Astronomical Data Analysis Software and Systems IX
ASP Conference Series, Vol. 216, 2000
N. Manset, C. Veillet, and D. Crabtree, eds.

XML at the ADC: Steps to a Next Generation Data Repository

E. Shaya[1], J. Gass, J. Blackwell, B. Thomas, B. Holmes

NASA/RITSS

C. Cheung

NASA/GSFC

Abstract. The staff of the Astronomical Data Center at GSFC (ADC, `http://adc.gsfc.nasa.gov`) is involved in a research project to define the XML format for the metadata of an astronomical repository and for large data tables. In the process, an XML tool box is being developed for importation, enhancement, and distribution of published data and their metadata documents. There is now a working draft Document Type Definition (DTD, `http://messier.gsfc.nasa.gov/xml/dataset.dtd`) which specifies the required elements of content and their attributes. The documentation for each data set will be viewable in several different styles via eXtensible Stylesheet Language Transformations (XSLT, (`http://www.w3.org/TR/xslt`) scripts.

The ADC is actively creating designs for the flow of data through automated pipelines from authors and journal presses into an XML archive, as well as data retrieval through the web via the XML Query Language.

1. Introduction

The data published in the scientific journals generally contain the essential information content of the raw observational or experimental data highly integrated with the discoveries of numerous other projects. It is very useful to have these tables, plots, diagrams, and images available in machine understandable formats that are readily retrievable. They are essential for the basic activities of scientist. Specifically, they provide a greater understanding of ongoing research and aid in the planning of new missions, observations, or experiments. A great deal of researcher time is presently spent searching libraries and on-line databases of individual journal presses to find these data. But these databases and card catalogs are not at all well designed for finding very specific scientific datum. And once found, the data are not in a format that can be ingested into the researcher's analysis package.

The ADC at GSFC/NASA has a long history of reformatting tables and catalogs of tables into machine understandable form and documenting the meta-

[1]Physics Department, U. of Maryland

data descriptions of these data. The ADC is now engaged in a research project to convert this data and metadata into the eXtensible Markup Language (XML, `http://www.xml.com/axml/testaxml.htm`) and explore XML style solutions to automated ingest, data query, and interchange. It is expected that the ADC and the Centre de Données astronomiques de Strasbourg, France (CDS, `http://cdsweb.u-strasbg.fr/CDS.html`) will be ingesting a few thousand tables each year in the near future. The task of ingesting and documenting such a large volume of data will require new processes to be developed. The need for precision search to find specific data in the enormous database that is developing also requires a rethinking of the management and organization. As it turns out, the recent acceptance of XML by the computer science community has resulted in numerous applications and standard practices that appear to be well suited to solve the problems of large and highly complex data repositories such as the one at the ADC.

2. The Advantages of XML

XML provides a concise specification for marking up documents so that they are understandable to a wide audience and are highly reusable. Developers can specify subject specific tags or attributes in order to parameterize or otherwise qualify data in their field. It supports deep structures needed to represent the information content in complex schemas or object-oriented hierarchies. It supports a language specification that allows applications to check documents for structural validity upon creation or importation.

Applications of XML include the ability to improve loadleveling performance. A significant portion of the processing load of the server can be transferred to the client. A simple example of this is in on-line air reservations. Once the origin, destination, and day of travel is established, the server can send all possible routing information in XML format to the client and an applet on the client can then be used for exhaustive search.

More generally, search for data can be tailored to the needs of the individual user. Each element of XML documents are available for independent search. XML documents are easily converted into a highly indexed database for focused search.

XSL and other transformation methods of XML make it easy to tailor views on the data and format of output can be highly differentiated depending on the user's request and the media for display. XSL documents can allow XML documents to be transformed into other XML, or HTML. They can display selected pieces of data, perhaps only those for which the user has permission or clearance. And, the display can be formatted coarsely for small screens or with great finesse for high resolution printing.

3. Document Type Definition

One of our first tasks after defining the scope of the XML project was to develop a Document Type Definition which specifies the structure and content of our metadata documents. It had to support both the information in our legacy documents which we call ReadMe files plus allow for new content types expected in the future plus additional markup for query. The goal of making maximal use of nested hierarchies helped to make a very logical structure in which relationships are made clear and related information is nearby.

We tried to use element names that were simple words that best described the content and avoided abbreviations. This means that when the documents are translated into other languages, say by an automatic translator, the tags can be translated also. Since we plan on having data submitters fill out forms that are generated automatically, this means that the forms can also be translated into other languages.

3.1. Data

XML does not have a native array datatype. One could mark up each value in a data table, but that would create inefficiencies in data transfer and storage that are unacceptable for the large data files at the ADC. We settled upon creating an element type called array that can wrap the data file and provides either the format for fixed width format reads or a delimiter for character separated values. The downside is that a special browser or browser helper application will be needed to interpret the data. One way of specifying this is to use the Notation entities which are in the XML 1.0 spec. Notations are meant for data that is not parsed by the XML parser but is sent to an external application for processing.

We also developed a general purpose XML data format named XDF (`http://messier.gsfc.nasa.gov/xml/XDF_DTD.txt`). This is a small core language that treats tables, sets of tables, images, high dimensional images, animations, and spectra in a similar manner. As each new dimension of the data is introduced within a data structure, axes or table headers are required to enable full use of a single visualization tool common to all documents with XDF as a core.

4. Ingest

4.1. Legacy

The legacy documents at the ADC are in plain ascii. To convert these to our XML markup we developed a general purpose conversion tool called txt2XML. First an XML language for rules of conversion was developed to describe the transformation(`http://messier.gsfc.nasa.gov/xml/ingest_demo/`). These were of the form:

< match start="string1" end="string2" tag="XML_tag" statusOfEnd="include" statusOfStart="drop">

This says grab all text between string1 and string2 where string1 and 2 can be regular expressions that cross line feed boundaries. It also specifies that string2 should be included in the element of type XML_tag and that string1 is dropped.

A Perl processor for the txt2XML language was created that uses XML parsers written in Perl. Then PerlTK was used to provide a graphical interface to debug complicated rules and to provide an easy way to determine causes of errors for nonstandard documents.

4.2. Validation

The XML parsers provide structural validation of documents. That is, it checks that the elements are properly nested and have the right number of occurrences and are in the correct order. Additional applications must be used to determine that the values within the tags are of the correct datatype and form. We developed a tool in which the datatype and/or a regular expression is checked against each of the values in the document to ensure, as well as one can in an automated fashion, that the values are as expected. In addition, we have facilities to convert documents into Web forms in which the values are editable. This greatly eases additional input and minor revisions. Lastly, there are a number of XML editors available commercially.

5. Search

For a search and retrieval tool, we use Java classes to parse the large ($\approx$ 23MB) set of XML documents once and store it on disk as a "persistent DOM", a binary format developed by GMD-IPSI that provides very fast access to any element in the tree (`http://messier.gsfc.nasa.gov/xml/search_demo/`). The DOM is a document object that provides standard interfaces to all elements.

The user's query is entered in an HTML form. Upon submission, a Java servlet translates the search criteria into an XQL query that is executed on the database. The servlet processes the matching datasets with an XSL script to transform the XML results into HTML for display to the user. Soon we hope to deploy a more sophisticated user interface that allows queries to act on any element in the dataset DTD and provides the user a choice of stylesheets for the search results.