Astronomical Data Analysis Software and Systems X
ASP Conference Series, Vol. 238, 2001
F. R. Harnden Jr., F. A. Primini, and H. E. Payne, eds.

ADASS Web Database XML Project

M. Irene Barg, Elizabeth B. Stobie, Anthony J. Ferro, Earl J. O'Neil

University of Arizona, Steward Observatory, Tucson, AZ 85721

Abstract. In the spring of 2000, at the request of the ADASS Program Organizing Committee (POC), we began organizing information from previous ADASS conferences in an effort to create a centralized database. The beginnings of this database originated from data (invited speakers, participants, papers, etc.) extracted from HyperText Markup Language (HTML) documents from past ADASS host sites. Unfortunately, not all HTML documents are well formed and parsing them proved to be an iterative process. It was evident at the beginning that if these Web documents were organized in a standardized way, such as XML (Extensible Markup Language), the processing of this information across the Web could be automated, more efficient, and less error prone. This paper will briefly review the many programming tools available for processing XML, including Java, Perl and Python, and will explore the mapping of relational data from our MySQL database to XML.

1. Introduction

The ADASS POC formed a Web Site Working Group (WSWG), chaired by Richard A. Shaw (STScI), whose charter is to define the scope of the existing `www.adass.org` Web presence. Two of the many issues the WSWG will be addressing are the focus of this paper. These are: 1) the development of an ADASS Conference database, and 2) `www.adass.org` site management. POC member Betty Stobie, of the University of Arizona (UofA), volunteered to take on the task of building a database of participants, invited speakers, and papers presented at past ADASS conferences. To accomplish this, she enlisted help from the other members of the NICMOS Software Team at the UofA, and this paper chronicles their efforts.

2. Building the ADASS Web Database

MySQL was chosen because it is freely available and is a small efficient database server. Three tables were created within this database:

1. Invited speakers: speaker's name, affiliation, title and special topic.
2. Participants: name and affiliation.
3. Programs: presentations from oral, poster, demonstration, and BOF.

2.1. Data Ingest Tools Used, Perl to the Rescue

Most of the source for the data came from online HTML documents found at former ADASS Conference host sites. HTML documents can be created in many ways, sometimes resulting in poorly formed HTML (i.e., missing end tags). HTML is a structural markup language based on the Standard Generalized Markup Language (SGML) metalanguage. While HTML documents do have hierarchical structure, processing HTML documents using an XML parser would require the following steps:

1. preprocess all HTML documents using something like HTML TIDY[1],
2. convert the HTML documents to XML,
3. write a program to parse the XML document to obtain the required data.

Perl, on the other hand, can reduce the process to a single step. Using the Perl modules HTML::TokeParser[2] and HTML::TableExtract[3], an HTML document can be parsed as arbitrary chunks of text, which is a much simpler technique. A Perl program was written to process online ADASS Conference Programs to extract column data such as author, title, and special-topics. The extracted data was used to populate the MySQL *program* table. Starting with ADASS IX and working backward, we found that our original Perl program had to be modified for each conference year. This was expected, and it was easy to tweak the Perl code to accommodate documentation differences.

3. Web Site Management

Now that the ADASS Conference database has been built, how should it be managed? ADASS is a collaboration between several hosting institutions from North America and Europe. The easiest way to provide access to the `www.adass.org` Web Database from all of these institutions, is through a Web development platform. When looking for a Web application development platform, the key criteria are:

- ease of configuration,
- management through the Web,
- support for XML.

The object-oriented nature of the XML API's SAX (Simple API for XML) and DOM (Document Object Model) makes processing XML more suited to object-oriented programming languages. Three such development environments were considered:

1. Webware for Python (`http://webware.sourceforge.net`).
2. Java tools by the Apache XML Project (`http://xml.apache.org`).
3. Zope (`http://www.zope.org`).

Of the three development environments reviewed, Zope was the only environment that met all three criteria outlined above.

---

[1] HTML TIDY, written by Dave Raggett.

[2] HTML::TokeParser, written by Gisle Aas

[3] HTML::TableExtract, written by Matthew P. Sisk

3.1. Zope

Zope was the last development environment to be evaluated and the easiest to configure. Zope is an Open Source application server developed by Digital Creations. Because it is written in Python, it is object-oriented and extensible; moreover, Zope has a large, active community of users. Most importantly, Zope provides through-the-Web management.

3.2. Content Management, Data Access and Data Sharing

Central to Zope is the Document Template Markup Language (DTML). It is a variable insertion and expression language that provides "safe scripting" of Zope objects and dynamic HTML content generation. DTML uses a server-side-include syntax (analogous to PHP:Hypertex Preprocessor). It was built from the ground up to use and extend the Web. Zope encourages building a site whose structure maps to the structure of the content. This is best illustrated by the following piece of DTML code.

```
<dtml-var standard_html_header>
<table>
<dtml-in get_program>
<tr><td><dtml-var adassnum></td></tr>
</dtml-in>
</table>
<dtml-var standard_adass_footer>
```

This example calls an SQL query method *get_program* and builds a table by iterating over the resulting record objects using the "dtml-in" tag and inserting the variable values with the "dtml-var" tag. The two variables *standard_html_header* and *standard_adass_footer* are DTML documents located in the Zope hierarchy, and made available through Zope's concept known as "acquisition". This is very useful for centralizing resources to implement a common design across a Web site. Because objects in Zope are hierarchical, URL's map naturally to objects in the hierarchy. For example, the URL:

```
http://localhost:8080/ADASS/Database/program/
```

will access the *index_html* document object located in the *program* folder.

3.3. Web Management and the Zope Security Model

Zope's security model makes it easy to provide developers access to a Web site without compromising security. Zope manages users with "User Folders", which are special folders that contain user information. An example would be to create a separate folder for each ADASS conference year, managed by the conference host. The host site would link its online registration and abstract submission pages to the `www.adass.org` Zope folder. The host site maintains complete control of that folder, while having access to the common objects in the Zope hierarchy. Other conference specific information would continue to be located and maintained at the host's Web site.

4. XML Support

4.1. ADASS Web Database and XML

Currently the ADASS Web Database provides for XML content through the optional "Download Data" button on any of the database query forms. When querying the database, pressing the "Search" button will render the results into HTML, but by pressing the "Download Data" button, the results will be mapped into XML. The mapping is simple, a table maps to a table element that can contain zero or more row elements. Each row element will contain zero or more field elements.

4.2. Zope and XML

Since Zope is written in Python, program developers have access to the Python XML library through external methods in Zope. Internal to Zope, XML support consists of XML-based protocols such as WebDAV and XML-RPC. There is an "XML Document" add-on and additional XML methods are being developed by Digital Creations.

5. Summary and Conclusions

So, what is the *ADASS Web Database XML Project*? More specifically, what role will XML play in the future of the `www.adass.org` site? The problems we experienced in building the ADASS Conference database showed us that the old way of delivering Web content must change. The `www.adass.org` Web site should make all interfaces capable of delivering XML content. When the client is a browser, the XML can be rendered into HTML. When the client is a program, the XML can be delivered directly to that program via XML-RPC or other mechanisms that support XML-over-HTTP protocol. The main role Zope can play is to become the central site for ADASS Web Database management. Requiring future ADASS Host Sites to link their online registration and abstract submission pages to the Zope site would:

- insure consistency of the data structure from year to year;
- insure consistency in the way participant names and affiliations are entered into the database;
- provide up-to-date information on any upcoming conference (like statistics on categories of papers submitted to date, participants, affiliations, etc.).

The WSWG will be looking at other development platforms. We would like to conclude with one final observation. When choosing a development platform for any Web site, the "keep it simple" scenario keeps returning. It is essential to choose a platform that promises to be the easiest for configuration, maintenance, and code development and still manages to be robust. If it's easy to use, it will be accepted by all, but if it's difficult to use, then individual host sites will continue to re-invent the wheel each year.

Acknowledgments. We are grateful to Dr. Rodger I. Thompson, University of Arizona, NICMOS Project, for providing the resources that allowed us to do this project.