Astronomical Data Analysis Software and Systems XII
ASP Conference Series, Vol. 295, 2003
H. E. Payne, R. I. Jedrzejewski, and R. N. Hook, eds.

HDX Data Model: FITS, NDF and XML Implementation

David Giaretta, Mark Taylor, Peter Draper, Norman Gray, Brian McIlwrath
Starlink Project, UK

Abstract. A highly adaptable data model, HDX, based on the concepts embodied in FITS and various proposed XML-based formats, as well as Starlink's NDF and HDS will be described, together with the Java software that has been developed to support it. The aim is to provide a flexible model which is compatible with FITS, can be extended to accommodate VO requirements, but which maintains enough mandatory structure to make application-level interoperability relatively easy.

1. Introduction

HDX is a flexible and extensible data model for astronomical and other data. The ideas underlying HDX have been tested in a large volume of deployed software. The resulting system is designed to be highly interoperable: it is platform independent, and neutral as regards file formats, though its 'natural' (in the sense of first implemented) formats are FITS and XML. This paper is a progress report on work we have been carrying out on a Structured approach to data; updates will be available on `http://www.starlink.ac.uk/hdx`.

2. Motivation

Increasingly complex data structures are becoming necessary as more complex instrument data becomes available. Some indication of this comes from proposals for additions to the FITS format such as the Hierarchical Grouping Convention[1] proposal. On the other hand the basic FITS format does not readily lend itself to such extensions. In addition there is a growing recognition that astronomical applications must deal with data quality as well as track data errors as a matter of course.

The Virtual Observatory (VO) brings the promise of yet more complex, interrelated, distributed, data; the development of VOTable[2] shows a recognition of the importance of XML. Metadata is seen as a key component of the Virtual Observatory, as well as provenance of information gathered or created by a remote, automated process, using the VO. Complex interrelationships between

[1] `http://fits.gsfc.nasa.gov/group.html`

[2] `http://cdsweb.u-strasbg.fr/doc/VOTable/`

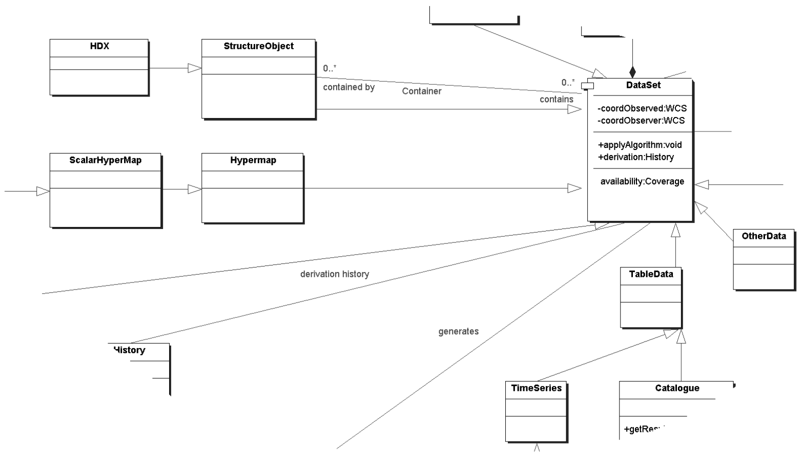

Figure 1. Section of the Data Model

large numbers of files are likely to become commonplace in the VO. Rather than develop an astronomy specific semantic toolkit it seems sensible to position oneself to be able to use the tools which are being produced in the context of the wider WWW community, such as RDF and related standards.

We see therefore requirements for something flexible, extensible, capable of storing hierarchical information, able to deal with distributed data but usable locally, and with the backing of a sophisticated astronomical data model. In addition it should be open to the new tools and standards which are bound to be produced in the near future outside astronomy. It should also facilitate interoperation of applications—something which will become increasingly difficult as complex structures are generated.

3. The Structure Object

Figure 1 shows an extract from a proposed hierarchy of data objects.

In addition to the data containers such as Table and N-Dimensional array, there is an additional Structure Object which can contain other objects, including other Structure Objects—allowing a hierarchical data structure to be developed.

It is important to remember that we are not talking about any particular data format such as FITS. Instead we are considering conceptual data structures which may be serialised in a number of different ways.

The advantages of separating the data container elements from the structuring elements include allowing the containers to avoid becoming more complicated than necessary. It leaves one free to consider additional metadata, which is recognised as being of fundamental importance for the VO, without being forced to think of encodings which would fit within the constraints of FITS keywords. Instead one is free to consider the use of something like XML, with its promise of a large number of standards and tools as a serialisation mechanism.

3.1. Dangers of Using of Hierarchical Data

Using hierarchical data does have dangers, at least until such time as structural metadata is adequately developed—which is probably some way off. The danger is that one application will not understand the relationship between components which another application has written out.

Starlink's experience with hierarchical data structures, based on the Hierarchical Data System[3] (HDS), over the past 10 years or more is of use here. This experience shows that one must strike a balance between being very proscriptive in what applications can write out, e.g., simple FITS files, on the one hand, and allowing anarchy on the other. The most common problem was for applications to not understand the relationships between components, leading to erroneous processing, or that pieces of metadata which were not understood were not correctly passed on to downstream applications.

Our experience is that one needs some fairly simple rules which applications must obey, and that there should be some pre-defined components within which to hide additional structures in order to allow common operations to be dealt with uniformly and correctly. An example of this is NDX (based on Starlink's NDF) which is described below. In addition it must be possible for an application to adequately check the validity of a hierarchical file with which it is presented. We refer the reader to the HDX documentation for a full discussion of these rules.

4. Candidate Structure Object: HDX

Starlink has been developing a candidate Structure Object called HDX. It embodies a flexible data model based on many years experience with HDS, which uses local data. One aim is to support distributed processing and data holding, using URI's to point to data. HDX can be serialised as XML, and in addition can be packed within a FITS file if the data are local. It is independent of the platform and format of the data containers, although for astronomical purposes its natural data holding formats are XML and FITS.

HDX is a particular, simple, Structure Object. From an applications point of view an HDX is a W3C DOM (`http://www.w3.org/DOM/`) which has a top-level element `<hdx>`, and which is valid. It is valid if each of the document element's children is either unknown to the HDX system or, if known, is validated by its declared validator (a software component which HDX can find).

The abstract HDX data model has been implemented in a Java data-access library, but others such as a Perl implementation will be produced. Note however that support for the underlying data containers is distinct from the support for the various HDX types which are defined. Further design aims are to have low (or even zero) overhead to the extent that applications can work using, for example, bare FITS files; to be easy to extend the system to support new types; to be easy to extend the system to support new data storage resources, such as new file formats or a database serving an archive; and to be able to implement these in very efficient ways.

[3]`http://www.starlink.rl.ac.uk/star/docs/sun92.htx/sun92.html`

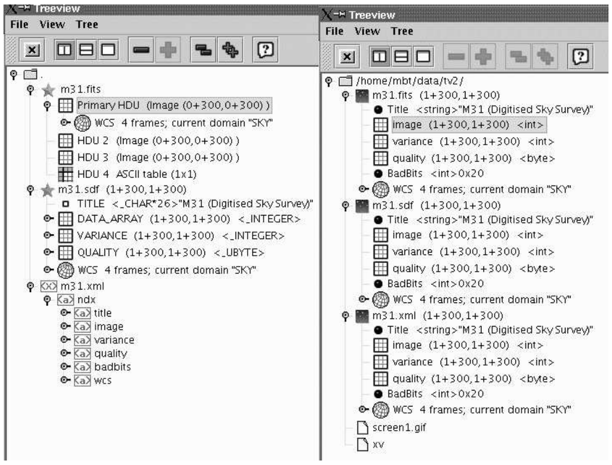

Figure 2. Treeview of the same data as XML, HDS and FITS plus the NDX view where they appear identical.

5. NDX: A Generalised Astronomical N-Dimension Image

NDX represents an N-dimensional chunk of astronomical data and contains pixel arrays for Image data plus Error estimate and pixel Quality. In addition there is World Coordinate System information, History, Title, Units, and User-defined extensions.

Simple operations on NDXs (e.g., ndx1.add(ndx2)) take care of variance, quality, WCS, etc. (where these components are present). Access is available to individual arrays (called NDArray objects) to allow more complex algorithms to be used.

The philosophy and design goals behind NDArray/NDX included being able to process arrays of unlimited size, comprehensive and transparent bad value processing, direct and transparent array access between different formats and location transparent resource naming.

To help to understand the relationship between underlying data containers and NDX, Figure 2 shows an application (Treeview) looking at the same data which is held as FITS, HDS and XML. On the left of the figure one sees the individual components; on the right one sees that all the data is viewable as identical NDX components.

6. Summary

This is a report on work in progress, on the use of Structured data, based on many years' experience. It is expected that there will be changes to support developing VO standards, however we believe that the underlying ideas are sound, practical and extensible, and unique in their format agnosticism. Finally it is worth reiterating that HDX/NDX is aimed at supplementing such data containers as FITS or VOTable with structure information rather than replacing them.