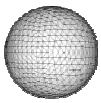
A customizable database server:
MCS, a flexible resource for astronomical projects

Luciano Nicastro, IASF-INAF, Bologna, Italy & Giorgio Calderone, IASF-INAF, Palermo, Italy

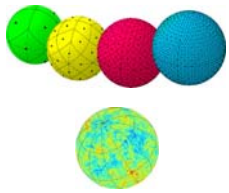


HTM: Hierarchical Triangular Mesh sky pixelization. Mainly used for objects catalogues. Good to perform cross-matching between databases. "**www.sdss.jhu.edu/htm/**".
MCS can manage automatically MySQL tables indexing using the HTM scheme. Moreover the full C++ library will be included as a class therefore making it accessible from any language.

Background

Nowadays medium-large size astronomical projects have to face the management of a large amount of information and data. Typically dedicated *data centres* manage the collection of raw and pre-processed data and consequently make them accessible to the (authorised) users. Access is performed either via (s)ftp or http (web) and typically foresees only files transfer. Also selecting the data of interest is usually performed acting on a few parameters (e.g. object name or coordinates). In a few cases, when large amounts of data are involved, no data transfer is allowed but the user can submit batch jobs that return the results of a particular analysis. In some cases the data are delivered to the user on tapes, DVDs, etc. In all cases the data acquisition, archiving, delivering, processing and the results accessibility are managed separately. Often the information are not collected into relational databases tables and when this happens, the delay between the date of collection and the archiving is of the order of days or even months. The same happens for the data production logging and project documentation. Luckily the use, in many cases, of standard file formats like FITS can help to track the data origin and processing status.

International projects like GRID and the International Virtual Observatory Alliance (IVOA) represent an effort to give a robust and standard framework to physicists and astronomers. However these projects size and ambitions cause them to proceed quite slowly and the potential users do not get immediate advantages from them. Small and medium projects/experiments tend to optimise the data management for their internal use only. Moreover only large international facilities (ground or space based) put some effort into keeping a standard data format to make easier the exchange and exploitation of the data by using standard analysis packages.



HEALPix: Hierarchical Equal Area isoLatitude Pixelization. Mainly used for sky maps in CMB experiments but can also be used for any large area survey and sources catalogue. "**healpix.jpl.nasa.gov**".
MCS will be able to manage automatically MySQL tables with this pixalization scheme. As for the HTM library, we plan to import this library to make it accessible from any language.

Why a Customizable Database in Astronomy?

Here is a short list of answers:

- To track in a consistent way what a project produces and let the rest of the world know about it.
- To manage all the information aspects of a project within a single framework.
- To forget worrying about data management but concentrate on the analysis and interpretation.
- To perform automatic real-time data processing and *on demand* tasks producing data files, tables, graphics (1, 2 ,3-d) without caring where the data are and in which format.
- To have a simple per user backup and restore management system.
- To make data accessibility uniform from any internet computer using any programming language, including the Virtual Observatory.

What do we propose?

MCS is a set of C++ high level, easy to use, classes aimed at implementing an application server, that is an application that provides a service over the network. Its main features, amongst others, are the possibility to customize the server behaviour through the derivation of the classes **LocalThread** / **UserThread**, and the usage of a well defined communication protocol (the **MCS** custom protocol). It uses **MySQL** as DB server.

With **MCS** you can easily implement custom services on top of which different users (scientists, technicians, etc.) can perform requests to a common database through several tools obtaining different types of data, depending on the tool used and the user permissions. The **MCS** protocol and software tools let user's application obtain data in a well defined binary fashion or plain text, so that they are ready to be processed further. So **MCS** and its protocol are for s/w applications what a web server and HTTP are for the WWW: a simple way to access data. In this comparison customizing the **MCS** server is like writing a web page.

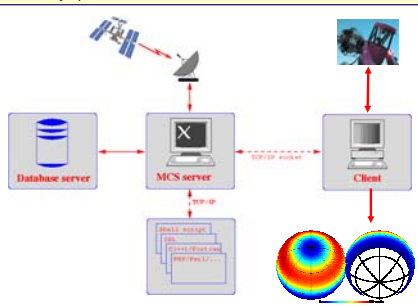
***MCS** provides a number of interfaces to different programming languages: C/C++, Fortran, IDL, PHP, Python and soon Perl and Java. By using any of these interfaces you'll get the full potentiality of **MCS** and its binary protocol. SSL encryption is supported and automatic binary byte-swapping for Little ↔ Big endianess machines is under consideration.*

The fact that the user can use any programming language (or even an interactive shell) to interact with the such server makes collaborations more profitable and the (existing or new) software easily sharable. Data input/output can be performed in several standard formats like XML, FITS and VOTable. The latter immediately makes accessible data and products to a Virtual Observatory. Noticeably communication between an **MCS** server and a VO allows a real time view and access to the Observatory products. In other words **MCS** makes the Virtual meet the Real.

*It is worth noting that the data can be either left in files (of any format) or reported in DB tables. In the latter case it could be worth implementing DB engines which allow a transparent I/O on these tables as if they where files with the original format. **MCS** foresees FITS and VOTable engines at the moment.*

*In a typical scientific experiment we have an instrument producing data, a main storage system, a set of software tools to perform analysis and people with different needs who wish to access the data. Once the data have arrived to the **MCS** server the* LocalThread *class can be used, for example, to perform data quick-look and to store the results into the DB as well. User provided s/w can be easily integrated into the server to perform dedicated tasks.* MCS_CMD_GET *will return the result.*

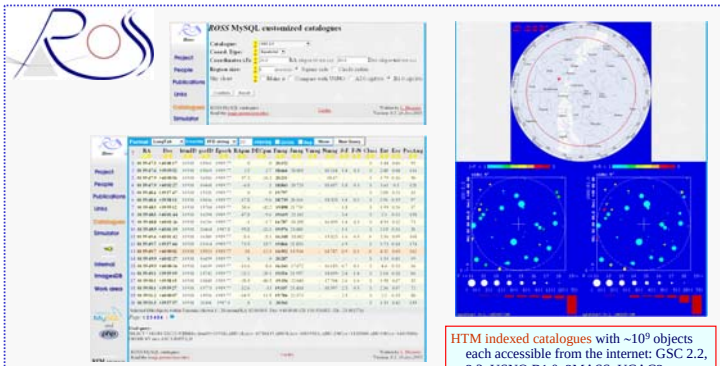
Flow diagram for an **MCS** Server/Client system. One Server accepts multiple connections from Clients on the internet performing the requested tasks.



This diagram summarises the way **MCS** works. A Server collects data coming from an Experiment (ground or space based) and saves log information and data into DB tables. A Client sends requests (using any language capable of socket connection) and retrieves info/data. Eventually it requires the Server to perform specific tasks on the data getting the results back, for example a sky map. Programs can be written in any language.

http://ross.iasfbo.inaf.it/mcs/
(nicastro@iasfbo.inaf.it | gcalderone@ifc.inaf.it)

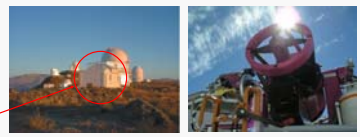
MCS is open source and it is downloadable from the Web or requested to us. Documentation is included in the distribution and it is available as doxygen HTML pages. A descriptive document (TeX+PDF) is also included. The distributed tarball provides the typical and simple configure/make/make install sequence. The only required external library is the MySQL 5 *libmysqlclient* (*libpcre* and other optional libraries are distributed with the code). A full test suite is included together with demo programs in the various supported languages. If you wish IDL/PHP/Python/Perl/Java compatibility then these packages need to be pre-installed too. The *row oriented* MySQL tables management is accomplished by **MyRO**, a Perl+SQL package we developed and distribute with **MCS**.
Please note that the library is in continuous improvement so we ask the interested people to get in contact with us. We are seeking collaborations and we plan to have a distribution mailing list.



HTM indexed catalogues with ~10^9 objects each accessible from the internet: GSC 2.2, 2.3, USNO B1.0, 2MASS, UCAC2, … "**ross.iasfbo.inaf.it/**"

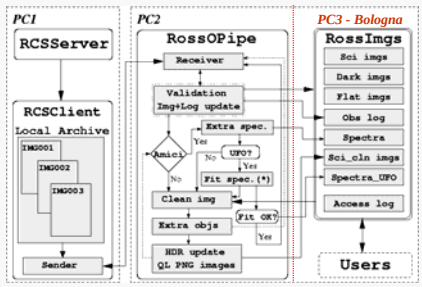
ROSS

ROSS is the **REM** (Rapid Eye Mount. robotic telescope) Slitless Spectrograph producing direct or dispersed (via an Amici prism) optical images. Observation logging and real-time image processing/archiving is performed accessing local and remote DBs. A schematic flow chart is shown below. As soon as the image is (pre)processed, it is available to the owner in the database. As usual it is accessible from any internet node. Typically the use of the dedicated web interface would be appropriate. Each user has his/her own account and can access only proprietary data. The observation log is freely accessible. As shown below, the web interface (written in PHP) allows a simple and fast access to the log and products (images and spectra). It is very easy to implement new facilities performing more tasks on images or spectra.

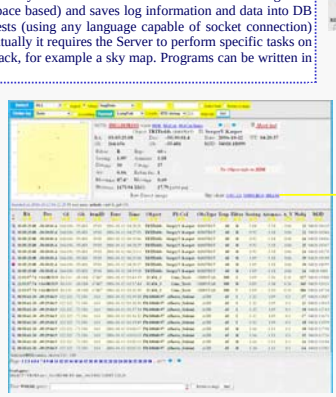


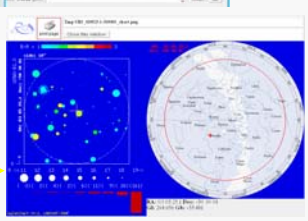
REM

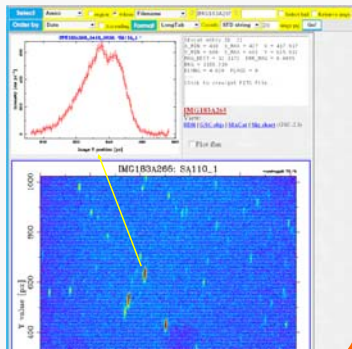
La Silla, Chile



ROSS observation log: Updated in real time and freely accessible from the internet. "**ross.iasfbo.inaf.it/**"







ROSS images archive: Updated either in real-time or during daytime. Accessible to authorised users only (e.g. the images' owner). Images selection and retrieval is very easy and does not use any extra disk space. Objects photometry / astrometry can be performed interactively via a custom version of **SExtractor**. Sky charts of the field, for various catalogues, need just one click.
Spectra: data extraction (counts / flux in FITS or ASCII format) and plotting requires again just one click.