

Joint Astronomy Centre
James Clerk Maxwell Telescope
United Kingdom Infrared Telescope
T. Jenness, F. Economou, P. Hirst, E. Archibald,
R. P. J. Tilanus

9th May 2001

JAC OBSERVATION MANAGEMENT PROJECT 3.1

The MSB Server and Database

Contents

1	Introduction	2
2	Science Program Server	2
3	MSB Database	4
4	MSB Server	5
4.1	Queries	5
4.2	“Done” flag	6
4.3	Summaries	7
4.4	Method Interface	7
4.4.1	Exceptions	7
5	Access control	8
A	Prototypes	8
A.1	JCMT Flexible Scheduling Queues	8
A.2	UKIRT Service Queue	10

Abstract

This document describes the MSB server and database design.

1 Introduction

Whereas the Observing Tool (OT) works with Science Programs (entire projects) the OMP deals solely with schedulable blocks of observations (MSBs). The OMP must therefore provide the following facilities for processing MSBs:

- Extraction of MSBs from OT science programs.
- Reconstruction of a science program from MSBs.
- Retrieval of appropriate MSBs for observing.

Rather than dealing directly with a particular implementation (e.g. direct querying of tables in a database or modifying the OT to upload individual MSBs directly to the correct location on disk) a server task is provided that will process science programs and implement queries.

Conceptually there are two servers: a science program server and an MSB server, and this document will talk about each separately. In reality it is likely that the two servers will be combined into a single server task. This reduces the number of active tasks required for observing and simplifies locking strategies (locks need to be implemented in order to prevent two tasks attempting to modify a science program simultaneously.)

The overall design of the MSB server architecture is shown in Fig. 1.

2 Science Program Server

The Science Programs generated by the OT are stored and retrieved using the interface described in OMP/SN/002. In summary, the OT creates an XML representation of the science program, compresses it and sends it to the server using CORBA. Each time a science program is stored the following steps are taken:

1. Locate the MSBs within the XML. The OT has two ways of indicating an MSB, either by using an <SpMSB> element or a lone <SpObs> element¹.
2. For each MSB extract the information relevant for scheduling. If there are multiple targets in a single MSB the average position will be used. For simplicity, initially at least, inheritance external to an MSB will not be supported. This means that it will not be possible to specify an instrument configuration at the top of a science program and apply it to each MSB.
3. Store the science program (modified to contain references to the individual MSBs rather than the MSBs themselves) and MSB XML data to disk.
4. Remove all previous entries for this project from the MSB database and replace them with the new entries. *We must make sure that it is not possible to remove the entries without replacing them*

¹The OT may in fact be made to recognise lone observations as MSBs and mark them as such when creating the XML

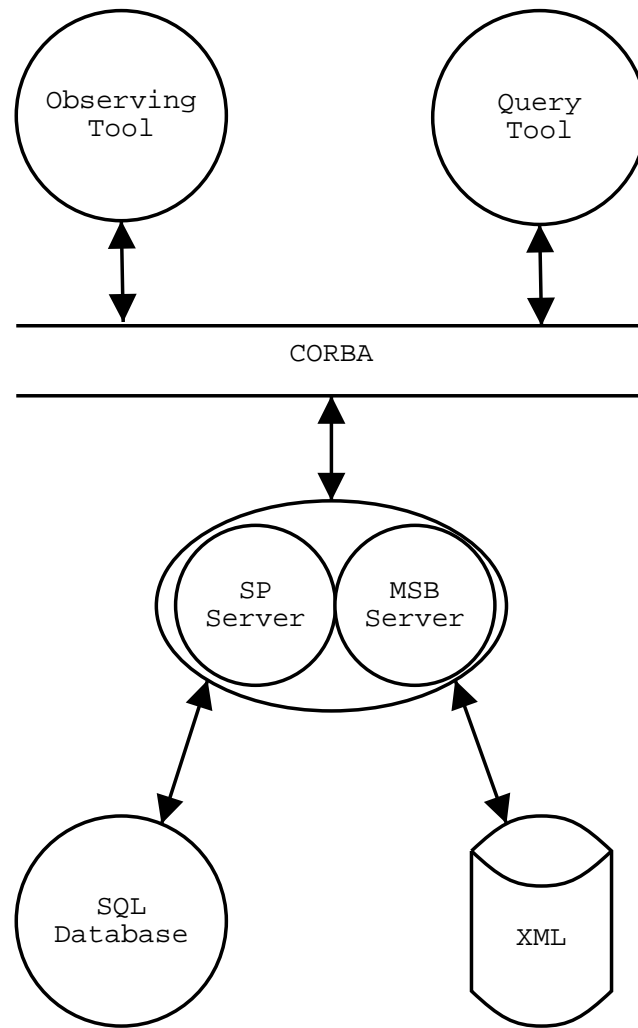


Figure 1: Block diagram of MSB server and database.

When a science program is retrieved the MSBs are simply put back into the Science program “husk” and returned to the OT.

It is important to realise that a historical record of the science program is *not* recorded. Each time the PI (or observer) submits an updated science program all entries in the database are cleared and the XML files removed. The only way to change an MSB is through the OT and an active project may be changed many times throughout its life (whether it be minor edits or major additions) and keeping track of the Science Program history is not required. All the relevant information is available from the feedback tool: a list of data files associated with the project and the current MSBs on the system. The OMP itself will not remove MSBs from a science program but will mark them as “done”. The PI can remove old MSBs if they wish to remove clutter from their programme but there is no requirement for them to do so. The PI is allowed to edit their program as often as they want and it should be remembered that the science program can be treated as a “live” document.

3 MSB Database

In the OMP the XML used to specify the MSBs by the OT is treated as the “master” source of information. There are 3 approaches to dealing with retrieving stored XML that matches certain criteria:

- Search the XML files to determine matches
- Convert the XML data to a series of database tables and query the database. Recreate the XML on demand.
- Extract information from the XML files that is most commonly used for queries and store that in the database whilst retaining the XML on disk.

The last option (the hybrid approach) has been adopted by this project. This approach has the advantage that the queries will be very fast (querying a single table using SQL is very efficient) and data extraction only has to be done when a program is submitted rather than for each query. Additionally, it bypasses the complexity of designing a table structure that stores arbitrary XML.

The schedulable information (e.g. target information, estimated time required for the MSB, instrument configuration and weather requirements) will be stored in a database table within Sybase². A unique ID will be allocated to each MSB (also used for indexing) and a reference will be included that will allow the relevant XML for the MSB to be located. This is because it is not particularly efficient to store large text blocks that will never be searched in a relational database, and because the presence of the files on disk aids troubleshooting at the telescope.

It is worth noting that whilst records of an MSB id used may be retained (eg in the data headers or the feedback tool), there is no way to guarantee that that MSB will still be available in the repository since a new science program can be submitted at any time and may not include that particular MSB (and indeed there is no guarantee that an ID will be retained through a new submission of the program even if an MSB has not been changed).

The schedulable information stored in the database will match that implemented in the query tool interface (see OMP/SN/004).

²We are using Sybase rather than Postgres or MySQL mainly because it is already in use for the data archiving system and we wish to minimize the introduction of new systems

In this system the XML stored on disk is the fundamental repository of data, while the database is being used simply to optimize searching. The server task will be used to update the database in order to guarantee that the data in the database table and the information on disk remain identical.

4 MSB Server

The MSB server is responsible for providing access to individual MSBs rather than entire science programs. The facilities provided by this server can be summarised as:

- Return MSBs that match a particular query.
- Provide a means to mark a particular MSB as “done”.
- Provide a summary of all MSBs associated with a particular project.

An interface is not provided for editing MSBs other than to modify their status. All normal editing of MSBs should be done using the OT.

4.1 Queries

The fundamental task of the MSB server is to return a list of MSBs on the basis of search criterion supplied by the user. The main interaction with the system will be via the Query Tool (see OMP/SN/004). Three methods of querying the MSB database were investigated:

- Direct SQL queries.
Simply passing SQL query strings to the server was discounted because it breaks the encapsulation on the system requiring that the client knows intimate details of the table layout.
- A single method on the MSB server for each query type.
Since we believe the number of distinct queries required for the OMP are fairly small this technique would simplify the interface to the MSB database but it would complicate the Query Tool. The QT would need to know the types of queries supported by the server rather than remaining fairly generic.
- Simple query string that is translated by the MSB server.
This is our favoured option as it allows the QT to know nothing about the MSB server queries; simply requiring that the information present in the GUI is passed to the server to be analysed.

In order to simplify the support requirements for this tool the MSB server will use an XML specification of the query derived simply from the GUI itself. The XML layout will map directly to widgets on the GUI. An example query could look like:

```
<query>
  <tau>
    <band>1</band>
```

```
<band>2</band>
</tau>
<projects>
  <project>m01au43</project>
  <project>m01au28</project>
  <project>eandc</project>
</projects>
<seeing>
  <max>1.0</max>
</seeing>
<elevation>
  <min>30</min>
</elevation>
</query>
```

The aim here is to provide the information to the server in a simple and extensible format without constraining the GUI. All the intelligence for the query (for example, mapping seeing requests to a database column) is handled by the server itself.

The range of information required to form a query is small and well-defined and there is little danger of the query “language” emulating SQL. If there are any complex or difficult to optimize queries, the server may use a database stored procedure rather than constructing the query itself.

4.2 “Done” flag

When an MSB is selected by the observer, in order to be sent for observation, there has to be a way to indicate to the OMP that the MSB should not be returned by subsequent queries.

The MSB server will provide a method that will allow a specific MSB to be marked such that it will no longer be returned in queries. This will require modification of the MSB XML and the relevant entry in the database. This mark is commonly referred to as the “done” flag.

There is, of course, a distinction between the OMP knowing that the observation is to be attempted (indicated by sending it to the telescope) and knowing that the MSB has been successfully completed (all data files are on disk). The OMP will assume that *sent for completion* indicates *observed*. This is because experience shows that faults come to light either during an observation sequence or during data analysis. In the former case, the sequence is already in the hands of the acquisition and can be repeated as soon as the problem is fixed with no recourse to OMP-related software. In the latter case corrective action will most likely involve submitting a new MSB.

Our approach does assume that the ID still refers to the original MSB. This may or may not be the case depending on whether the science program has been uploaded during the interim. To overcome this problem an MD5 checksum will be calculated for each MSB and this will be used to locate the particular MSB in the database. However many times a science program is submitted the checksum for a particular MSB will only change if the MSB itself has been modified in some way. If the MSB has been modified between its retrieval in a query and its execution then the new MSB has not been observed and can therefore not be marked “done”.

4.3 Summaries

The feedback tool requires a simple summary of the MSBs available for a particular project. This can be used by support scientists and the PI to provide an indication of the project requirements without having to examine a full science program. The information will be returned in XML format.

4.4 Method Interface

Given the above the following method interface will be available from the server:

Summary = msb_summary(Password, ProjectID, SortMethod, MSBStatus)

Return a summary of all MSBs associated with the specified project. The sort method controls the order of the returned MSBs (e.g. by weather band, by priority, by original order in Science Program) and the Status controls whether to return all MSBs, observed MSBs or MSBs that are still to be observed.

The summary will be in XML format.

Count = msb_count(Password, ProjectID, MSBStatus)

Return the number of MSBs with the specified status. If a more complex count is required (such as the number of msbs in each weather band) that can be extracted from the summary XML.

MSBs = msb_query(Password, QueryXML, SortMethod, MaxCount)

Return up to MaxCount MSBs matching the supplied query. The order of the MSBs is governed by the query itself. The MSBs will be returned as an array of gzipped XML in the order specified by SortMethod. The format of the query XML is discussed in section 4.1.

msb_done(Password, ProjectID, CheckSum)

Mark the MSB corresponding to the supplied checksum (in the specified project) as “done”.

4.4.1 Exceptions

The following exceptions can be raised by the server in an application:

UnknownProject

The requested project is not present in the MSB database.

AuthenticationFail

The supplied password is not correct.

DBConnectionFail

Unable to connect to the database backend.

MSBMalformedQuery

The Query XML could not be understood.

MSBBadSort

The specified sort algorithm is not supported.

MSBMissing

The requested MSB could not be found in the current MSB database (it may have been removed or modified using the OT).

5 Access control

It is expected that PIs will be provided with a password per project, which can be used to submit and retrieve the science programme for the project using the OT. They may share the password with their collaborators at their own discretion. The same password can be used to protect access to the feedback system on a per-project basis.

As for using the QT to query the database, a likely scenario is that there will be issued a single privileged password to allow MSB queries. In principal the password could be given out and then changed for each new observer, or the TSS could authenticate the QT on run-up without divulging the password to external observers. Since the QT will not be accessing the system from outside of the JAC, another possibility is that the password could simply be embedded in the application code. This may compromise confidentiality if the source code is obtained by non-staff but the operations team may find the risk acceptable.

A Prototypes

During development of this project prototype databases have been implemented to provide some of the benefits of the OMP (such as determining which observation to select) and also acting as a test bed for some of the ideas generated by the OMP.

Both UKIRT, in their service queue, and JCMT, in the Dutch, UK and international observing queues, are using databases to track observations but neither are linked into the observing system. For example, at JCMT the database provides the querying abilities to determine the next observation but does not provide any means of then actually performing the observation; the observer must construct the Observation Definition File themselves.

In this section we describe the table layout used in both systems. Clearly the designs are not appropriate for the OMP since they sometimes combine scheduling and project information in a single table and are slightly different for each telescope. The OMP will use the lessons learned in both systems to generate a database useful for both telescopes.

A.1 JCMT Flexible Scheduling Queues

JCMT flexibly scheduling uses 5 weather bands (described as tau bands in OMP/SN/002) and a project can be allocated to the primary queue or the fallback queue. Fallback queues are used when no targets are available in the primary queues. The table columns are shown below.

Column name	type	Description	Source
proj#	id	database index	
semester	vc6	semester in format m01a	allocations file
country	vc6	uk,cn,int,nl	allocations file
queue_type	vc8	flex,fb	allocations file
proj_id	vc16	format m01au55, nlflex, etc.	allocations file
pi_1	vc24	primary PI name	template
pi_1_email	vc40	primary PI email	template
pi_2	vc24	secondary PI name	template
pi_2_email	vc40	secondary PI email	template
object	vc16	source name - conforms to catalogue restrictions: max 16 characters, no white space '-', '+', '-', '#' are ok	template
objprior	i2	observation priority within project	template
ra	vc16	ra	template
dec	vc16	dec	template
lst_fstrt	r4	rise LST corresponding to airmass=3	calculated given ra/dec
lst_fend	r4	set LST corresponding to airmass=3	calculated given ra/dec
wband1	i4	1=in wband1 0=not in wband1	allocations file
wprior1	i2	weather band 1 priority	allocations file
wband2	i4	1=in wband2 0=not in wband2	allocations file
wprior2	i2	weather band 2 priority	allocations file
wband3	i4	1=in wband3 0=not in wband3	allocations file
wprior3	i2	weather band 3 priority	allocations file
wband4	i4	1=in wband4 0=not in wband4	allocations file
wprior4	i2	weather band 4 priority	allocations file
wband5	i4	1=in wband5 0=not in wband5	allocations file
wprior5	i2	weather band 5 priority	allocations file
time_alloc	r4	time allocated	allocations file
time_rest	r4	time remaining	allocations file initially: software after that
freq	vc40	frequency/wavelength etc.	template
mode	vc24	observing mode: photom etc.	template
comp_param	vc40	completion criterion	template
odf	vc24	odf name	template loading program allows you to choose this
links	vc8	P for proposal, F for figure	template
comments	vc240	comments	feedback software
observed	vc240	dates of observed but unreleased data. format: 010524.sh1 010302.sh2 010815.sh12 010213.day	feedback software shift 1 shift 2 both shifts day time observing
released	vc240	dates of observed released data same format as observed	feedback software

completed	i4	0: not completed 1: completed -1: marked completed by observer but not confirmed by PI or queue manager >150: marked inactive >300: effectively removed	initially set to 0. can change value using software
temp_name	vc24	template name	comes from template loading program
scuba_temp	i4	1: scuba obs 0: not a scuba obs	template
het_temp	i4	1: het obs 0: not a het obs	template
ra_int	i4	field not used:given for free by sybase	
dec_int	i4	field not used:given for free by sybase	
tuple_id	i4	field not used:given for free by sybase	

where the database is populated using an allocations file (containing general PI information and TAG allocations) and a plain text file containing details of the observations required from the PI. This system does not use MSBs and is purely target based.

Web pages are provided to edit and review this information and the main focus has been on queue management rather than an automated means to control the telescope.

A.2 UKIRT Service Queue

The UKIRT service queue is now run using a PostgreSQL database. Since this database was constructed with the OMP model in mind, it is more MSB-based than the JCMT version. The system consists of a main MSBs table containing information on the MSBs themselves, a publically readable limited view of this table, a projects table containing project specific information, and tables for refereeing and observing reports.

The layout of the main table is shown below:

Name	Type	Comment
propid	integer	Proposal ID
msbid	integer	MSB ID within this proposal
ra	float8	
dec	float8	
name	text	Target Name
formobslist	text	See note (1)
inst	text	Instrument
grafilt	text	Grating / Filters needed
otlocation	text	Location of Science Program
grade	float4	Average Referee Grade
eetime	integer	Estimated Elapsed Time
moon	text	Moon-phase requirements
moisture	text	Atmospheric Moisture requirements
photometric	text	Photometric conditions requirements

seeing	float4	Seeing requirements
datasets	text	Location of datasets so far
observed	boolean	Has it been observed (2)
onhold	boolean	Is it on hold (2)
completed	boolean	Is it complete (2)
notes	text	Notes
history	text	History

Notes from table:

1. Says which observations off the web form the MSB covers. Normally there's a 1:1 relationship between observations on the form and MSBs, though this field allows you to note that this MSB covers several observations (for example if the PI wants them scheduling together cos it's a variable source or whatever)
2. When something gets observed, the observed flag gets set, and the PI informed. If PI accepts data, the completed flag gets set, if the PI rejects the data, then we un-set the observed flag if we want to re-do it cos we did it wrong. If we attempt the observations, but there's some show-stopping problem, it gets flagged as on-hold (prevents it coming up in the queue). Typically things like they request a 3 hr observation expecting good signal-to-noise. If we have no detection at all after an hour, we'd conclude it's screwed up and stop observing until the PI comments. If the PI withdraws it, it gets marked as complete but not observed.