the communication details between clients and SIAP services.

Acknowledgments. This material is based on work supported by the National Science Foundation under Grant No. AST-0121296 and under Cooperative Agreement No. AST-0122449. This effort is also supported by the Chandra X-ray Center under NASA contract NAS8-39073.