



LBT Distributed Archive: status and features

C. Knapic⁽¹⁾, R. Smareglia⁽¹⁾, D. Thompson⁽²⁾, R. Gredel⁽³⁾

⁽¹⁾INAF - SI / Oss. Astronomico di Trieste, ⁽²⁾Large Binocular Telescope Observatory, ⁽³⁾Max-Planck-Institute fuer Astronomie
(*) knapic@oats.inaf.it, (*) smareglia@oats.inaf.it, (*) dthompson@as.arizona.edu, (*) gredel@mpia-hd.mpg.de

Abstract :

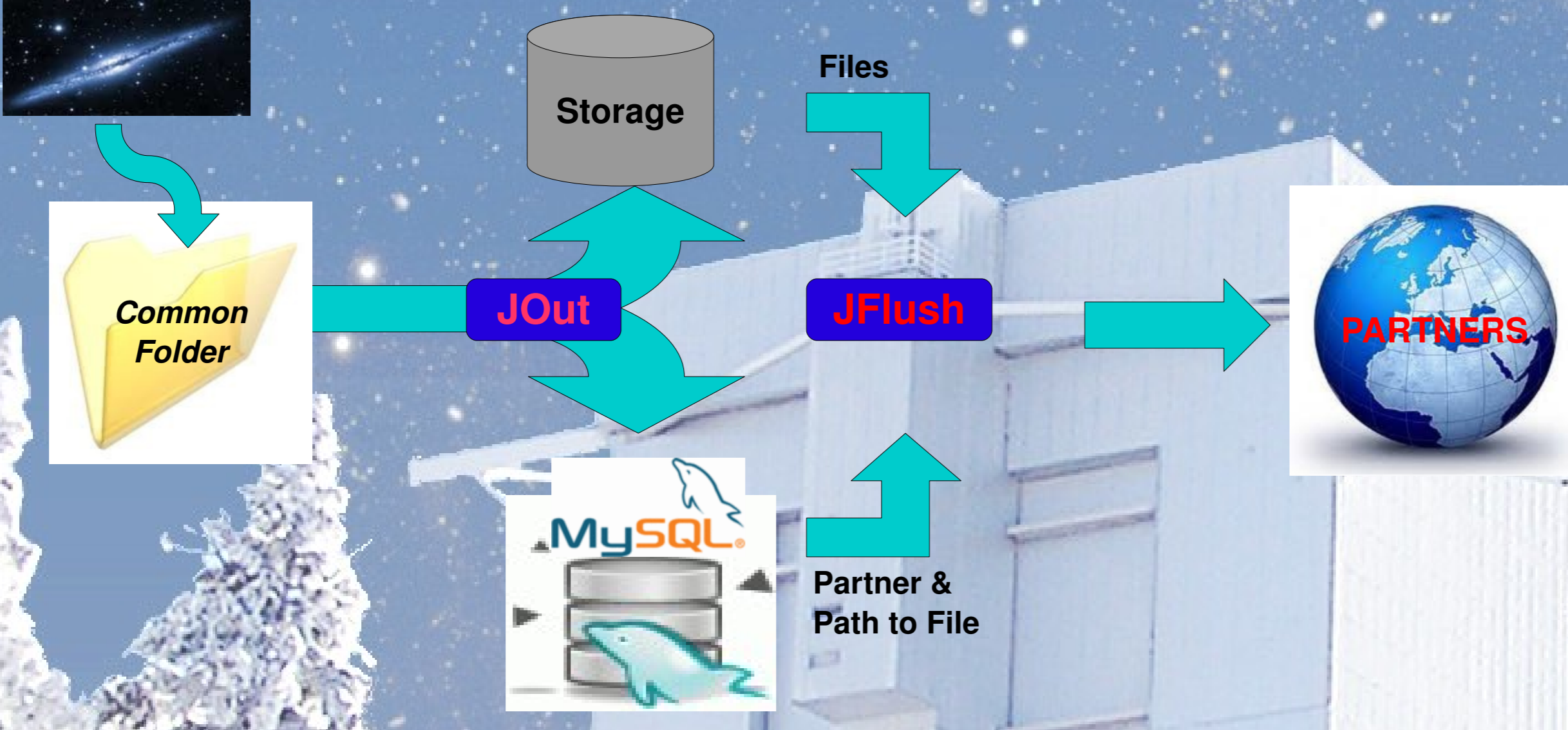
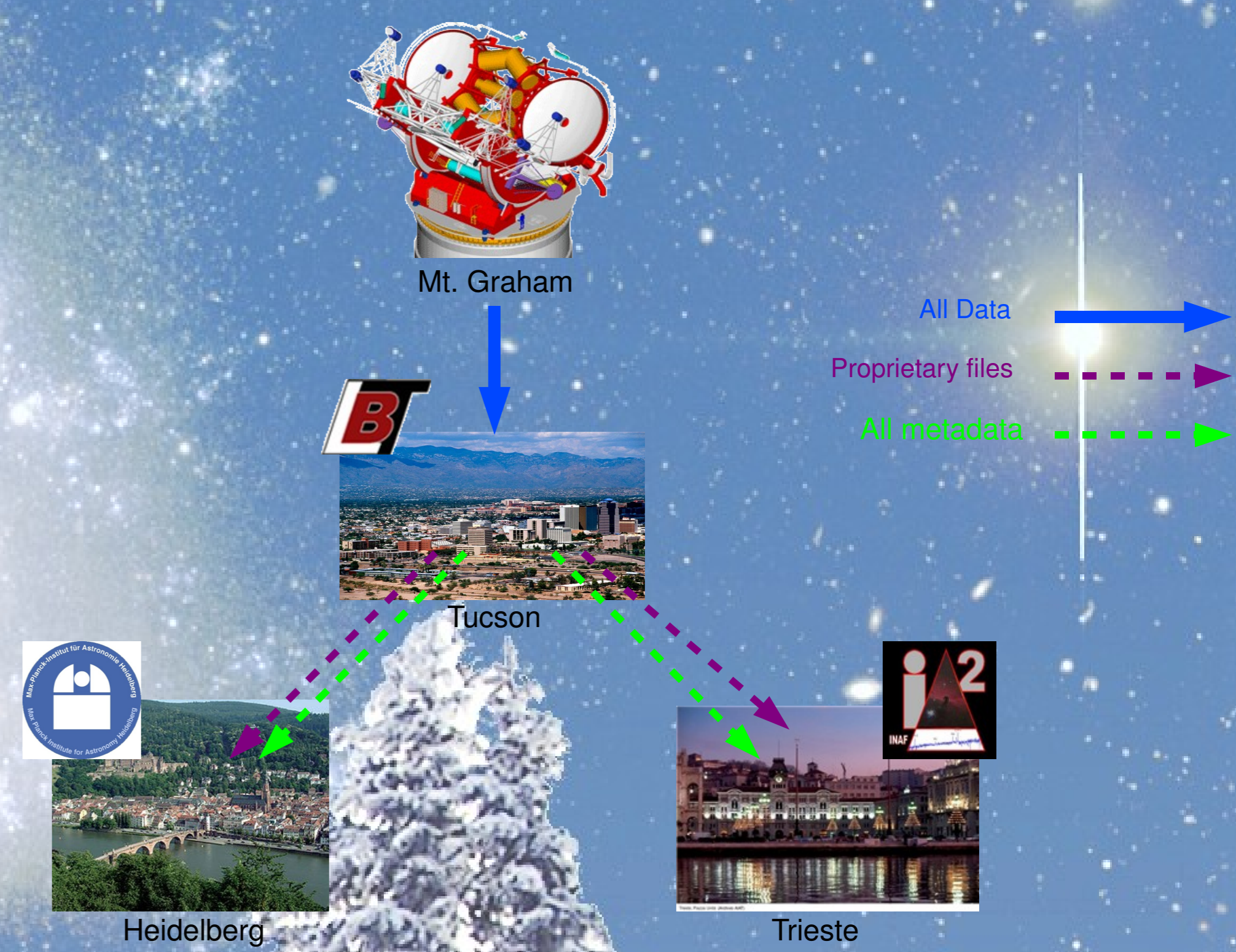
After the first release of the LBT Distributed Archive, this successful collaboration is continuing inside the LBT corporation. The IA2 (Italian Center for Astronomical Archive) team had updated the LBT DA with new features in order to facilitate user data retrieval also respecting VO directives. A database migration, a new data distribution software, as well as some additional features in the LBT User Interface has been developed for easy integrate any new instrument/s. The DBMS engine has been changed to MySQL and optimized to easily adapt to insertion of data from new instruments. Consequently, the data handling software now uses java thread technology to update and synchronize the Mt. Graham, Tucson (main storage archives), Trieste and Heidelberg archives with all metadata and proprietary data. The LBT UI has been updated with additional features, allowing users to search by instrument and some more important characteristics of the images. Finally, instead of a simple cone search service on LBT image over all data, new instrument dedicated SIAP and cone search services have been developed and their publication in the IVOA framework is planned to be during this autumn.

Requirements and general choices.

LBT Data Archive must provide the storage, maintenance, efficient transmission and release of all LBT scientific data to LBTO and Astronomical Communities with an appropriate policy and by VO standards.

LBT/DA has been developed from IA2 (Italian Center for Astronomical Archives) team. Web Page: <http://ia2.oats.inaf.it/>

- The structure of the whole database and archive is thought to easy integrate any new instrument/s data of fits format, allowing an insertion of new instrument related fits-header keys if they respect fits standards (<http://fits.gsfc.nasa.gov>).
 - The technologies used are open source, VO compliant and multithreading Java technology is used to send up to ten files at time in order to maximize the efficiency of transmission band.
 - Softwares are fast and have an robust error tracking, in order to monitor all files operations.
- The two main archives, located in Arizona (Mt. Graham and LBT@Tucson), are identical and store all LBT data. They provide an efficient distribution of proprietary data to the Trieste and Heidelberg archives. All metadata and calibration files are present in each archive, while proprietary data is sent only where appropriate. At the moment, only INAF raw data (Trieste) have a one year proprietary policy after which files are considered freely distributable.
- At all archive sites (Tucson, Trieste and Heidelberg), a User Interface (UI) for data retrieval are present and allow enabled users to download single fits files, tar archives with more than one fits file or a simple VOTable describing the current results of a query. The UI and the consequent query results are formed depending on user account type (public, administrator, partner or pi-name).
 - SIAP and Cone Search VO-web services are ready to be published; they will release LBT-INAF public raw data to VO Alliance clients.



Data Retrieval: User Interface & Web Service

A IVOA compliant and user-friendly web application (User Interface) is ready and it works also in clustered web servers. It ingests data from all LBT working instruments and is ready for the new ones. The UI could perform general query on all LBT data or a more specific query about a single instrument. Release of public data is dependent on Partner 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. In the future we plan to use of a FileServer, a web service that works as an applications-independent 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. LBT application is dedicated to 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 just the calibration and INAF public files are available. In the third case, all metadata and files of that user and of calibration are available to download. Another developed application is the LBT web service, a SIAP and Cone Search compliant WS that is under publication. It serve a VOTable with all available public data, while the delivery of the files is delegated to the FileServer service. N.B. Only INAF partner have a data policy: all INAF raw data older than one year are public.

Data handling: Management and forwarding.

Data handling applications are distributed throughout the four archive sites. Depending on location they are use in different mode, but the code is the same in order to be consistent everywhere.

Data management:

Newly acquired data at the LBT are temporarily stored in a common folder. This folder is monitored by a Java system utility (JNotify) that starts the manage files process (JOut). A MySQL RDMS is used to store metadata. Once a file is detected, JOut inserts a subset of a fits keyword in a instrument dedicated schema table and in a general table (LBT) of all the LBT data. The general LBT table contains also technical informations like file location, file status, transfer file status and so on.

Some operations are performed on metadata using MySQL procedures to adjust occasional forgotten keywords or null values, and to synchronize instrument tables with LBT table. Then a new record is inserted in database and the correspondent 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.

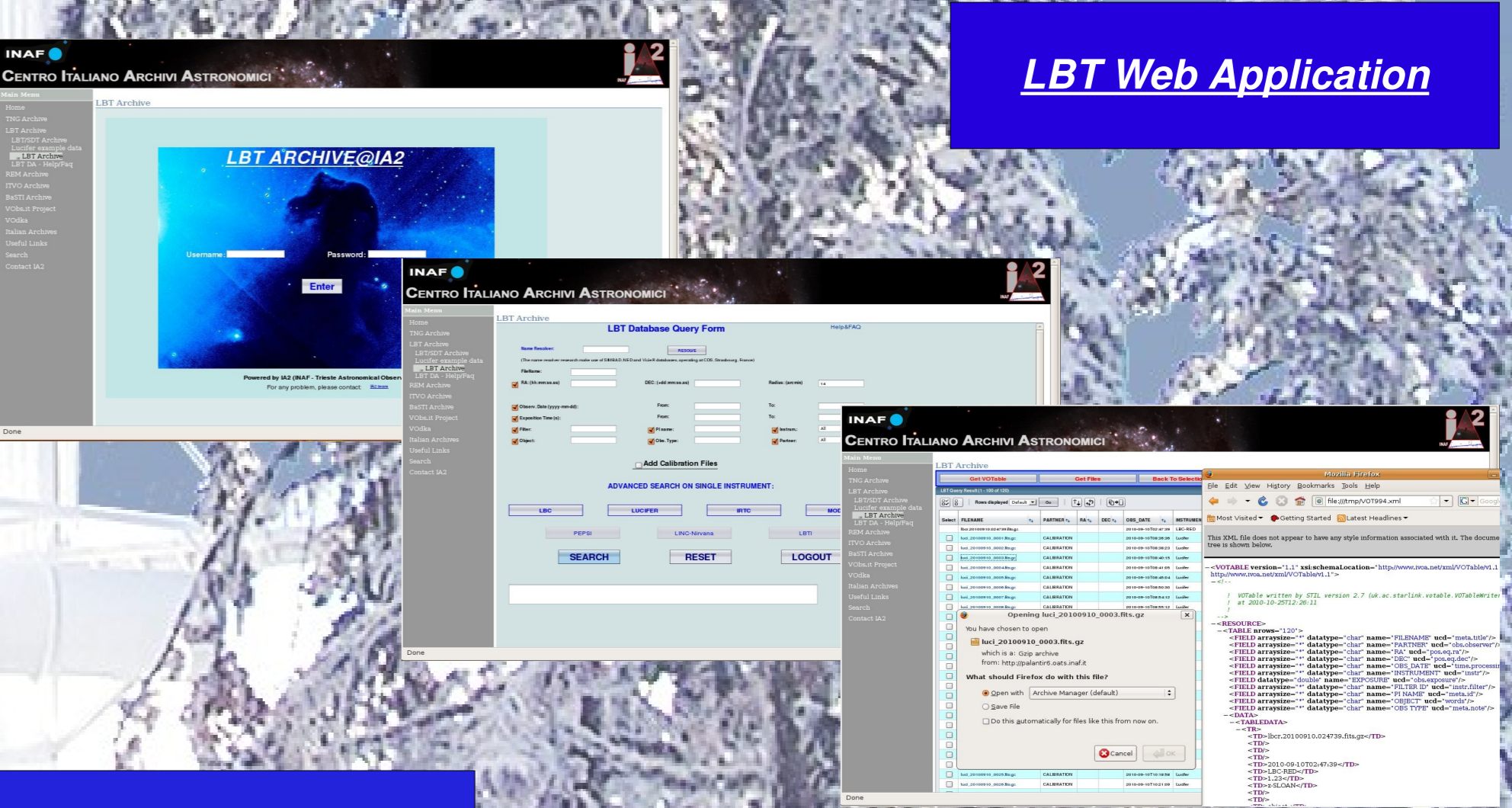
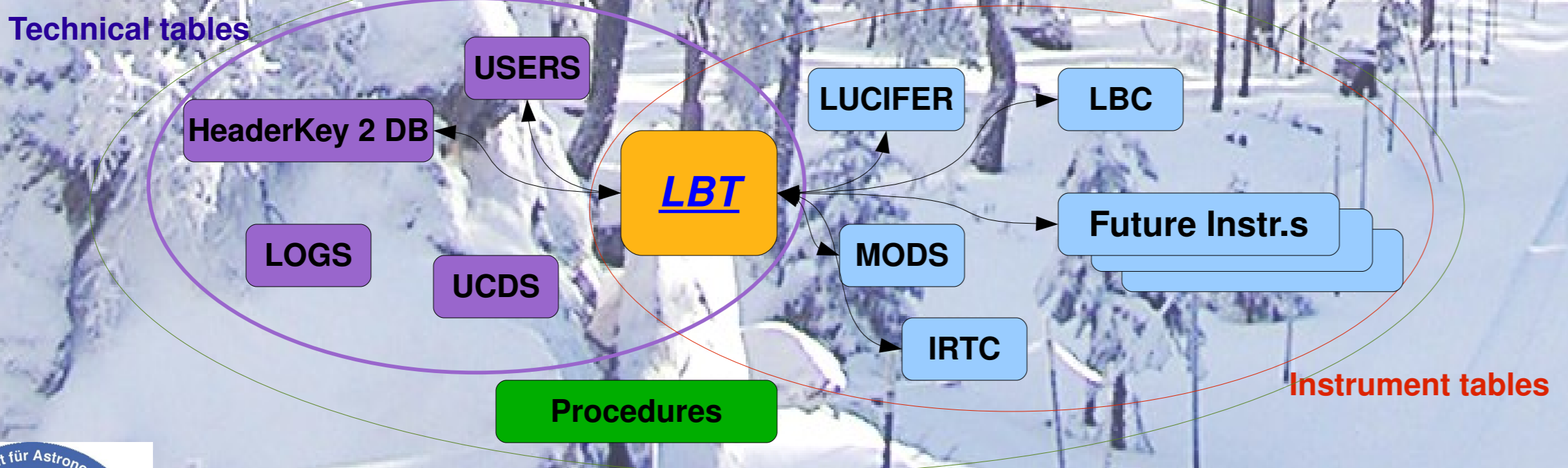
In the Archives, files are first compress and then stored, while in Repository files are stored as they are. Compression could have two different modes: gzipped (.gz) or fpacked (.fz): the selection between the two should be done at JOut start. Gzip format is the default mode.

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.

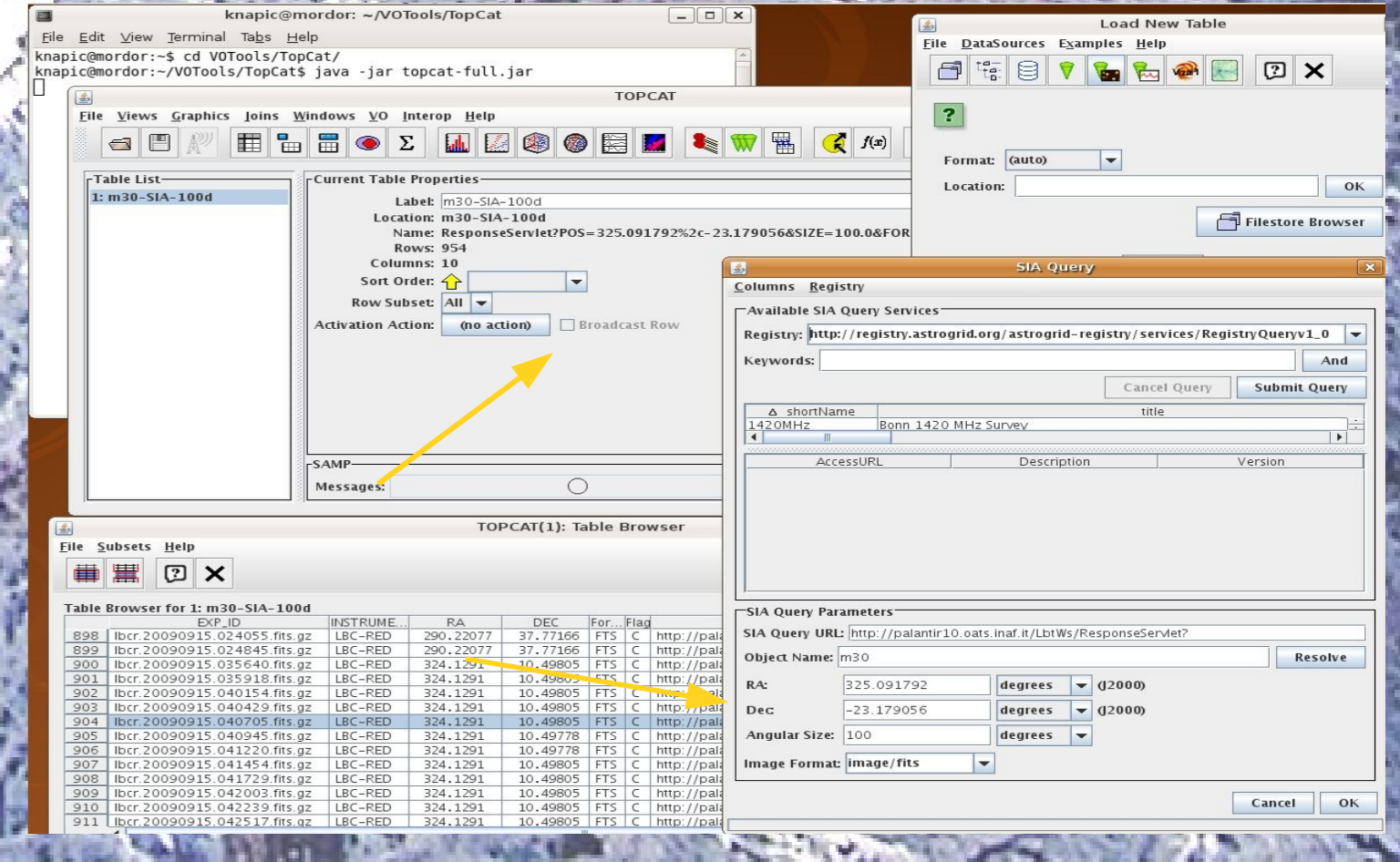
Data forwarding:

Once data are correctly inserted in database, stored in archive and all updates are performed, files are ready to 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 name in the general table. It use a blocking queue to create a list of commands to be executed and a "WorkerThread" object that perform the operations as independent 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 instantiating "n" WorkerThread objects that execute the command string, that are the first n-elements availables of the blocking queue list. Each 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 "comunication 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.

Database Schema Structure:



VO Service & TopCat



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 available to the Community and proprietary data are secured by sturdy logic and dedicated database query.
- VO compliant service is working.
- New features will be developed to make data retrieval more efficient and utilities more user friendly. 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 themselves 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.

