Simplifying the Web User's Interface to Massive Data Sets

Roberta Allsman
Supercomputer Facility
The Australian National University
Canberra, ACT 0200
Australia
robyn.allsman@anu.edu.au
tel +61-2-6125-4154
fax +61-2-6125-8199

Abstract

Many modern projects create massive data sets which are carefully archived in anticipation of further analysis. We present a system designed to simplify the web access to such a data set. The security implications of intermixed proprietary and non-proprietary data are also addressed.

1 Introduction

1.1 Motivation

The Australian National University Supercomputer Facility (ANUSF) manages a mass data storage archive [1] on behalf of the University's academic and research areas. The astronomers, in particular, have embraced near-line data storage for their multi-terabyte data archives. Their initial purpose was simply to reduce the burden of managing their own data tapes. Soon, they also wanted to distribute their data to their world-wide collaborators. The only method possible was to either hand out the password to their mass data storage system account (discouraged due to security reasons) or request that ANUSF set up secondary accounts for these collaborators (discouraged due to the ever increasing overhead of managing accounts for non University affiliated users).

Astronomy data is generally 'owned' by the astronomer for a set period before being released into the public domain. However, the information that data has been taken and the conditions under which it was taken is considered public and is generally maintained in a database by the Observatory managing the telescope. Not so long ago, an astronomer was given a tape containing the night's observations and the Observatory kept a second copy for eventual release. Now, more frequently, an astronomer is given password protected access to his observations which have been archived on a mass data storage system [2].

1.2 Requirements

Recognizing that data maintained on the ANUSF mass storage system needs to be easily accessible to different classes of users (those with access to all data and those with access only to non-proprietary data), the ANUSF developed a web interface model to the mass data storage facility which

- does not require the web user to modify his browser;
- does not require the web user to provide account and password information into his local system prior to data delivery;
- provides data service to the random web user requesting non-proprietary data without requiring preliminary user authentication on the data server;

- minimizes intermediate hops during data transfer; and
- maintains a secure data server yet minimizes account management overhead on it.

1.3 Process Overview

The web user needs to determine which archive data files are relevant and then retrieve that data. This process involves four client/servers:

- a web browser which provides the user with a window into the web's resources;
- a web server which mediates between an HTML-based web browser and a non-HTML-based metadata server;
- a metadata server which provides summary information and locators for archive data files; and
- an archive server which delivers the archive data to the web user.

Figure 1 identifies the basic communications paths between the servers.

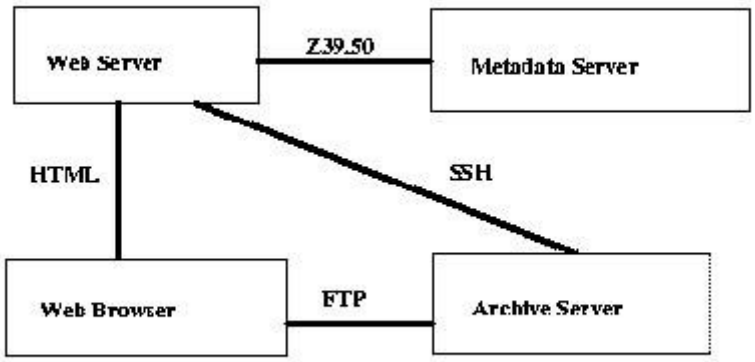


Figure 1. Server Communications Paths

During the discovery phase, the web user fills out a web search form which has been tailored to a specific project's data. Refer to Figure 2 for an example of archive data search selection based on user selected criteria. The web server translates the form's input into the appropriate metadata server syntax and then submits the reformatted query to the metadata server.

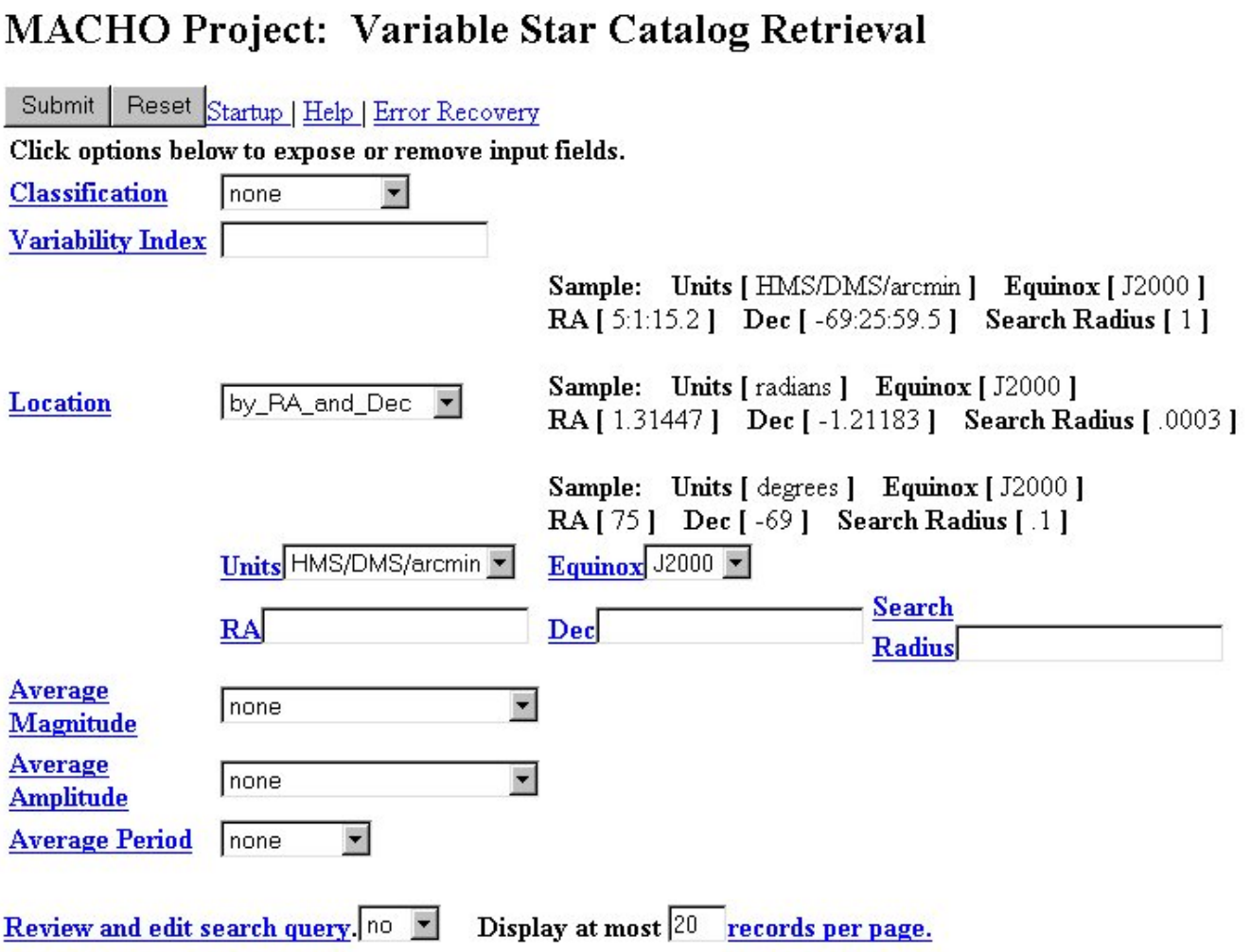


Figure 2. Search Query Form

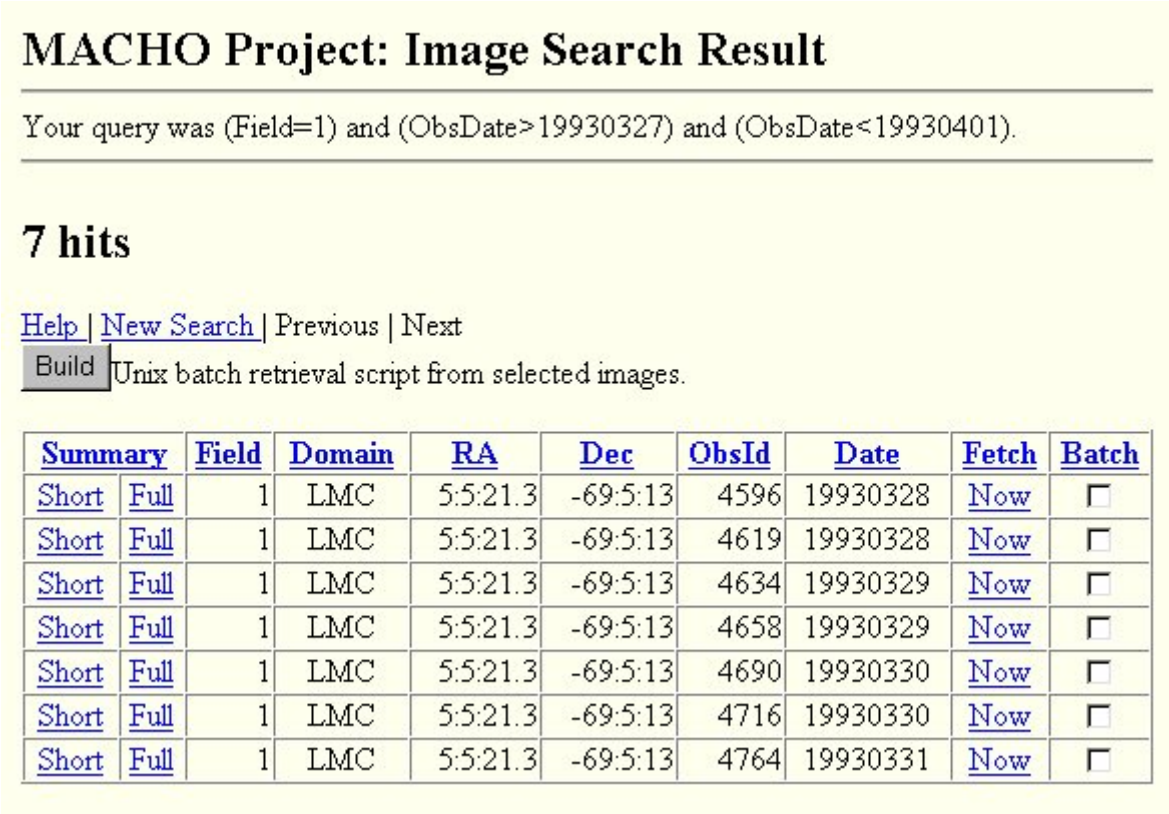
The metadata server might be so trivial it simply returns archive directory listings or the server might be a database management system which maintains relevant attributes, also known as metadata, of each archive data file's contents.

The web server encapsulates the information returned by the metadata server so that the user is provided a means of selecting the archive data to retrieve. The web server authenticates the user's session, informs the archive data server over a secure link of the authentication, and sets the stage for a pre-authenticated, direct data transfer between the web user's browser and the archive data server.

When the web user selects the archive data to retrieve, he is provided with the archive data's locator. The locator might be a plain filename, a hot-button retrieving the data automatically, a form submitted for off-line processing of the archive data, etc.

The action taken on selection of an archive file is defined by the project and encoded within the query's results page. Currently most projects either download a single file in real-time or provide a user-retrievable script for subsequent batch download of multiple files. Refer to Figure 3 for an example of real-time, indicated by **Now**, and batch, indicated by **Batch**, archive file retrieval. Refer to Section 6 for

a description of a project which invokes a specialized web-based data browser enabling real-time analysis and graphical display of the selected data set.



MACHO Project: Image Search Result

Your query was (Field=1) and (ObsDate>19930327) and (ObsDate<19930401).

7 hits

Help | New Search | Previous | Next

Build Unix batch retrieval script from selected images.

Summary		Field	Domain	RA	Dec	ObsId	Date	Fetch	Batch
Short	Full	1	LMC	5:5:21.3	-69:5:13	4596	19930328	Now	☐
Short	Full	1	LMC	5:5:21.3	-69:5:13	4619	19930328	Now	☐
Short	Full	1	LMC	5:5:21.3	-69:5:13	4634	19930329	Now	☐
Short	Full	1	LMC	5:5:21.3	-69:5:13	4658	19930329	Now	☐
Short	Full	1	LMC	5:5:21.3	-69:5:13	4690	19930330	Now	☐
Short	Full	1	LMC	5:5:21.3	-69:5:13	4716	19930330	Now	☐
Short	Full	1	LMC	5:5:21.3	-69:5:13	4764	19930331	Now	☐

Figure 3. Search Query Results

The division of responsibility and ownership of the various components is crucial to the model's acceptance by the various parties involved. Neither the ANU Facility nor the project carries the full onus of data distribution management. The web user is not asked to compromise local security by handing over account access on the user's local system to an unknown web broker.

ANUSF manages the archive data server and the single user account required by the project in order to manage the project's archive data set.

The project manages the web server and the metadata server describing (and locating) project archive data. The project also manages the granting of web password access to any proprietary data within the archive data set. The project is responsible for creating the metadata catalog which summarizes the more important characteristics of the archive data.

The web user is provided access to data via his standard web browser. Although all such transfers are authenticated by the data server, only if proprietary data is requested will the user need to provide a valid web password.

Project staff are generally scientists 'wanting to get on with their work' so ANUSF works closely with the project staff to define and implement the metadata database defining the archive data's attributes and also the web form interface to both the metadata server and to the archive data. After the initial

implementation is complete, the daily maintenance, such as loading new archive data and adding records to the metadata server describing that new information, becomes the responsibility of the project.

2 Metadata Catalog

Development of the metadata catalog involves: the selection of the database manager, the specification of the database record content, the creation of the actual input records for the database, and finally the implementation of the procedures for loading, updating, and maintaining the metadata database.

We looked to the field of library automation to determine an appropriate standard to follow for the use of metadata describing archival material. The Z39.50 protocol standard [3] was widely used for metadata management and retrieval. At the time of our review, the Z39.50 standard was being used by the GIS community [4],[5] whose data is roughly similar, with respect to using geospatial attributes, to the astronomy field.

In retrospect, this was a noble but flawed decision. The standard attribute definitions [6] defined for Z39.50 catalogs are not rich enough to fully describe astronomy metadata. We were required to create our own non-standard mapping of astronomy attributes onto the Z39.50 attribute scheme. This defeats the purpose of using a standard. The use of an SQL-based database, whose query language is familiar to many astronomers, would have been more sensible, especially since the project is expected to update and maintain the metadata catalog subsystem. And, a more recent ad-hoc web scan indicates that SQL-based metadata managers are now in the forefront.

The use of freely available software was mandated by the budgeting constraints of the local community so we searched the 'Net' for a product with an active user community and a small bug list. We selected the Z39.50 protocol suite implemented and supported by Index Data [7]. Their web interface product, ZAP! [8] was initially developed for the US Geological Society and has been extended and generalized for wider use. The suite of tools includes the YAZ Z39.50 applications library [9] , the ZEBRA Z39.50 database server [10], and the ZAP! web-based Z39.50 query engine which is implemented as an Apache [11] module. The Index Data toolset provided a ready built structure in which to implement our proposed security modifications.

The Astronomy community primarily needs a method of distributing their vast collections of observational images. Each image reflects a request, made by an observer, to point the telescope at a specified region in the sky and then capture the image via a CCD camera. The digital version of the archive image conforms to the FITS standard [12] developed by NASA. The FITS file standard defines a header block of <parameter>=<value> information so that, generally, the environmental parameters are co-located with the image itself.

When defining the database record specification, we focussed on the eventual uses of the metadata:

- the summary information which embodies the essence of a particular archive file; and
- the attributes which best distinguish the various archive files from each other. Or to rephrase: the fields used by the scientist when searching for similar data within other astronomy data sets.

The attributes described by the summary information such as: observer, observation date, telescope name, CCD name, etc., become the fields comprising the metadata record. The distinguishing attributes are used as search keys and are generally the information displayed in the search result list. This domain specific information is chosen by the project scientists. Together with the search query form, the

metadata catalog is used to reduce secondary data filtering by the user who is interested in locating only the set containing his domain critical attributes: no more and no less.

Once we develop the database record specification, we then collect and format the information for database ingest. In our case, the data needs to be converted into SGML-like record syntax.

The National Optical Astronomy Observatories (NOAO) developed a FITS parameter naming convention [13] which has been adopted by many observatories for images taken from their telescopes. Associated with the FITS header block naming standard is a complementary logical record definition [14]. This mapping is used, when possible, to define the record specification. One project opted to define their own FITS parameters and corresponding logical record structure.

The information collected for the metadata catalog may come from a variety of sources. Generally, most relevant information is found within the FITS header of the image file. The non-conformist project maintained their environmental information within a purpose built database.

3 Web Pages

Web pages present the results of our labors to the global community. As such, the web pages are developed in close collaboration with project staff in order to ensure they present an accurate and comprehensible window onto the archived data.

3.1 Web Login

On linking to the web address of the project's search query form, if the project supports any proprietary meta or archive data, the web user encounters a login banner. If the project supports a mix of proprietary and non-proprietary data, the random web user may gain access to non-proprietary data by using the standard anonymous login response. Access to proprietary data requires the web user to enter a valid web user account. If the project's data is totally non-proprietary, the login request is not issued and the search query input form is immediately displayed. Details of the web account management are discussed in Section 4.

3.2 Web Search Query

The search query form must cater to varying levels of expertise in the data domain. The interface must enable the neophyte to browse the potential of the archive while also enabling the expert to rapidly and succinctly pinpoint specific archive data.

The demo search query page (see Figure 4) provided by Index Data, which allowed simple logical expressions entered via HTML **form** input, was useful while setting up the prototype package. However, it was soon apparent that a more computationally complex front-end (refer to Figure 2) was required by the astronomers in order to satisfy their "wouldn't it be great if" list of enhancements. The astronomers wanted the freedom to select the units in which most data were entered. For example, when referring to coordinates on the sky (i.e. right ascension and declination), there are three units: radians, degrees, and HMS/DMS, routinely and randomly used. These units are further moderated by the choice of calendar reference (i.e. equinox): B1950, J2000. A front-end query parser needed to convert the user's preferred units into the standard maintained within the metadata catalog.

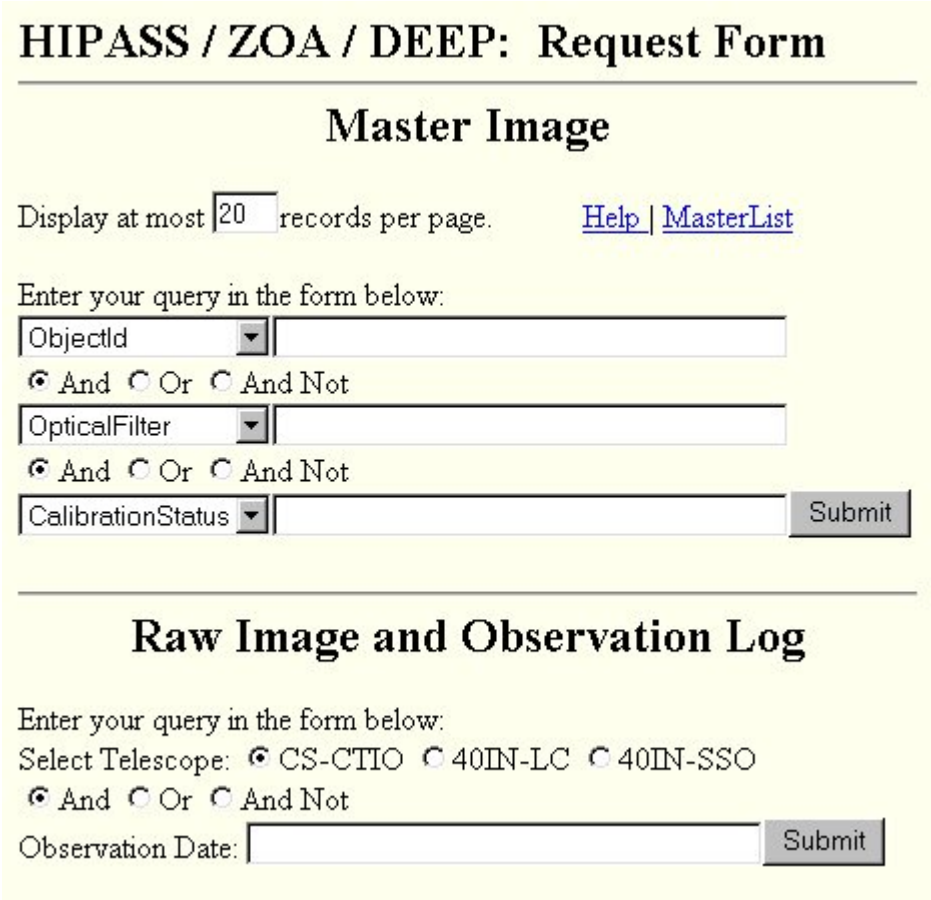


Figure 4. Original Demo Search Query

They also requested the capability to select an image if it overlapped a bounded region on the sky. Since the IndexData database server does not provide this as an atomic construct, an equivalent but complex logical expression needed to be fabricated by the query front-end. And finally, the IndexData database manager doesn't allow floating point numbers to be used as search fields. So although users enter data in domain appropriate units and syntax, the front-end normalizes the values into the integer representation maintained by the database.

Ultimately, the front-end was implemented in Perl [15] using the CGI.pm module [16]. The Perl code emits HTML with a smattering of Javascript [17] used for minor event driven operations impossible to accomplish otherwise. The front-end is recursively invoked whenever the user alters a setting which causes new input fields to become exposed. The front-end validates the user's input and forwards a 'legitimate' Z39.50 search query to the Apache Z39.50 search engine module when the HTML **submit** button is selected.

3.3 Web Search Results

Since the purpose of the search results display is to enable the web user to select relevant archive files for further analysis, each record satisfying the search is summarized in the display. The summary contains attributes, selected by the project scientists, which best characterize their archive files. Refer to Figure 3. Additionally, the summary may contain a variety of active links: to a more comprehensive display of the archive file's attributes; to real-time file retrieval; to an analysis browser activated with the selected archive file; or to embedded HTML **form** input for subsequent batch retrieval of multiple files.

Display of the records, which successfully satisfy the search constraints, is moderated by whether the user has access to proprietary data or not. If the user's access to proprietary data is authorized, an active link to the archive data is available. If the user has no access permission, non-proprietary summary information is presented but no active link is available.

The Index Data Z39.50 web module provides a rich environment to format the response page. The module uses a template to define the full page layout. The template specifies the HTML emitted for each database field in a selected record. Tcl [18] coding may be embedded within the template so that complex constructions, depending on the record's contents, may be fabricated.

4 Security Model

The terms, **authenticate** and **authorize,** have specific meaning in the following sections. **Authenticate** is the process whereby the identity of the requester is determined by virtue of userid/password validation. Upon authentication, the requester's process is granted access to all data and processes available to the authenticated userid. **Authorization** is the process whereby an authenticated userid is determined to have legitimate access to a specific datum or process.

The focus on simplifying the user's web interface posed some interesting problems with regard to transferring the data from the archive to the user's local system. To avoid imposing initial set up requirements on the web user, the package doesn't require either anonymous or password protected access into the user's local system. This implies that any archive file transfer request must be initiated by the user.

However, the archive data server still needs to protect proprietary data. Direct file transfer requests from the user need to be authenticated as to user identity and authorized for the particular datum. Since the user is both authenticated and authorized by the web server, there were two options:

- the web server could request the file on behalf of the user and then pass it through to the user (tunnelling); or
- the web server could notify the archive server that a specific user is allowed access to a particular datum.

We chose the latter option since tunnelling requires the data to make an extra hop through the web server's system on the way to the user's local system. Since the archive data file size is frequently on the order of 100MB, efficient data transfer is important.

A final issue remained: how would the archive server validate the request from a random user? We chose to pass an authentication token to each party in the transfer. The token needed to be carefully constructed so that it was unique and difficult to fabricate out of context. The fabrication of the token is discussed in Section 4.4.

4.1 Authenticating Web Access

Currently, the Apache **htpasswd** access model is used to authenticate access to a project's metadata query form if the project maintains proprietary information. The initial web account for the project is created by ANUSF staff. Additional and subsequent web accounts are created by the project's staff.

This feature is exploited by the Observatory project which manages access to many independent subprojects, each having collaborators. In this case, account creation is further devolved so that

Observatory project staff create the initial web account for a subproject and then the subproject's manager validates additional web accounts for collaborators. The account userids must remain unique among all users accessing the Observatory project data.

4.2 Authorizing Web Access

The Observatory project has an unusual constraint when considering the proprietary status of data. A subproject's data is proprietary for a given period of time from the data's creation; that period may be altered under special circumstances. This implies that part of a subproject's data will be non-proprietary and part will be proprietary at any given instant.

The projects prefer that metadata be considered 'write-once, read-many' so that modifications after ingest are to be avoided. Thus we did not want to embed a date indicating the end of proprietary status since a change in the proprietary period would involve modifying all the subproject's records. We solved the problem by (1) adding a new field, <projectid>, to the metadata record definition which indicates which project owns the data; and (2) creating a secondary database which maps <projectid> with the period of time the data remains proprietary. Validating web access to the archive data is now a matter of checking, when the request is from a non-authenticated user, if the proprietary period has elapsed for the particular archive file.

To summarize the authentication and authorization requirements:

- each authorized user of a project's web server is assigned a user identifier (UID);
- each project identifies one user, by his UID, as the project authenticator (PA);
- each PA is empowered to authenticate additional UIDs to access his project's proprietary data;
- each web server project is assigned a project identifier (PID);
- each archive data item is assigned to a PID;
- each project may assign a proprietary period such that data remains proprietary from the instant of its creation until the proprietary period has elapsed, and after which it becomes non-proprietary;
- a web user is authorized access to a project's proprietary data only if the user is authenticated using a project UID.

4.3 Simplifying Security Restrictions

Each project has its own security requirements. The security model presented allows adequate flexibility to tailor the level of security to the project's need.

- If the project maintains only non-proprietary data, authentication checks are unnecessary on the initial web query access or on the retrieval request to the data server. In this situation, and with the project's full agreement, we've set up a standard **ftp** server on the mass data storage system with anonymous **ftp** access restricted to their archive data repository.
- If the project maintains only proprietary data, the web password validation authorizes access to all data. Additional authorization during hit summary reporting is unnecessary. The token-based authentication check on the retrieval request to the data server is still required.
- Only when the archive data contains a mix of proprietary and non-proprietary data is the full security model invoked.

4.4 Authorizing Archive Data Retrieval

As each metadata record is retrieved, the user's access of the archived data is authorized if:

- the data is, or has become, non-proprietary; or

- the user's previously authenticated UID is authorized to access that project's data.

When a user is authorized for retrieval of a specific archive file, an authentication token is created and both the token and the archive filename are forwarded to the archive data server. The authentication token and filename together include enough information to enable the data server to identify and validate the subsequent direct request from the user. In fact, the same authentication token is created for each record authorized during a single web session.

The authentication token is also embedded within the search query response page sent to the web user. The project's web page programmer determines how the token will be used and incorporates the token appropriately in the HTML emitted by the web server. For example, if the programmer implements an active ftp link in order to retrieve the archive file, the token will be used as the password to a modified **ftp** server on the archive data server. Currently, the token is used to gain access to archive data files through interactive file retrieval via **ftp** and through batch file retrieval via an automatically generated script. The token is also used by a web-based lightcurve archive data extracter and browser for its transient file naming. Figure 5 clarifies the token and filename distribution process.

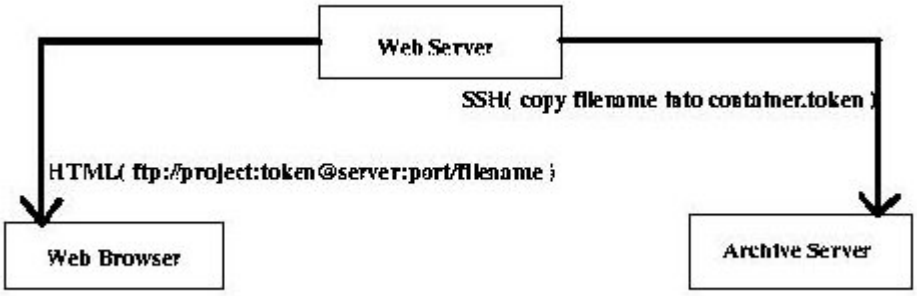


Figure 5. Token and Filename Distribution

The token, which is generated by a modified version of the Apache **mod_uniqueid** module, creates a uuencoded bit stream containing a timestamp and a unique handle to the active web session. The timestamp is used to ensure the retrieval transaction occurs within the project's designated timelimit. Upon expiration of the timelimit, the token is deleted from the archive data server so that any subsequent use of the token is denied. The token lifetime is maintained on the data server within a system configuration file and is on a project wide basis.

A minor modification to the standard **mod_uniqueid** module was required in order to allow the token to be used as an embedded password to the **ftp** server. The original encoding, which allowed an '@' character, was altered to use a different character in order to disambiguate the **ftp** command syntax.

The transmission of the authentication information from the web server to the archive data server is accomplished via a secure shell [19] to ensure tamper-proof and spoof-free transfer. On the archive data server, a container file is created using the token as its filename. The accompanying archive filename is appended to the container file. The container file accumulates the names of all archive files which the current token, and hence user, is authorized to retrieve and which have already been summarized in search result pages.

5 Archive Data Retrieval

An **ftp** server on the archive data storage system was modified to restrict access only to requests using the project name and authentication token in place of the more usual <userid> and <password>. Additionally, prior to retrieving a file from the archive, access authorization is validated. First, the timestamp of the authentication token is verified to be within the project's transaction timelimit. Second, access authorization to the specific file is checked against the list of retrievals previously authorized by the web server for that authentication token.

At the moment, this **ftp** interface is the only retrieval method available to projects with proprietary data. The creation of batch file retrieval scripts is based on this **ftp** interface.

The ProFTPD **ftp** server [20] was selected because it was designed and implemented as a secure, concise server which, we felt, was also too new to have many extraneous options grafted onto it. By design, the tool conforms to the structure of the Apache **httpd** server to with which we were already familiar, so modification of this tool was straight-forward.

Each project has a non-archiving directory on the mass storage system which contains transient files related to web access. These transient files include authentication token containers, batch retrieval scripts, and bundled tar files. Each filename embeds the authentication token so that its expiration date is easily derivable.

The garbage collector is a simple script which descends each project's transient directory finding and removing expired files. It is periodically invoked by the Unix **cron** daemon.

6 Building on the Web Interface

The local astronomy community responded favorably to the data search and retrieval interface providing (near) immediate access to their multi-terabyte repositories. One project decided to interface their graphic analysis tools to the web-based data selection and retrieval thus providing web users with the means of browsing the project's archival data. Their non-proprietary data set, modestly sized at 1 terabyte, consists of time-history sequences of star photometry measurements. Figure 6 illustrates the user's browser interface via the **View** selector.

MACHO Project: Variable Star Search Result

Your query was (Field=1) and (Tile=3319) and (Seqn=10:12).

3 hits

Help | New Search | Template Images | Previous | Next | ftp Bundle

Light Curve	Field. Tile.Seqn	Location (J2000)		Variability Index	Classification	# Obs	Obs w/2 Pts	Focal Plane	# Pts	Period (Days)	Magnitude Ave (K-C)	Amplitude Ave	Sup RSA	Sig	Chi2r
View	1.3319.10	RA	5:1:15.2640	24.11	none	1077	490	r	497	89.2214	15.702 V	0.226	3.11	0.073	13.12
Bundle ☐		Dec	-69:25:59.5200					b	1041	88.7447	14.084 R	0.265	3.75	0.1	20.86
View	1.3319.11	RA	5:1:16.2000	1.64	none	1077	541	r	548	0.1226	16.966 V	0.052	1.06	0.029	0.7
Bundle ☐		Dec	-69:25:41.8800					b	1056	0.3314	17.112 R	0.053	1.14	0.031	0.79
View	1.3319.12	RA	5:1:16.8480	1.6	none	1077	548	r	555	0.166	16.958 V	0.124	1.49	0.055	1.96
Bundle ☐		Dec	-69:26:2.4000					b	1043	1.0962	17.206 R	0.13	1.36	0.053	2.12

Figure 6. Invoking the Lightcurve Browser

The first tool ported to the web environment is a light curve browser, CLC [21], which graphically displays the time-history photometry data for a selected star. The tool allows zooming, recalibration of units, and data extraction. Extensive attributes may be displayed, either for a single point or for the whole data set, such as the observation which imaged the data point, the template image used to reduce the data point, and error bars calculated when deriving the photometry. The project is in the process of integrating the use of the tool with two separate metadata catalogs. The first catalog interfaces to a subset of stars whose photometry rises and falls periodically; the metadata catalog includes the statistical evidence for their classification as variable stars. The second catalog will interface to the project's entire 1 TB photometry repository.

Refer to Figure 7 for an example of the web-based lightcurve browser [22]. Selection of any active button initiates recalculation of the lightcurve and redisplay of the page.

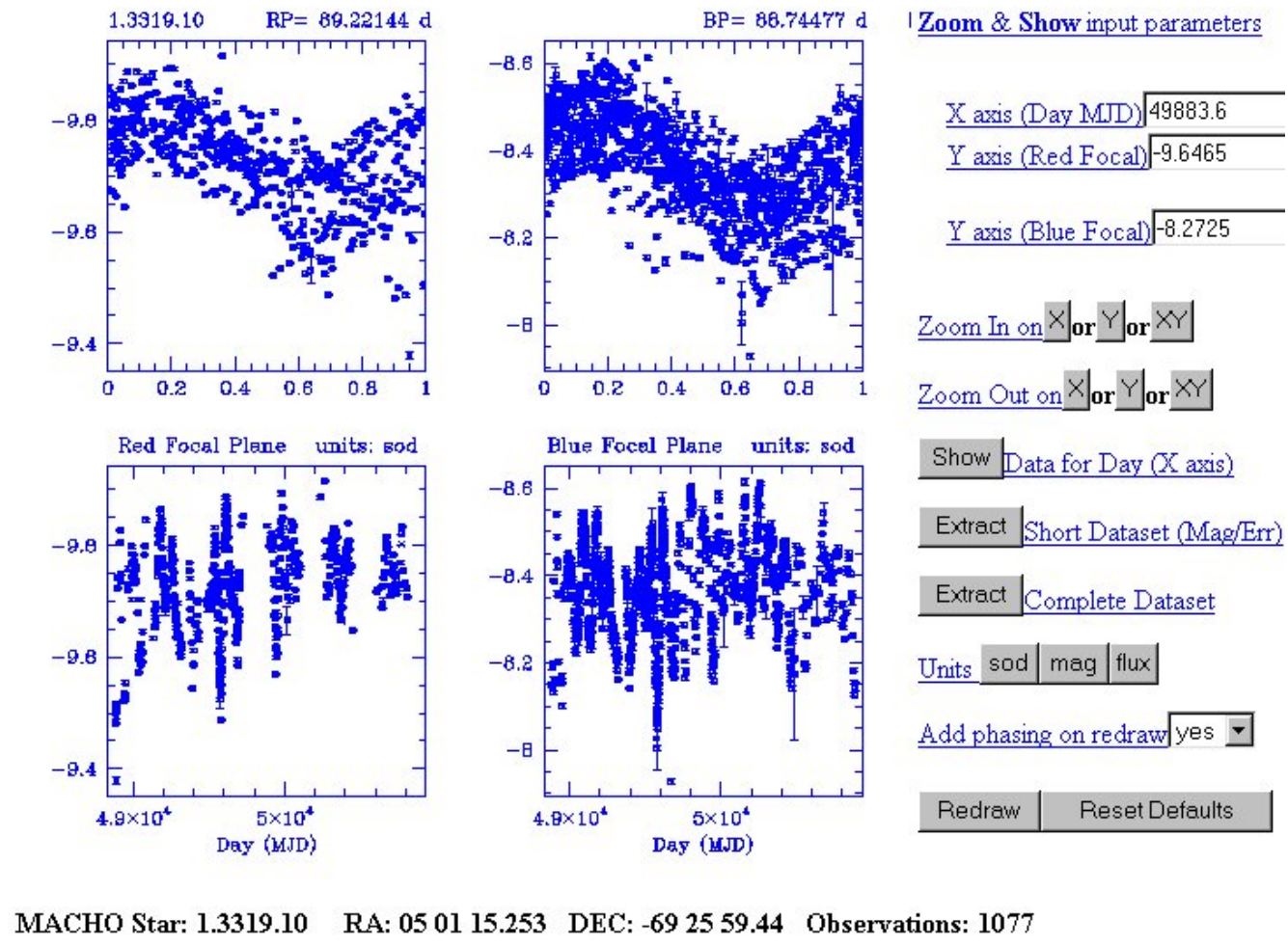


Figure 7. Web-based Lightcurve Display

Porting the tool was both straight-forward and non-trivial. Again, we did not want the random web user to need to modify his existing computing environment. This implied the graphical display was created on the server-side and not client-side. Some issues which had to be addressed included:

- how to maintain history when web interactions are generally presumed to be stateless;
- how to render the actual graphical component in the web display;
- how to mimic mouse driven selection of operations; and later,
- how to stop cacheing of the embedded graphical image.

Major components of the original tool were either removed or totally rewritten in order to convert the tool to standard web behaviour. The web page display is essentially a graphics-decorated HTML **form** on which the user selects operations. State history required by subsequent displays is maintained through HTML **hidden input** definitions. The appropriate graphics, relating to the user's last command, is

created and transiently stored on the web server. The graphics file is referenced as an **img src** in the HTML emitted for the web page display.

The most vexing problem was the undesirable cacheing of the **img src** graphics file referenced in the web page display. Although the graphics content changed on each page redisplay, the name of the transient file was reused to simplify garbage collection of the transient file. Attempts to turn off cacheing via the standard HTML commands failed. However, cacheing can be subverted if a unique tag is appended to the **img src** filename [23]. For example, the **defeat_cache** tag used in:
<img src="/~macho/OgHuuZbLBSwAAAPIBiI_1.3319.10.gif?defeat_cache=973562207">
was assigned the current date and time in order to ensure its uniqueness within each display page.

One important feature is yet to be implemented: mouse driven input is currently mimic'ed by keystroke input. In the future, the use of imagemap cursor management will facilitate mouse driven input.

7 Lessons learned

Many lessons were learned over the course of the web interface development and its subsequent use by the various astronomy projects. Throughout the paper we've referred to technical decisions we've reconsidered and why. But some of the lessons dealt with the intangible traits of human nature.

We initially designed our web query forms to include active links on keywords which brought up explanatory information on syntax, semantics, and sample use. We soon discovered that many users avoided the keyword links, not because they didn't know how to use them, but because they wanted to avoid the infinitesimal page redraw time. The user's requested solution was to provide terse usage examples within the search form's display in order to guide them into appropriate usage. The comprehensive documentation link remains as backup in case the user needs more help.

We also learned it's important to match the level of ongoing web maintenance required with the programming commitment of the project. One project, small in manpower but large in archive size, constructed an archive interface which required routine updating of the metadata catalog so that comprehensive searches on various attributes could be made. They also included a simpler access method similar to a directory listing. The project has never updated the catalog themselves but instead has devolved into using the directory listing retrieval. Access to the project's data is restricted to project members so perhaps their intimate knowledge of the data removes the need for an attribute oriented search engine.

8 Future Work

Although the basic web access system is functional and being used by a number of projects [24, 25, 26], there remain additional tasks to be done which would further enhance the package. We've identified the following areas:

- To protect against web password compromise through packet snooping, the Apache **httpd** server will be wrapped by a secure socket layer using **apache_ssl** [27] in order to encrypt all web traffic. To date, the projects haven't felt the extra level of security is necessary.
- With the success of the web based lightcurve browser, we have additional requests for back-end analysis tools to be interfaced to archive data repositories.
- The metadata catalog needs to be converted to a database more conducive to scientific data sets.
- Updating the metadata catalog needs to be totally automated. And, finally,

- The baud rate between Australia and the rest of the world needs to be increased so that long-haul data movement completes before user frustration manifests.

9 Summary

We undertook this project since we feel strongly that massive data archives contain a rich lode of information which is under utilized due to the difficulty of selecting and retrieving relevant data. The web access system described enables projects to distribute their data conveniently, safely, and efficiently to the web community.

The astronomy projects we've worked with are pleased with the web based access of their archive data repositories. One might say they feel noble about releasing their data to the entire world. However, they don't always commit continuing project resources to maintain their portion of the web package since it is, more frequently, viewed as a useful by-product of their scientific efforts and not as a scientific deliverable.

References

[1] Mass Data Storage System, November 2000, <http://anusf.anu.edu.au/MDSS>

[2] Hubble Data Archive, Hubble Space Telescope, November 2000, <http://archive.stsci.edu/hst>

[3] "Information Retrieval (Z39.50-1995): Application Service Definition and Protocol Specification, ANSI/NISO Z39.50-1995", Library of Congress, July 1995, <http://lcweb.loc.gov/z3950/agency/document.html>

[4] Douglas D. Nebert, "Z39.50 Application Profile for Geospatial Metadata or 'GEO', Version 2.2", US Federal Geographic Data Committee, US Geological Survey, Reston, Virginia, 27 May 2000, <http://www.blueangeltech.com/Standards/GeoProfile/geo22.htm

[5] Benjamin Hatton, "Distributed Data Sharing". The 25th Annual Conference of the Australiasian Urban and Regional Information Systems Association, November 19-21, 1997

[6] "Registry of Z39.50 Object Identifiers", Library of Congress, September 20, 2000 , <http://lcweb.loc.gov/z3950/agency/defns/oids.html>

[7] Index Data, Denmark, <URL:http://www.indexdata.dk>

[8] "ZAP Administrator's Documentation", Index Data, Denmark, June 2000, <URL:http://www.indexdata.dk/zap>

[9] "YAZ User's Guide and Reference", Index Data, Denmark, September 2000, <URL:http://www.indexdata.dk/yaz>

[10] "Zebra Server - Administrator's Guide and Reference", Index Data, Denmark, March 2000, <URL:http://www.indexdata.dk/zebra>

[11] Apache Software Foundation, Apache HTTP Server Project, <URL:http://httpd.apache.org>

[12] "Definition of the Flexible Image Transport System (FITS)", NOST 100-2.0, NASA/Science Office of Standards and Technology, NASA Goddard Space Flight Center, Greenbelt, Marlyland, Marcho 29. 1999, <http://archive.stsci.edu/fits/fits_standard/>

[13] "NOAO FITS Keyword Dictionary: Version 1.0 ", National Optical Astronomy Observatories, Tucson, Arizona, January 2000 <http://iraf.noao.edu/projects/ccdmosaic/Imagedef/fitsdic.html>

[14] "Classes Describing Astronomical Observations", Francisco Valdes, National Optical Astronomy Observatories, Tucson, Arizona, January 27, 2000 <http://iraf.noao.edu/projects/ccdmosaic/Imagedef/classes.html>

[15] "The Source for Perl", <www.perl.com>

[16] "CGI.pm - a Perl5 CGI Library", 9/13/2000, Cold Spring Harbor Laboratory, Cold Spring Harbor, New York, Lincoln D.. Stein <http://stein.cshl.org/WWW/software/CGI/>
[17] "JavaScript Reference " , <http://developer.netscape.com/docs/manuals/communicator/jsref/index.htm>
[18] "Tcl Developer Xchange", 2000, <http://www.scriptics.com>
[19] "The Secure Shell", 2000, <http://www.sshorg/>
[20] "ProFTPD : The Professional FTP Daemon", 2000, <http://www.proftpd.net/docs>
[21] "CLC Lightcurve browser", John Doug Reynolds, MACHO Project, Mt. Stromlo and Siding Spring Observatories, Australian National University, April 1998
[22] "Variable Star Catalog Search Form", MACHO Project, Mt. Stromlo and Siding Spring Observatories, Australian National University, November 2000, <http://store.anu.edu.au:3001/cgi-bin/varstar.pl>
[23] Private communication on 20 June 2000 with Ancilla Allsman, Linuxcare, Inc. who queried and then passed on information provided by Rasmus Lerdorf, Linuxcare, Inc.
[24] "WFI - Wide Field Imager", WFI Project, Mt. Stromlo and Siding Spring Observatories, Australian National University, 4 May 1999, <http://msowww.anu.edu.au/observing/wfi/intro.shtml>
[25] "Optical Follow-up Program of the Multibeam HI All-Sky Surveys (HIPASS & ZOA) at Parkes Radio Telescope", HIPASS Project, Mt. Stromlo and Siding Spring Observatories, Australian National University, 27 October 1999, <http://msowww.anu.edu.au/~jerjen/multi/MBOFP.html>
[26] "The MACHO Project", Mt. Stromlo and Siding Spring Observatories, Australian National University, 31 October 2000, <http://wwwmacho.anu.edu.au>
[27] "Apache-SSL", 7 November 2000, <http://www.apache-ssl.org/>