From LBT to TNG: an easy way to inherit archiving system

C. Knapic, R. Smareglia, M. Molinaro
INAF - Osservatorio Astronomico di Trieste

Contacts: knapic@oats.inaf.it - http://ia2.oats.inaf.it/

Abstract :
Large Binocular Telescope Archiving System and Web Interface is developed to be extremely flexible about adding new instruments. This flexibility is very useful also in the case of an archiving system rebuilding and extension for another telescope, in our case Telescopio Nazionale Galileo (TNG). This article presents the archiving system structure, programs and web applications adaptations done in order to ingest, transfer and distribute files and web user interfaces. The interesting thing is that database structure is strongly modified to easy ingest any fits file like TNG ones but this is done with a very little code adjustment.

General Requirements
Telescope Data Archive must provide the storage, maintenance, efficient transmission and release of all scientific data to the Astronomical Community with an appropriate policy and by VO standards.
Very loose constraints are needed to guarantee correct data insertion in archive:

➔ *Scientific constraints*
- Astronomical data must be archived in standard FITS format (to verify your files go to: http://fits.gsfc.nasa.gov/fits_verify.html)
- Some needful information must be coded in FITS header:
  - File name;
  - Acquisition date and time in YYYY-MM-DDThh:mm:ss.sss format;
  - Instrument / Telescope / Camera;
  - Coordinates;
  - Owner / Partner / PI;
  - Data Type / Image Type.

➔ *Hardware requirements:*
- They are directly dependent on telescope acquisition rate and life cycle;

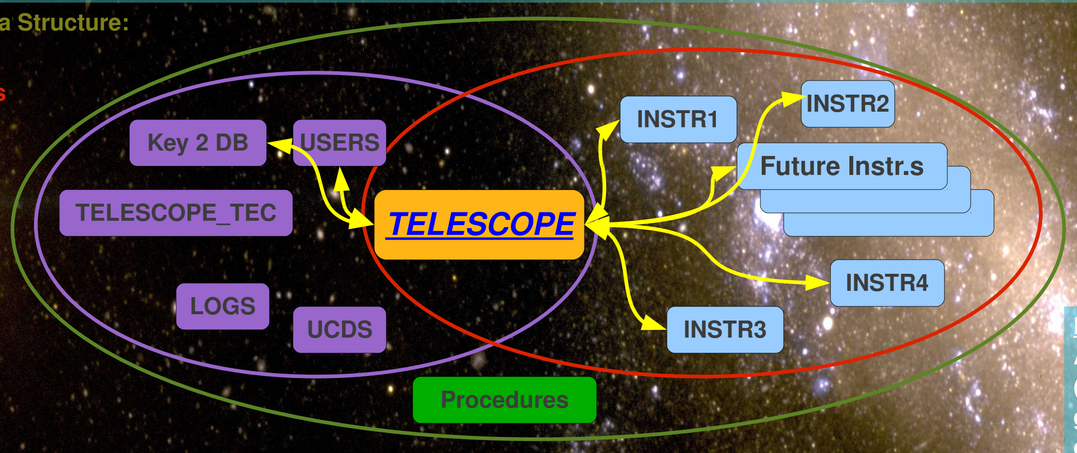
➔ *Software requirements:*
- Linux OS (currently CentOS and SUSE are used);
- Java Runtime Environment (JRE);
- Relational Database Management System (MySQL, Oracle etc..)
- Apache Tomcat web server or GlassFish web server;
- Some system utility or libraries like fuser, inotify/jnotify, tar (archiving tool), CFITSIO (fpack,fitsverify etc.), ssh , rsync …

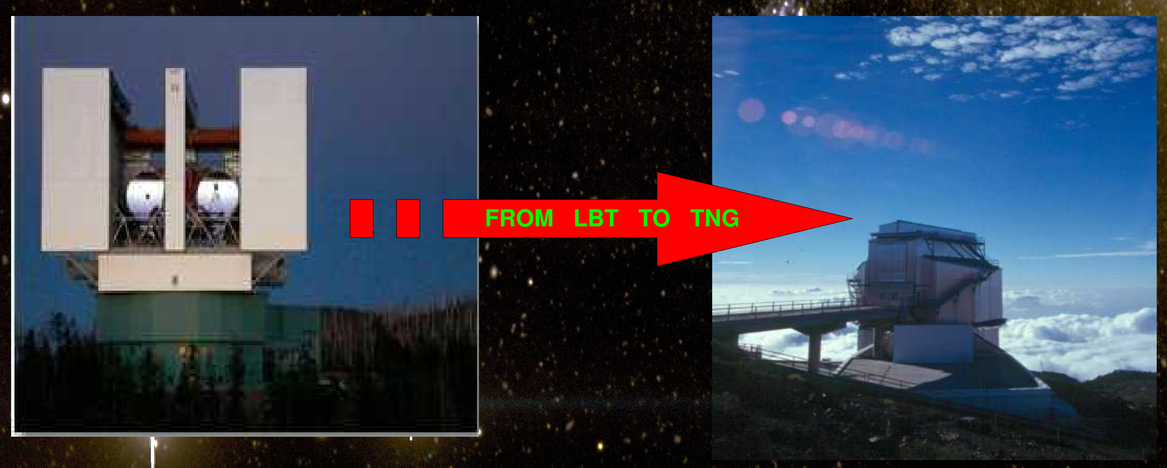
➔ *System environment variable and folders hierarchy:*
- The archiving system needs a dedicated system user in order to set some environment variables that manage database accesses, input folders locations and storage location.
- To ingest data into archive (Jout process) the input directory ($DB_INDIR) must have two dedicated subdirectories: tmp and warning. 'tmp' is used to manage data outside the incoming directory and 'warning' to handle non compliant fits files, errors and corrupted files. Those files should be repaired and re-elaborated.

➔ *Database structure:*
- The idea is to implement the faster and as easy as possible structure, with
  - a dedicated table for each instrument about scientific metadata;
  - a technical table to store informations about location, ownership, status and time of ingestion, status of delivery to remote hosts and the technical most important informations;
  - a generic table in which store basic and generic information like coordinates, data type, acquisition date etc..;
  - a user dedicated table in order to allow automatic management of permissions, destinations and user type;
  - a key-to-column_name table in which are coded the relations between header keywords and columns name and columns types;
- Two web dedicated tables for logging and ucds definitions;
  - a log table to store informations about accesses to web services and interfaces;
  - a 'Unified Content Descriptor' table about appropriate definitions of header keys.
- Tree procedures to check, organize and handle possible exceptions to the general ingesting model:
  - K2db procedure to check if all tables are consistent each other and consistent with those information should enter in db;
  - radec_own<telescope_name> procedure to convert coordinates from radiants to hours and viceversa;
  - tab2<telescope_name> procedure to duplicate informations from instrument dedicated table and generic telescope table (if needed)

The Archiving tools are developed for a Distributed Archive and therefore could also deliver data in different location by ssh and rsync.
LBT/DA has been developed by IA2 (Italian Center for Astronomical Archives) team. Web Page: http://ia2.oats.inaf.it/

Database Schema Structure:
Technical tables
Instrument tables

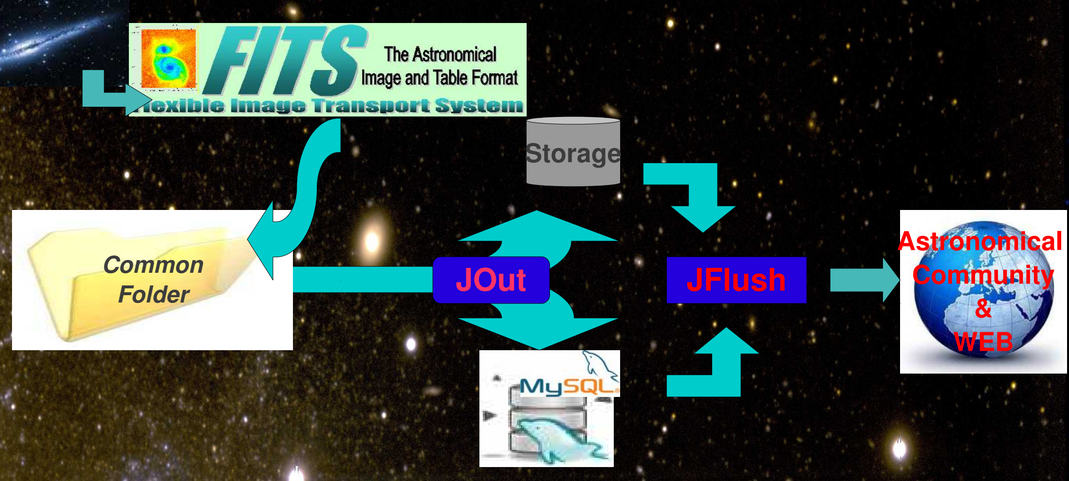




Two different telescopes need two different archives but they can use the same tool!
Standard FITS files and a dedicated database schema can solve every problem of archiving, because all the specific peculiarities of each instrument lies in the key2db table and in database procedures. Using different Users (for es. archa_LBT@localarchive and archa_TNG@localarchive), different system variables have to be used to set different configurations and distinct accesses to the database, tables and procedures. Key2db table guarantees the correspondences between the fits header keywords and the columns name and type, columns that store the instrument file metadata. Procedures are telescope dedicated in order to meet the instrument specific necessities, features and capabilities. To build the new archive it is sufficient to load the FITS files in the incoming directory and wait all data are managed by Jout.

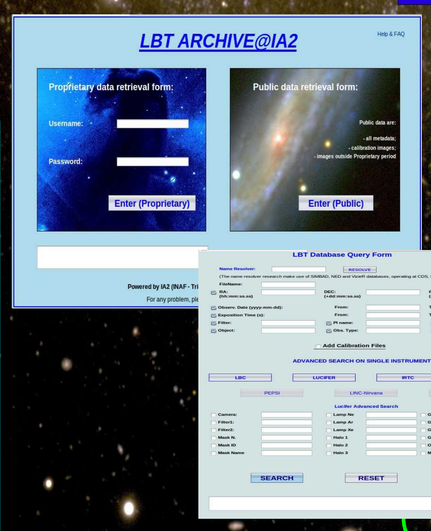
Two different Telescopes needs two different web sites but the existing one can easily be adapt!!
Redoing the "make-up" of the first and second web page of the user interface, a intuitive and user friendly web interface is ready to be used. At the moment the restyling of the web pages is manually done but for the future we plan to automate the process.

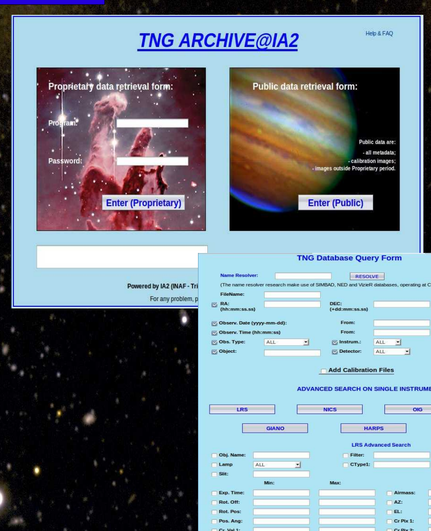
Data handling.
Data handling applications can be distributed throughout different archive sites. Depending on location, they can be use in different mode in order to guarantee flexibility and safety.
Data management:
Newly acquired data have to be temporarily stored in a common folder. This folder is monitored by a Java system utility (JNotify) that starts the manage files process (JOut).
At the moment a MySQL RDMS is used to store metadata, but it could be easily adapt to any relational database management system based on SQL. Once a file is detected, JOut inserts a subset of a fits keyword in a instrument dedicated schema table and in a general telescope table. A technical table contains specific file informations like file location, file status, transfer file status and so on.
Some operations are performed on metadata using MySQL procedures to adjust occasionally forgotten keywords or null values, and to synchronize instrument tables with telescope table. Then a new record is inserted in database and the corrispondent file could be moved/stored in Archive and, in case of the Main Archives, also in a well defined Repository directory, directly accessible from local computers. In the Archives, files are first compress and then stored, while in Repository files are stored as they are. Compression could have two different modes: gzipped (.gz) or fpacked (.fz): the selection between the two should be done at JOut start. Gzip format is the most used mode. If an error occurs, different alternatives are possible depending on error type: in case of fatal errors (Jnotify error), JOut terminates while in case of errors related to fits file content the behaviour is to move the file to a warning directory and to report the error in database. In any case, errors are reported in a dedicated log file.
Data forwarding:
Once data are correctly inserted in database, stored in archive and all updates are performed, files can be sent to recipients (Partners/Owners). Depending on which recipient is selected, a java multi-threading program (JFlush) finds the associated URL for the recipients and search for files with that ownership in db. It use a blocking queue to create a list of commands to be executed and a "WorkerThread" object that perform the operations as indipendent threads. The constructor takes in input an array of strings (list of commands), the number of threads to make run simultaneously and the owner of data. It works istantiating "n" WorkerThread objects that execute the command string, that are the first n-elements availables of the blocking queue list. Eeach thread executes independently and when one thread ends another starts the next command, till the end of the list. The procedure is terminated when the list is empty and a special string is sent. The command is basically an invocation of a shell script that uses Rsync. Rsync software application minimizes data transfer and serves files in a remote shell SSH mode. If Rsync ends successfully, a flag with information about destination and current time is set, else a "communication Interrupted" is signaled. Every "interruption" is re-entered in the blocking queue for a later run of Jflush. Program is provided of a detailed error log file. All metadata in all databases are synchronized.

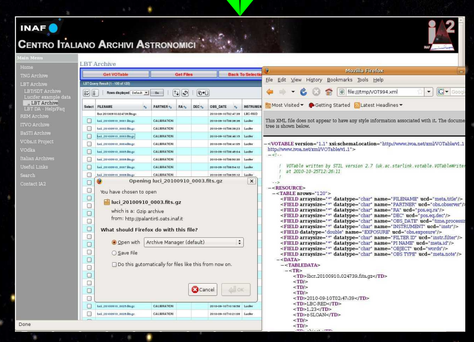


Data Retrieval: User Interfaces & Web Services
A IVOA compliant service (SIAP and Cone Search VO-compliant WS) and user-friendly web application (User Interface) is easily adaptable and can work also in clustered web servers. The UI could perform general query on all data or a more specific query about a single instrument. Release of public data is dependent on instrument specific policy, but in general all metadata are accessible and they are issued in the form of simple VOTable. Currently all files, VOTables and tar archives are served directly from the UI. About the web services, files are served by a FileServer, a web service that works as an applications-indipendent validator for the incoming file request: if the query fully meets the file access policy, the required file is served, else an exception is thrown. The telescope dedicated application is intended to serve administrative, public or user specific use. In the first case all data (files and metadata) present in archive are available. In the second case, all metadata and policy free files are available. In the third case, all metadata and files of a specific user are available to download.
Policy can be defined at database level and manage all operation of UI and web service.
For more information on VO-WS on IA2 please visit VO-Dance page:
http://ia2.oats.inaf.it/index.php?option=com_content&view=section&id=20&Itemid=110

Web Applications







Conclusions:
- Java applications, MySQL procedures and shell scripts handle files automatically from receipt from the instruments through delivery to the user.
- All metadata and non proprietary data are avalaible to the Community and proprietary data are secured by sturdy logic and dedicated database query.
- VO compliant services are working.
- Flexibility and easy adaptation to instrument requirements make this tool to serve other telescopes very easy . A tool to easy build a UI web application is planed to be developed so as to allow users to build their own application by themselve for all the data (fits format), needing only a database and a web server.
- Data managed from IA2 staff will shifted on this new tecnology for first.