TCS Observation Planning Tool Software Design Document

APPROVAL SHEET

TITLE : TCS Observation Planning Tools Software Design Document

DOCUMENT NUMBER : 1732AE0008 ISSUE: B

SYNOPSIS : This document describes the software design of the Observation Planning Tools software of the TCS.

KEYWORDS : Telescope Control System, Scheduling

PREPARED BY : E. Romero Colmenero

APPROVED : Gerhard Swart
SALT Systems Engineer

Kobus Meiring
SALT Project Manager Engineer

DATE : June 2003
This issue is only valid when the above signatures are present.
Printed: 16/09/03 14:38
ACRONYMS AND ABBREVIATIONS

ATP  Acceptance Test Procedure
ATR  Acceptance Test Report
BMS  Building Management System
CDR  Critical Design Review
CIN  Code Interface Node (a LabVIEW function to interface to other SW)
ELS  Event Logger Software
EDS  Environmental Display System
HET  Hobby-Eberly Telescope
I/O  Input/Output (Device)
ICD  Interface Control Dossier
MMI  Man-Machine Interface
OPT  Operational Planning Tool
PC   Personal Computer
PDR  Preliminary Design Review
PFIS Prime Focus Imaging Spectrograph
PI   Principal Investigator (Astronomer)
PIPT Operational Planning Tool
PLC  Programmable-Logic Controller
PMAS Primary Mirror Alignment System
SA   SALT Astronomer
SALT Southern African Large Telescope
SAMMI SA Machine Interface
SC   Software Component (e.g. part of the TCSS)
SCAM Salticam (Acquisition camera)
SCL  SALT Command Language (sent to TCSS)
SDB  Science Database
SDD  Software Design Document
SDP  Software Development Plan
SI   Software Item (the TCSS is a Software Item)
SO   SALT Operator
SOMMI SO Machine Interface
SRS  Software Requirement Specification
STARCAT Object Catalogue
SW   Software
TBC  To Be Confirmed
TBD  To Be Determined
TCS  Telescope Control System
TCSS Telescope Control System
TPM  Telescope Pointing Machine (software for Astrometric Pointing)
VI   Virtual Instrument (LabVIEW function)
WCS  World Coordinate System
WEB  SALT web-server
## DEFINITIONS

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposal</td>
<td>Requirements for a full scientific project</td>
</tr>
<tr>
<td>Target</td>
<td>A particular astronomical object (e.g., star, galaxy) to be observed</td>
</tr>
<tr>
<td>Group</td>
<td>A set of observations that must be performed in a specific order</td>
</tr>
<tr>
<td>Script</td>
<td>An instrument-readable set of commands that define the configuration and observational sequence for an observation.</td>
</tr>
<tr>
<td>Observation</td>
<td>In general, a pointing to a target with a particular instrument configuration. Please note, however, that there may be several consecutive exposures of a target with minor differences in instrument configuration (all contained in a single instrument script).</td>
</tr>
<tr>
<td>Exposure</td>
<td>A single CCD readout, resulting in a single data file. An observation will probably consist of several exposures, e.g., an arc (calibration) followed by a science observation etc.</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

1 Scope ................................................................................................................................. 7
2 Referenced Documents .......................................................................................................... 7
3 Software Context ..................................................................................................................... 7
  3.1 Software external interfaces .............................................................................................. 8
    3.1.1 SDB .......................................................................................................................... 8
    3.1.2 SA ........................................................................................................................... 8
    3.1.3 SAMMI ...................................................................................................................... 8
  3.2 Development environment ................................................................................................. 9
  3.3 Hardware platform ............................................................................................................ 9
4 OPT high-level design ........................................................................................................... 9
  4.1 Architecture ....................................................................................................................... 9
    4.1.1 SDB Interaction .......................................................................................................... 9
    4.1.2 SA Interaction (via GUI) ............................................................................................ 9
    4.1.3 Filter ......................................................................................................................... 10
    4.1.4 Scheduling/optimisation algorithm ............................................................................. 10
    4.1.5 Pointing model ......................................................................................................... 10
    4.1.6 Error handling .......................................................................................................... 10
  4.2 High-level functional flow ................................................................................................ 10
  4.3 Data flow .......................................................................................................................... 11
  4.4 The OPT growth path ....................................................................................................... 12
    4.4.1 Basic ....................................................................................................................... 12
    4.4.2 Intermediate ............................................................................................................. 12
    4.4.3 Advanced ................................................................................................................ 12
    4.4.4 Sci-Fi ....................................................................................................................... 13
5 OPT detailed design .............................................................................................................. 13
  5.1 OPT functional flow diagram and description ................................................................ 13
  5.2 OPT architecture .............................................................................................................. 15
    5.2.1 SDB Interaction ........................................................................................................ 15
    5.2.2 SA Interaction (via GUI) ........................................................................................... 15
    5.2.3 Filter ......................................................................................................................... 16
    5.2.4 Scheduling/optimisation algorithm ........................................................................... 16
    5.2.5 Pointing model ......................................................................................................... 16
    5.2.6 Error handling .......................................................................................................... 16
  5.3 The GUI design .................................................................................................................. 16
    5.3.1 The tables ................................................................................................................ 16
      5.3.1.1 The top table ...................................................................................................... 17
      5.3.1.2 The bottom table .............................................................................................. 17
    5.3.2 The DISPLAY EXTRA pull-down menu ................................................................... 17
    5.3.3 The SELECT NEXT button ..................................................................................... 17
    5.3.4 The REMOVE NEXT button .................................................................................... 17
    5.3.5 The SELECT FIELDS pull-down menu ................................................................... 17
    5.3.6 The UPDATE button ............................................................................................... 17
    5.3.7 The FILTER button ............................................................................................... 17
    5.3.8 The OPTIMIZE button ........................................................................................... 18
    5.3.9 The SCHEDULING PERIOD button ......................................................................... 18
    5.3.10 The SAVE SCHEDULE button .............................................................................. 18
5.3.11 The LOAD SCHEDULE button
5.3.12 The EXIT button
1 Scope

This document defines the design for the Observation Planning Tool (OPT) software of the SALT Telescope Control System (TCS). The requirements for this software are defined in the OPT Software Requirements Specification, document number 1732AS0002.

The purpose of the OPT is to aid the SA in producing an optimal schedule of observations. The OPT will display all possible observations in the time interval required, it will sort through various weighting algorithms and produce a list of suggested targets with their suggested observation order for that period.

The major factors that have influenced the design of the software are the following:
- The OPT will have a GUI in which a lot of information must be displayed for the SA to make his/her scheduling decisions. Thus the GUI requires careful design.
- The OPT will display information from the SDB, the design of which will change with time. Therefore, the OPT must also be flexible enough to cope with these changes in a transparent manner, with the minimal amount of code adjustment.
- The OPT will follow a growth path and its weighting algorithms will have to be adjusted regularly. The OPT must therefore be designed to allow easy algorithm modification without requiring major code redesigns.
- The OPT development must lead towards total independence from SA interaction. This would lead to maximum SA and telescope efficiency.
- The OPT is an important aid for the SA, but it is not critical for telescope operations. Therefore, its development schedule is more flexible than other software items of the TCS.

The functional flow and design of the software is shown primarily using diagrams together with some explanatory text.

2 Referenced Documents

The following documents are referenced in this design document and are applicable to the extent mentioned herein.

1000AB0044  SALT LabVIEW Coding Standard
1000AS0040  SALT Operational Requirements
1700BP0009  TCS Software Development Plan
1700AS0001  TCS Specification
1741AS0001  TCSS Software Requirements Specification
1000AS0049  SALT Data Interface Control Dossier
1741AA0002  Astrometric Pointing Model
1741AA0003  Geometric Pointing Model
1000AA0058  Geometric Model and Error Corrections
1732AE0002  TCS Observation Planning Tool Software Specification
1732AE0003  TCS Science Database Software Specification
1732AE0009  TCS Science Database Software Design Document

3 Software Context

Figure 1 displays a graphic representation (that Gerhard likes so much – do I get any brownie points?) the software context of the OPT. The SOMMI, SAMMI and the TCS are shown in the diagram because, although the OPT does not interact directly with any of them, its function is highly dependant on all three inserting essential information into the SDB which the OPT will query for.
3.1 Software external interfaces.

Figure 1 shows the software external interfaces as specified also in the OPT Software Specifications, of which this is a summary. Please note that, although the OPT will be displayed on the SAMMI, both programs will run from different computers (True?) and will not interface directly with each other.

3.1.1 SDB
The OPT will query the SDB to obtain:
- All (filtered) target details available for schedule
- Which observation is currently being carried out
- Seeing/sky brightness/transparency information (from the previous/current track) in order to compare with observation requirements of each target still to schedule.
- Current telescope position (from current track) in order to estimate acquisition times for the next target

The OPT will insert into the SDB:
- The scheduling order
- Once a (series of grouped) observation(s) has been completed, the next time the OPT is requested to update the schedule it will first update the relevant entry(s) in the SDB to indicate that the group is now available for re-schedule, for example in case of multiple visits per night or if the data is rejected.

3.1.2 SA
The OPT will require input from the SA regarding:
- The time range to schedule for
- Expected seeing/moon/transparency conditions (if scheduling in advance or at the beginning of the night) in order to filter out unfeasible observations
- Initial telescope position (if important when scheduling ahead or at the beginning of the observation). Otherwise it will assume a standard telescope “parked” position.
- Filtering information (eg. instruments down for repairs, PFIS filters available for the night, priorities, etc).
- In the OPT basic and advanced levels, to make up the actual schedule.
- Confirm schedule (at least until the OPT reaches level 4 of development (Sci-Fi level)).

3.1.3 SAMMI
The OPT GUI will be integrated into the SAMMI display through an active-X window.

### 3.2 Development environment

The OPT will be developed in Java. The object-oriented environment and the way Java interacts with the SDB make it a good choice of programming language. *(Not if the SAMMI must run it locally because it will be memory intensive...)*. Its one disadvantage is the more involved GUI design and implementation.

### 3.3 Hardware platform

The OPT will reside in the Data Processor computer, which will run under Linux (?) It will therefore share its hardware with the SDB. As the selection of the hardware is not critical for this application, the hardware platform selection will be specified in the SDB Software Design Document.

## 4 OPT high-level design

### 4.1 Architecture

![OPT high-level architecture diagram](image)

**Figure 2: OPT High level architecture diagram**

Figure 2 Shows a high-level diagram of the architecture of the OPT. The OPT is divided into five modules:

**4.1.1 SDB Interaction**

This module deals with all database interactions.

**4.1.2 SA Interaction (via GUI)**
This module refers to the GUI as being the main tool of communication between the SA and the SDB contents (including the current schedule). Therefore the GUI must be able to display only the required information and allow the SA to set filtering parameters and modify the schedule as required.

4.1.3 Filter
This module deals with the data filtering selection.

4.1.4 Scheduling/Optimisation algorithm
This module will be the “heart” of the OPT: the algorithm that will automatically choose targets and schedule them for observation, with emphasis in optimisation of telescope efficiency.

4.1.5 Pointing model
The pointing model will be extensively used in order to calculate track times and durations for each of the short-listed candidates in order to ensure optimal scheduling. It will be a ‘scaled down’ version of the TCS pointing model, as the level of accuracy required for the OPT is not as high as that for the TCS.

4.1.6 Error handling
Error handling, although indicated here as a single module, will be implemented throughout the software.

4.2 High-level functional flow
Figure 3 shows a high level functional flow for the OPT.
Figure 3: High Level OPT functional flow

The OPT starts by prompting the SA to input the scheduling range. On receipt, it obtains from the SDB the list of viable observations for the required time range and launches a full GUI in which to display them. It will then wait for further user input:

- It will further filter the observations, which will prompt further SDB queries and re-displays to the GUI.
- It will be requested to either schedule the observations or to modify the schedule – this will also cause the program to contact the SDB and re-display its contents.
- It will be required to modify the data to display. Since java makes a full copy of all the return database query contents, only a subset of the database contents will be queried by default. Therefore, the result of such a request will depend on whether the OPT has all of the information it is now requested to display already in memory:
  - If so, it will be a fast internal process.
  - If it does not, the OPT will query the SDB for the required information.
- The OPT will exit on request.

4.3 Data flow
Figure 4: OPT High level data flow

Figure 4 shows a high-level data flow for the OPT. The OPT’s main function is to schedule the data contained in the SDB in the most efficient manner. The SA can input any filtering information required in order to narrow down the number of possible targets for the particular time range. This filtering information will be turned into a database query. The SDB will return the filtered dataset, which will then be displayed. The SA can also modify the schedule eg, by asking the OPT to optimise the shortlists, by adding/removing targets or by loading a previous schedule. This schedule information will be stored in the SDB and it will also be displayed in the OPT.

4.4 The OPT growth path

It is envisaged that the OPT functionality will follow a growth path. Here are the suggested stages:

4.4.1 Basic
At this stage the OPT will be essentially a “dumb” display tool of the proposals contained in the SDB:
- It will be able to select what objects are visible in the requested period
- It will be able to show which observations are grouped together (i.e. must be observed in sequence)
- It will be able to apply some limited filtering of the proposals (eg. based on seeing conditions)
- It will display, on request, some limited extra information that cannot be shown in tabular form (eg. scientific justification, finding chart).
- It will interact with the SDB to mark objects selected by the SA for schedule.

4.4.2 Intermediate
At this stage the rudiments of the optimisation algorithm will be incorporated into the program. The OPT will be able to:
- Make its own suggestions as to which objects are scheduled next
- Display a “schedule efficiency” factor to aid in the scheduling process
- More filtering will be available to the SA
- More non-tabular information about the proposal will be displayed on request
- It will still depend on the SA for selection/ordering of targets

4.4.3 Advanced
The OPT optimisation algorithm should be fairly well developed by now. Therefore, the OPT will be able to:
- Make up its own schedule every time it runs/updates (but still allowing for SA input if
necessary)

- If an SA were to select an alternative target, the OPT will schedule it in its optimal location, given the current schedule, and remove all other clashing targets from the schedule.

4.4.4 Sci-Fi

The OPT optimisation algorithm is so good that it can run on its own, independent of the SA (but retaining all of its previous functionality and subject to SA interaction if required).

5 OPT detailed design

5.1 OPT functional flow diagram and description

Figure 5 shows the functional flow of the OPT with all the proper symbols. I hope.
The OPT will work as follows. The SDB contains a list of all outstanding observations. The OPT will load its GUI initially with no data.
1. The program will begin by prompting the SA for the required time range for the schedule.
2. The schedule time range will be converted into a range of visible RA, Dec for the requested period, which will be used for querying the SDB.
3. The maximum track length of each selected source will then be calculated. This will be compared to their requested track length in order to filter out impossible observations for the night.
4. This base shortlist will not change unless the scheduling period changes, thus it will be stored as a temporary table in the diagram (called shortlist1 in the diagram) for OPT/SAMMI/SOMMI use.
5. This shortlist will be copied “as is” into a second shortlist (named shortlist2), which will be displayed in the GUI.
6. The OPT will now wait for SA input.
7. The SA now has 10 courses of action:
   a. Load a schedule already worked out – this will update the SDB and shortlists and the results will be displayed on the GUI.
   b. He/she can further filter the data (for eg seeing conditions) and shortlist2 will change to reflect the (now smaller) shortlist2. The GUI will now display the updated version of shortlist2.
   c. If required, the shortlist can then go through the schedule optimisation routine– this process will result in the shortlist being “sorted” into some preliminary schedule, the SDB and the two shortlists will be updated to reflect the schedule and the changes will be displayed in the GUI.
   d. The SA can also remove filter constraints (eg. if the weather conditions improve during the night), in which case the filtering will occur from shortlist1 and again stored into shortlist2 and re-displayed.
   e. To aid the SA in his/her scheduling process, the SA can also select which fields he/she would like displayed on the tables. In the interest of memory saving, not all the SDB contents will be stored in memory. Thus if the field is in memory, it will be displayed/removed. If the extra field requested is not in memory, the shortlist2 will be queried in order to provide the information.
   f. The SA can also display information that cannot be displayed in tabular form about a particular proposal, such as the configuration scripts, the abstract, finding chart, scientific justification... This information will be obtained from shortlist2 and will be displayed as a popup display.
   g. He/she can manually select a source from the GUI’s top list to schedule next, or from the bottom GUI to be removed from the schedule. This information will be updated in the SDB as well as in both shortlists, and the GUI will update its display to reflect the changes (and show repercussions).
   h. The SA can save the schedule at any time. This will create the SDB entry required for storage.
   i. Change the time range for schedule. This will reset the OPT to step 2.
   j. The SA can exit the application.

5.2 OPT architecture
The OPT is divided into six modules. Here is a basic description of what each module does.

5.2.1 SDB Interaction
This module deals with all database interactions.
   o Input: the filtering information
   o Process: generate a query and relay it to the SDB.
   o Output: a copy in memory of the SDB “reply” that the rest of the programs can interact with.

5.2.2 SA Interaction (via GUI)
This module refers to the GUI as being the main tool of communication between the SA and the SDB contents (including the current schedule).
   o Inputs:
From the SA via selection buttons
From the Filter module, the details required for a particular filter.
Memory copy of the SDB query results

- **Process**
  - It will wait for SA input.
  - On input receipt, it will relay it to the relevant module
- **Output**: it will further query the PI and wait for input, or it will then display the memory copy of the results in the required format.

### 5.2.3 Filter

This module deals with the data filtering selection.

- **Inputs**: The name of the filtering process (e.g., seeing) and its range.
- **Process**: If the range is within that contained in memory, it will filter the data and it will then return the filtered data.
  - If the range is not within that contained in memory, it will relay the information to the SDB Communications module in order to update the memory copy with the required information.
- **Output**: It will modify the memory copy (and thus the display) according to the filtering requirements.

### 5.2.4 Scheduling/Optimisation algorithm

This module will do the actual scheduling optimisation

- **Inputs**: memory copy of the SDB
- **Process**: optimise the schedule. This process will be described in detail later on.
- **Output**: the order of the selected observations, which will be relayed to the SDB via the SDB Communications module.

### 5.2.5 Pointing model

The pointing model will be extensively used in order to calculate track times and durations for each of the short-listed candidates in order to ensure optimal scheduling.

- **Inputs**: Either RA, Dec or time range
- **Process**: calculate from a given RA, Dec the observable time range (i.e. when an object will become observable and the maximum track time). Given the time range, the range of RA, Decs that will be observable by SALT during that period.
- **Output**: the result of the calculations.

### 5.2.6 Error handling

Error handling, although indicated here as a single module, will be implemented throughout the software.

- **Inputs**: all errors
- **Process**: evaluate the error.
- **Output**: act accordingly.

### 5.3 The GUI design

Screenshot of GUI will go here (distributed at CDR).

The GUI is currently envisaged to contain two tables for observation information display, and a series of buttons and pop-down menus. This is a summary of its expected functionality. Please note that, as mentioned above, not all of the functionality will be required from the start!

#### 5.3.1 The tables

The tables are responsible for displaying the relevant observation details. They will indicate observations that are grouped (i.e., must be observed in sequence) by displaying them together and highlighting them in colour and in italic font. A different colour will be used for a different group.

Time-critical observations (and their grouping) will be highlighted in bold.
5.3.1.1 THE TOP TABLE
The top table contains the shortlist of available observations (except for those that have either been filtered out (see filtering section) or selected as next.
To select an observation to be scheduled next, select the relevant row and click the NEXT button. Initially, if there are already targets in the schedule, this observation will be scheduled last (at least until the OPT becomes intelligent enough to insert the observation in the best position within the current scheduled targets.
If an observation is selected, the rest of the observations that would clash with this one will be highlighted in red until the observation is selected for schedule (using the SELECT NEXT button – see below).

5.3.1.2 THE BOTTOM TABLE
The bottom table contains the scheduled targets for the relevant period in their current scheduled order.
To remove an observation from this list, select the relevant row and click the REMOVE button.
To alter the order of the observation schedule, drag the relevant row to the required position on this table.

5.3.2 The DISPLAY EXTRA pull-down menu
By selecting an observation and clicking on this button, the OPT will display the selected information in a separate, pop-up display. The pull-down menu will contain the type of information available for display, eg. scientific justification, finding chart...

5.3.3 The SELECT NEXT button
This button can be used to add an observation to the schedule, if there is a selected row in the top table.

5.3.4 The REMOVE NEXT button
This button can be used to remove an observation from the schedule, if there is a selected row in the bottom table.

5.3.5 The SELECT FIELDS pull-down menu
This button will contain a list of fields that can be displayed on the tables. It is impossible to display all of the proposal information all of the time, thus the button provides some level of customisation. (Growth path: SAs can store their own individual field settings).
The field will contain an entry called TRACKS: if this is selected, a timeline of the track times and durations of each observation will be displayed instead of the regular information fields (rather similar to a project schedule diagram).

5.3.6 The UPDATE button
Clicking on the update table will update the alt-az of all of the displayed targets. This will be useful during the night, as observations get completed and the observing windows change all the time. It is NOT envisaged that the OPT will update automatically, as this would interfere with regular telescope operations.

5.3.7 The FILTER button
Clicking on this button will pop-up a list of available filtering options. When selected, the filtering will be active. Some will require input from the SA (eg seeing conditions).
The list of fields so far:
  o Instrument configuration (for those configurations which are currently unavailable)
  o Seeing requirements
  o Sky brightness requirement
  o Sky transparency requirement
  o Priority (eg. too many targets in this region of the sky, so ignore priorities S2 and S3)
  o Proposal_id (proposal suddenly on hold?)
Scheduling period

5.3.8 The OPTIMIZE button
The future heart of the program! This will cause the OPT to schedule the current shortlist in an optimal way. These results will initially be indicated to the SA by reversing the colour table of the relevant observations.

5.3.9 The SCHEDULING PERIOD button
This will allow the SA to change the time range of the scheduling period under consideration.

5.3.10 The SAVE SCHEDULE button
The SA will be able to save a schedule, eg. if he/she was working out tomorrow’s schedule. This schedule will be stored as a table in the SDB, which will only contain the name of the schedule, the ids of the observations in the schedule and their ordering.

5.3.11 The LOAD SCHEDULE button
The OPT will query the SA for the name of the schedule to be displayed on the lower table. The SDB will be modified accordingly.

5.3.12 The EXIT button
The OPT is envisaged to work throughout the night. Therefore, on exit, the OPT will prompt the SA whether to save the current schedule before exit. It will then proceed to reset the SDB OPT ordering fields, remove the temporary shortlist tables and exit.