# Providing Support for n-Cubes in the Virtual Observatory

*A Vision Document*

Arnold Rots
Omar Laurino
Mark Cresitello-Dittmar
Janet DePonte-Evans
Giuseppina Fabbiano

*Smithsonian Astrophysical Observatory*

## 1. Usage Context

The access and analysis of data cubes can be divided into several facets, which encompass both the data products and the analyses that can be applied to such products.

### 1.1. Data Product Types

**Domain specific data products:** This category includes waveband, instrument, and/or configuration-specific data products, including calibration files, visibility data, un-calibrated event lists, and so forth. Metadata are usually observatory-specific and not standard. These products need specific expertise in order to be handled in a scientifically meaningful way.

**Standard n-cubes:** Domain-specific raw data products may be processed by pipelines (or reprocessed by users) to create standard n-dimensional hypercubes (hereafter, "n-cubes"). The elemental component of a standard n-cube is a voxel, i.e., a discrete element in the n-dimensional space determined by the "n" quantities measured by the instrument. The voxel's value occupies a hypervolume defined by the pixel size along the different axes, and may be characterized by resolutions and statistical errors or upper/lower limits. In some cases, multiple physical axes may be collapsed onto a single, degenerate n-cube axis (e.g., the spatial/spectral data axis from a slitless spectrograph or single-pixel axes). Standardized metadata express the position of the reference voxel, the projection of the voxels, the units, and any other information useful to make scientific sense of the data. The metadata also may reference the raw data products and the processing history of the n-cube. Using and analyzing these standard n-cubes usually does not require any domain-specific expertise. However, specific expertise is definitely required in order for the user to be able to regenerate standard n-cubes from the original raw data files.

**Abstract data model:** While the raw data and metadata are likely to follow customized data models, an abstract representation of standard n-cubes must be possible in order to allow interoperability. The mapping between the abstract data model and the actual n-cube realization will require low-level I/O software for interpreting the data file, and a higher-level layer on top of that for mapping the file contents to the abstract data model. In many cases, the mapping can be described as a simple transformation. More complex cases might require software plug-ins to implement the abstract standardized interfaces.

## 1.2. Operations, Processing, and Analysis

**Common operations on n-cubes:** Some operations can be performed on any standard n-cube, using generic standard tools. This list is somewhat arbitrary, but in general these operations should not depend on any domain specific information. Such operations may include: projections over a set of axes, aggregation of voxels along one or more axes (e.g., average, median, sum), extracting sub-arrays, and interpolation. More sophisticated analysis algorithms include derivation of moments along specific axes, pattern recognition, feature extraction, and cuts along arbitrary surfaces. Some of the operations can lead to other publishable products: 1D signals (e.g., spectra, time series, SEDs), 2D (e.g., images, visibility maps, polarization maps), and possibly higher dimensions (e.g., 3D SEDs with spectral, flux, and time axes). Common operations can be applied to these derived products (e.g., cross- and auto- correlations, phase analysis for 1D signals; source detection and photometry for images; fitting/modeling in the n-D space). Standard metadata also allow one to consistently display n-cubes, their projections, and their derived products.

**Custom operations on n-cubes:** Some operations can be specific to a science domain or to a particular energy band. Experts must be able to perform these operations in the specific domain on the standard n-cubes themselves, without requiring any additional information. For example, if the n-cube contains data about an object of a specific class, then domain-specific analyses can be performed (e.g., photometric redshift estimation for AGN).

**Reprocessing of the raw data:** Users might be interested in reprocessing the raw data by applying domain-specific operations on them. Domain-specific tools could allow one to annotate the resulting products with standardized metadata and according to the standard data model. Access to, and analysis of, raw data can be performed using customized protocols, data models, and metadata descriptions, since specific understanding of these components is required. However, the standardization is in the creation of the derived products that can then be used by standard applications, along with data coming from standard services/applications.

**Republishing of derived data products:** Derived data products, generated by standardized applications, can be easily republished in a set of specific services, becoming part of a derived, annotated knowledge base.

**Data Flow:** A data product's life cycle will generally follow this path: ***Observation*** – *pipeline* → ***domain-specific products*** – *processing* → ***VO n-cube*** – *analysis* → ***derived products*** – *further interpretation* – *publish* → ***VO or elsewhere***.

## 1.3. Usage Requirements

**Putting all together — discovery, access, and usage of large data cubes:** Data discovery allows the user to find out what data are available in a specific region of the high-dimensional space that represents the entire observational parameter space. Parameters are not only the coordinate axes in the (space, time, spectral, redshift, polarization, etc.) domain, but also include additional technical parameters that may be relevant, such as the desired instrumental resolution and accuracy, calibration properties, and so forth.

So, a query to a discovery service needs to be encoded in a structure expressive enough to represent the complexity of the region in this parameter space, and must not assume that the ROI boundaries are parallel to the parameter space axes. The query must also allow the user to narrow the search down to the kind of services of interest (e.g., mosaic, cut-out, and so on).

Access to the chosen data products must be provided in an efficient way. Since we may be dealing with very large datasets, attention must be paid to ensure that the transfer protocol is not naive, but is fine-tuned to this task. A possible solution, currently not supported by the IVOA SIAP protocols, is described by Kitaeff et al. (http://arxiv.org/pdf/1209.1877v1.pdf) in their presentation of SkuareView, a framework for accessing extremely large radio astronomy image data.

Having a standard means to discover and access data products has very little value if the discovered data products require specific expertise in order to be meaningfully used — if an astronomer needs to study the documentation of each data access service to understand how to use the products, then (s)he might as well use the local custom access interface instead of a standardized one. So, it is important that the n-cubes are provided in a standard format.

The standardization of the file annotation process, or language, is more important than the standardization of a controlled vocabulary, so that domain specific information can be carried along in a standardized description language. Thus, the abstract data model should be expressive enough to allow a generic, but rich, set of common tools to perform common operations of the n-cubes in a standardized software framework. On the other hand, it must be possible to extend the abstract data model and describe the extension in a standardized way, so that domain-specific information can be conveyed by the data providers in an efficient and effective way. Such standardization also should allow for the addition (by scientists, developers, data providers, and collaborations) of domain-aware plug-ins to the generic standardized software framework to implement custom operations.

**Data formats:** While the abstract data model defines the generic representation of an n-cube, actual data might come in different formats. And while it should be generally possible to map the file contents to the abstract data model itself, it is useful to review the file formats that are usually used for storing data cubes. The most used formats, at the moment, seem to be:

- FITS
- HDF5
- CasaTables
- RDF/TSV

These formats have been designed following different approaches. In particular HDF5 and CasaTables have been adopted because of their ability to store large complex datasets. However, they are quite different; for example, HDF5 employs a hierarchical model, while CasaTables use a more flexible relational model. A thorough comparison between these two formats (although from CASA perspective) is provided in http://www.astron.nl/~gvd/tables-hdf5-comparison.pdf.

A less detailed critical comparison of the first three formats is provided by Kitaeff et al. (see http://arxiv.org/pdf/1209.1877v1.pdf).

# 2. A Way Forward

## 2.1. Data Access Layer

### 2.1.1. Summary

The SIAP2 Working Draft (WD) needs the addition of several extra parameters, including:

- Required axis order;
- Preference for cut-out service;
- Redshift coordinate;
- Sparse vs. filled vs. pixel list n-cube.

SIAP2 allows POS and SIZE to be replaced by REGION; we propose that this be extended to allowing all coordinate axes specifications to be replaced by an STC-S string.

### 2.1.2. Types of Metadata

The SIAP2 WD, in its current form, does not provide sufficient functionality and is not sufficiently extensible. A major concern is that the WD poses significant restrictions on what can and cannot be asked for, limiting its generality. However, the WD is a good start and we propose some modifications that address the shortcomings.

Functionally, there are three categories of metadata in the query and the response:

1. Information about the volume in coordinate space: reference systems, bounds, uncertainties, resolution, voxel size, etc.
2. Information that allows the client to use the data: axis ordering, reference voxels and values, etc.
3. Information about preferences: exact match, overlap, axis order, flux limits, exposure times, type of service (e.g., mosaic, cut-out), etc.

A *query* will contain metadata from categories 1 and 3.

A *response* also needs to include metadata from categories 1 and 3, while metadata from category 2 may be helpful.

A *returned data object* needs to include metadata from categories 1 and 2, while metadata from category 3 may be helpful.

#### *2.1.2.1. Volume in Coordinate Space (Category 1)*

SIAP2 in its current form lacks full generality in category 1. Though, that may not be a major issue, a more serious concern is that the WD does not provide obvious extensibility. There is one part of coordinate space where the WD does allow full generality: spatial coordinates, through the use of the STC-S REGION specification.

We welcome this option, as the STC standard is intended to play this exact role, but note that this only provides full generality for one of the ten coordinate-related parameters: POS, SIZE, REGION, SPATRES, BAND, SPECRES, SPECRP, TIME, TIMERES, and POL.

The logical consequence of this is to propose that the option be extended to allow *all* coordinate-related query parameters to be replaced by an STC-S expression (see Section 2.1.4).

#### *2.1.2.2. General Metadata (Category 3)*

SIAP2 does a good job of enumerating the metadata in category 3 — essentially all parameters other than the ones listed above. However, we note four items that are missing from the SIAP2 draft:

- The client needs to be able to specify its preference for mosaic or cutout services;
- The redshift/Doppler coordinate is missing from the category 1 metadata; it is available in STC-S and STC-X;
- A query parameter should be added that allows the client to specify what type of data object the user is interested in: filled array, sparse array (i.e., a collection of sub-arrays), or a pixel list (a sparse array where individual voxels are provided in a table; this includes event lists); this parameter should be in category 3;
- There is no way in the category 3 metadata to specify a desirable or required axis order; STC-X can handle this (though it is somewhat involved), but the capability is not included in STC-S.

### 2.1.3. SIAP2 Client Implementations

We propose to explicitly allow two classes of server implementation:

- SIAP2 server class A only accepts "traditional" SIAP (i.e., no STC-S);
- SIAP2 server class B accepts "traditional" SIAP as well as STC-S logic (this works because traditional SIAP can unambiguously be converted to STC-S).

Class B could be allowed to respond with an STC-S or STC-X description.

### 2.1.4. STC Data Model

Here we provide some background on the STC metadata standard. The primary intent of the development of STC as a metadata standard was to enable us to express the volume of coordinate space that is occupied by a data object (or that we are interested in the context of a query), with the most essential properties of the mapping along the coordinates.

For that reason, one of the top-level elements in STC is the query element whose function and intent is to specify, in the most general sense, the volume in coordinate space for which the user is requesting data.

The second objective of STC was to provide metadata that will project the coordinate space onto a pixel coordinate system.

STC exists in two serializations: STC-X (XML) and STC-S (a linear string).

STC-S can only provide category 1 metadata, while STC-X can also include category 2 metadata. In other words, STC-S and STC-X can both provide the category 1 metadata for queries, responses, and data objects, while STC-X can also provide the category 2 metadata (the mapping to pixel space) for the data objects and responses.

We fully realize that the following need to be considered in this:

- The category 2 metadata description, telling the user how to use the data, is not fully tested in STC-X, but is readily available through FITS WCS;
- The additional query parameters in category 3 that are not included in STC are (mostly) represented in SIAP2;
- There is no reason to require that the full power of STC be implemented (certainly not initially);
- We need a reasonable transition from existing access protocols to a full STC-based one.

**This has led us to the concrete proposal:**

*That SIAP2 be amended to allow the coordinate-related query parameters to be replaced by an STC-S expression.*

Specifically, the parameters that may be replaced by an STC-S expression are:

- POS
- SIZE
- REGION
- SPATRES
- BAND
- SPECRES
- SPECRP
- TIME
- TIMERES
- POL (once added into STC)

It means that users will have the choice between specifying their query as is currently defined in the WD and using an STC-S expression. Consequently, queries that would not be possible using the current query parameter structure can be implemented with the STC option. All non-coordinate parameters remain as they are. To some extent, this is not a radical departure: STC-S expressions are already allowed in the REGION parameter. So, in a sense, this is only a further extension of the current WD.

Obviously, these queries can also refer to event list data objects.

We provide one example, taken from Section 5.4.2.3 of the SIAP2 draft.

SIAP2:

POS=52,-27.8&SIZE=0.5&BAND=2.7E-7/0.13&TIME=1998-05-21/1999

STC-S (spaces are replaced by the "□" character):

STC-S=TimeInterval□UTC□1998-05-21□1999-01-01□Box□ICRS□52□
-27.8□0.5□0.5□SpectralInterval□2.7E-7□0.13□unit□m

## 2.2. Data Model

### 2.2.1. Abstract Data Model

The abstract data model provides a unified description of standard n-cubes, sparse n-cubes, and pixel/voxel/event lists.

The root **abstract data model** is a collection of sub-arrays that all fit into a single super-array. The metadata (including WCS) for the super-array need to be provided. WCS metadata for each sub-array need to be provided with that sub-array. The sub-arrays in the collection need not be the same size.

With this model, a standard single n-cube is a special case — the collection of sub-arrays may consist of just one sub-array that is congruent with the super-array.

Similarly, a pixel/voxel/event list is also a special case, where each of the sub-arrays comprises a single element (scalar).

### 2.2.2. Data Model Realization

Here we give one example here of how such an abstract data model *could* be implemented in a specific serialization represented by FITS.

Root, sparse n-cube:

- A binary table where each sub-array occupies one row;
- WCS and voxel coordinate information for super-array are recorded in the header;
- WCS and voxel coordinate information for sub-arrays are recorded in discrete columns;
- One column with the actual sub-array data recorded as a variable length array;
- Conceptually, multiple n-cubes that share the same voxel coordinates *could* be recorded as multiple sub-array data columns — this would improve efficiency in the case of many pixel/voxel/event lists.

(Note: this is similar to the HEASARC response matrix format.)

Single n-cube:

- Derived from the root n-cube model;
- Consists of just one row since there is only one array (the super-array);
- Since all of the metadata are recorded in the header already, the super-array *may* also be represented in an image extension or even in the primary array.

Pixel/voxel/event list:

- Derived from the root n-cube model;
- This is a degenerate case — all sub-arrays are single elements.

### 2.2.3. Implementation

A particular format is prescribed with specific requirements regarding structure and metadata. Presumably, there will be data model implementations defined for FITS, HDF5, CasaTables, RDF/TSV, etc.

Plug-in converters can be written to convert existing data formats to the standard n-cube data format, eliminating the need to convert existing archival data products.