System Requirements Specification

for

A CONTROL AND MONITORING SYSTEM
FOR IUCAA and NCRA

Version No.	2.0
Date	11th June 2009
Authors	Milind Dere, Chaitral Purkar
Reviewer / Approver	Vivek Mohile

Document Control:

Document History:

Date	Version	Author	Comments
28/11/2008	0.1	Milind Dere Chaitral Purkar	Initial Draft without protocol/command set.
25/12/2008	0.2	Milind Dere Chaitral Purkar	Design Document with protocol/command set.
02/02/2009	1.0	Milind Dere Chaitral Purkar	Expanding Normal Operation state and GUI finalization for generic CMS.
	1.1	Milind Dere Chaitral Purkar	Added points following review presentation
18/04/2009	1.3	Vivek Mohile	Review and readability changes
11/06/2009	2.0	Vivek Mohile	Final release based on NCRA/ IUCAA review. Implementation plan shifted to separate document

Distribution List:

Name	Description
-	IUCAA – Scientists and Engineers involved in the CMS project
-	NCRA / GMRT Scientists and Engineers involved in the CMS project

Contents

1 Introduction

1.1 Project background

This document describes the high level specifications and requirements for a new Control and Monitoring System (CMS) for Optical and Radio telescopes. The Inter University Centre for Astronomy and Astrophysics (IUCAA) and the National Centre for Radio Astronomy (NCRA) are seeking to develop a new CMS encompassing at a high level control, data acquisition, and monitoring and user interface functionality respectively for their 2m optical telescope at Girawali and 15m radio telescope at NCRA campus, Pune University.

Given the complexity of the CMS after initial discussions it was decided to split the CMS project into two phases: a “Specifications / Requirements Analysis phase” and an “Execution phase”. Persistent Systems Limited (PSL) has developed the specifications after studying the to-be user requirements, existing system features and engaging in detailed interviews with the stake holders at these two institutions. The specification can form the basis for competitive bidding for the execution phase by organizations with the relevant experience.

It is also noted that while there necessarily have to be many references to the underlying telescope hardware and sub-systems in the specification, this project does not include a study or specification development for these hardware or drivers for the hardware. As part of the responsibilities for the overall system upgradation it was decided that IUCAA / NCRA together with its vendors would take responsibility for the sub-systems closer to the hardware that do not lie within the CMS scope.

1.2 Overview of the methodology, information sources and references for the project

For the purpose of developing the specification, the following methodology and information sources were largely used:

- Understanding the features of the existing system and its limitations; achieved through
    - Several site visits and demonstrations, interviews and interactions with the system stakeholders (scientists, engineers, operators)
    - Study of manuals and related documentation

- Development by Persistent of draft ideas and strawmen for architecture / UI etc. that were validated and refined with inputs from IUCAA / NCRA
    - During interim review workshops with joint participation by a majority of stakeholders
    - During one on one discussion with individuals
- Detailed inputs regarding module level specifications and data field requirements were derived by drawing up a data template that was filled out by the engineers owning respective systems.
- Highlights of the specifications to be proposed were presented for review and review comments incorporated into this document.

The study was carried out approximately between September 2008 and February 2009. Specification documents were prepared and reviewed subsequently.

The following are some of the published documents and information sources referenced during the study:

- Telescope Technologies Limited 2.0m TELESCOPE, Operators Manual (IUCAA).

- Teleset Software of GMRT Telemetry System.
- ABCcom Software of GMRT Telemetry System.
- Chapter 20-The GMRT Servo System, Low Frequency Radio Astronomy, 3rd edition.

1.3 Definitions, Acronyms and Abbreviations

XML- Extensible Markup Language.

DTD - Document Type Definition.

GUI - Graphical User Interface.

CMS - Control and Monitoring System.

DOM – Document Object Model

PCB - Printed Circuit Board.

RDBMS – Relational Database Management System

Java EE – Java Enterprise Edition

IUCAA - Inter University Centre for Astronomy and Astrophysics.

IGO – IUCAA Girawali Observatory

NCRA - National Centre for Radio Astrophysics.

GMRT - Giant Meter Wavelength Radio Telescope

Wrapper – A software component or program which translates the messages or data formats received from another program into a standard XML format chosen for the CMS implementation

Altitude and Azimuth (Alt-Az) – Represents the position of any celestial object, or the telescope orientation with respect to the earth co-ordinate system, and changes with location on earth and time of day

Right Ascension, Declination (Ra, Dec) – Represents the position of any celestial object with respect to the inertial celestial sphere co-ordinate system and for faraway objects, will be the same for an observer from any location on earth and time of day, although there is some change in these co-ordinates over several years

Servo system – System responsible for accurately moving the telescope or any of its components along one or more of its degrees of freedom. Includes the servo controller, servo drive, servo motor, mechanical gear train and feedback components such as encoders.

2 Summary of the new Control and Monitor System objectives; Specification overview

2.1 High level objectives

This project is a part of a joint effort by IUCAA and NCRA to develop a modern, state of the art control and monitoring system for their 2m optical and 15m radio telescope respectively that significantly improves upon the way the users interact with the telescopes' systems.

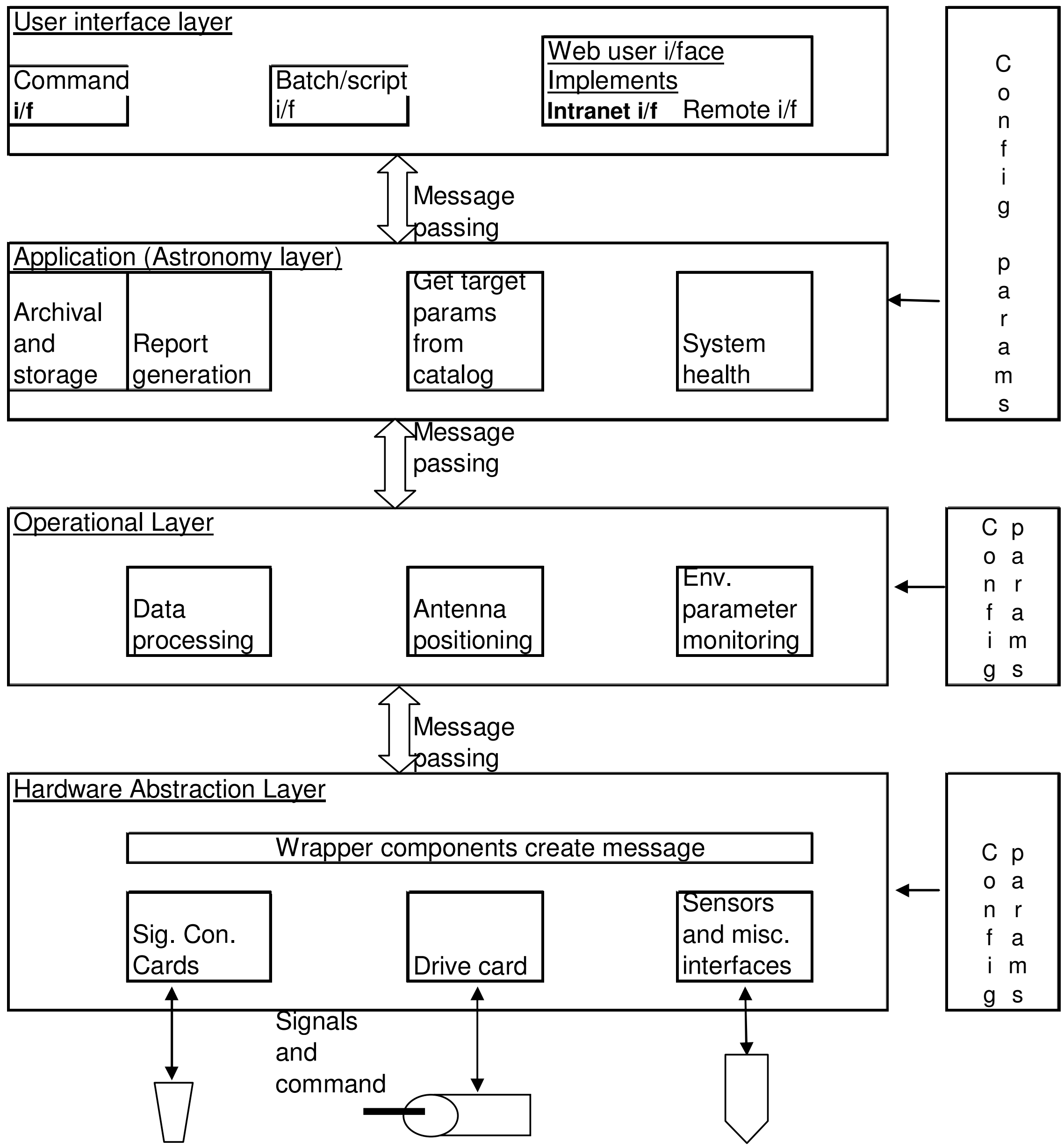
While there is considerable commonality, especially from the perspective of use cases, in the high level functionality of optical and radio telescopes, this is probably the first time that joint development of a control system for an optical and radio telescope is being attempted. A major requirement thus is ensuring that the architecture and specification is sufficiently generic to cater to both types of telescopes with most of the differences being captured as configuration parameters.

As part of the development of a new CMS, some of the modern software development paradigms that should be applied to the system are:

- As the CMS incorporates the GUI with which the end-users interact most frequently, the design of the same should incorporate modern human factors design / UI ergonomics principles
- Architecture should be modular and allow easier upgrade of individual hardware or software components with minimal impact to other systems.
- Message passing / service oriented architecture approach to the system
- Context based UI to make the system simple to use and providing each user only the exact information he needs based on his role

2.2 User expectations overview

As described above a key user expectation from the new CMS is that the architecture should be modular, with loose coupling between the components, and allow for separate storage of most of the system parameters and configurations, outside the CMS code base. This objective is depicted in the conceptual architecture representation below.



A description and few typical illustrations of some functionality to be implemented / information handling that would occur in the layers starting from the lowest is described here for clarity.

i) **Hardware and mechanicals:** This layer consists of the drives / actuators and sensors and the mechanical elements they interface with. Although there is no CMS software resident in this layer, yet it is described here as they form a part of the overall control and data acquisition scheme and the hardware abstraction layer interfaces to the same.

ii) **Hardware abstraction layer:** This is the software / firmware layer which is closest to the actual control and data acquisition hardware. At the one hand its function is to acquire the raw data from the sensor and do filtering / conversion, provides the command signals for driving the servo drive, reading the hardware error codes etc, and on the other hand to perform the interface from such raw data / command signals into an abstracted representation of messages to be exchanged with the layer above.

For example, a message from the layer above to focus the antenna on a certain azimuth, elevation would need to be converted by the hardware abstraction layer into actual command sequences for the servo drive. Similarly a diagnostic error code from the servo drive which could possibly be in a hexadecimal format would be translated into an appropriate message.

Wrapper components: Since many of the hardware drivers have been developed **over** time and in many cases by third parties, rewriting all of them may not be possible. Therefore a set of wrapper components would need to be created to provide the translation feature between the upper layers and the hardware drivers.

NOTE: Although the logical representation of the wrappers is shown in the Hardware Abstraction Layer, physically they could reside in the CMS application server as well. This depends on the specifics of each hardware implementation.

iii) **Operational layer:** This layer deals with the control and acquisition data and messages at a level abstracted from the hardware cards, drives etc. but which is relevant to the characteristics of the equipment. For example it would deal with physical elevation, azimuth angles and not be concerned with encoder counts. The configuration data at this layer would include the limits of the azimuth and elevation to which the equipment may be moved in either the operational or maintenance mode and ensure these are respected. The system state machine should also be implemented through modules and persistent data at this layer.

iv) **Application (astronomy layer):** This represents the events and data at an abstract level and the objects herein would be at the astronomy domain terminology level.

v) **User Interface layer:** This layer provides the User Interface to the actual user, including GUI and command line

The table below provides a high level view of the important expectations from the system expressed by the user to meet their needs and overcome the limitations of the current system. In that sense they define the overall objective and the first level of features that need to be provided by the CMS. Against these the brief description of the CMS specification that can be derived from these needs is listed.

S. No.	End user expectation / feature	Brief description of specification to achieve the same
1	New CMS should largely be based on Open Source platforms and frameworks and portable across client variants	Recommended development platform is Java EE and interface based on XML with wrappers developed in C or C++ at lower level using GNU tool chain.
2	CMS System should allow change of hardware with less impact at higher layers	Architecture for CMS should be well layered and abstraction oriented clearly separating application / middleware and hardware interface layers. Configuration and data interface to hardware or hardware drivers in the hardware's native command set can be through wrappers which communicate with CMS in well defined XML. Interface between these layers is also through XML formats. This insulates the upper layers from changes to hardware as long as wrappers are rewritten to preserve XML data formats
3	Configurability of existing hardware and expandability of hardware with minimal impact to upper layers	The layered and abstracted architecture as described above similarly supports expandability of hardware functions. Further the individual CMS components should support configuration files to store parameters for hardware which can be read by the wrapper layer allows changes to range and scale etc. with no software changes
4	It should be possible to reuse significantly most of the lower level hardware and drivers / software and interface to existing third party applications and drivers rather than rewrite the same	As described already the hardware and drivers are isolated from the applications through wrapper programs that format the configuration commands and responses into well defined XML. Reusing existing or third party hardware and drivers is then a matter of writing the appropriate wrappers that generate the XML data files
5	Graphical user interface, whether for local or remote users should be uniform. It should require only Web browsers for access	It is proposed that GUI functionality should be developed around Java web objects and exposed through web server both on the local intranet and through appropriate security layers on the internet. To allow for differences in browsers, the validation should happen with at least three major browsers and two versions of JVMs
6	CMS software supports high availability and redundancy	The architecture should support mirroring of key components

7	Both local and remote access over internet should be possible with appropriate authentication	It is proposed that GUI functionality should be developed around Java web objects and exposed through web server both on the local intranet and through appropriate security layers on the internet. To allow for differences in browsers, the validation should happen with at least three major browsers and two versions of JVMs
8	It should be possible to track, save and restore the state of the CMS and set the system into standard states automatically with minimum manual intervention	The middleware layer of the CMS should incorporate a state machine with state updates for both user commands and changes to the hardware state. State should be saved every 3 seconds for restoring when required and also a provision to save any state as a standard well known state should exist
9	Multiple access mechanisms to the system should exist: i) Online (keyboard / mouse); ii) Command line;	The CMS should include multiple ways for a user to interact with the CMS: i) Web GUI which has screens with command buttons, drop down, radio buttons etc. ii) A command set is required to be developed for all major actions which is achieved through running exe files from a command line mode. These will trigger the XML messages similar to those from the Web GUI.
10	Multiple access mechanisms: iii) A batch command execution / scripting mode supported with basic looping / flow control; iv) TCP/IP session v) Possibility of integrating an SMTP and SMS gateway optionally, especially for critical events, alarms	iii) Important sequences of functionality need to be combined as scripts. CMS should incorporate a Perl interpreter engine that can execute scripts where flow control / looping is achieved through Perl syntax and the control actions are triggered through the command set which are interpreted by the Perl engine as system commands and passed to the command environment. iv) It should be possible for an authorized user to open a session through a remote Telnet / rlogin shell over TCP/IP to access the command set of the CMS without going through the GUI v) Architecture should support the integration of standard SMTP and SMS gateways either during implementation or at a later stage
11	Key events should support audio alarm indication	There should be an audio interface component with an inventory of typical sound alarm indications which can be called by the middleware layer as a response to events in addition to visual alarms

12	Users should be presented with intuitive interface providing only information / controls that are relevant to the role e.g. Astronomer, Operator, Engineer	The access control module should determine access levels based on user role and the presentation layer objects appear based on the encoded access privilege level
13	The GUI should support visual presentation of data, trends and logs in user friendly graphical format	Basic trending of graphs should be achieved through Java packages including Java 3D. For advanced graphing options it should be possible to expose hooks to interface open source applets. It should be possible to represent monitor logs in quasi-real time. Long term archival of logs and the ability to search by date / time should be supported.
14	Ability to interface third party presentation utilities (such as DS9)	The built in graphing / data presentation utilities should also allow overriding. A link to external executable programs that can open in separate windows should be provided
15	Ability to link to applications, utilities and documents not part of CMS (manuals, cookbooks, proposal system, bug trackers, websites)	There should be appropriate links to these external executable programs or url links that can open in separate window

2.3 Scope of the specification project and overview of the contents of this specification document

The rest of this specifications document describes

- Proposed CMS architecture and its salient features and benefits
- Platform for the CMS deployment and tools that could be used for the development
- Software Requirements including
  - General specifications of the software approach to develop the modules and components
  - CMS state chart overview
  - GUI approach and look and feel
  - Details of the XML based interface mechanism
- Baseline software requirements: While the general architecture is required to permit changing of the fields on the screens and number of data elements represented, a baseline specification is still needed to form the basis for the first implementation of the system: This includes
  - Baseline requirements of the control and monitor data for the individual modules for the Optical and Radio telescope to be represented in the CMS
  - Description of the major GUI screens

(High level execution plan and effort estimate: intended to support the next stage of the project i.e. the execution phase is a separate document as decided during an earlier review stage)

As per the objectives of the first phase, this is primarily a requirement specification. It does NOT include a high level or detailed design specification that can directly be implemented into code. Further the requirements specifications are intended to be at a level of 90% completion, since requirements for certain sub-systems (e.g. servo for NCRA radio telescope) are still to be frozen and some changes during the implementation and validation phase are expected.

It should also be noted that the overall architecture and component models proposed herein are at a conceptual level with the purpose being to better specify the requirements and provide a high level overview rather than an exact implementation architecture. There are certainly expected to be alternative realizations that can be proposed by the implementing vendor. The architecture will also undergo refinements and changes during the phase of the design as implementation, performance and deployment issues are dealt with in more detail.

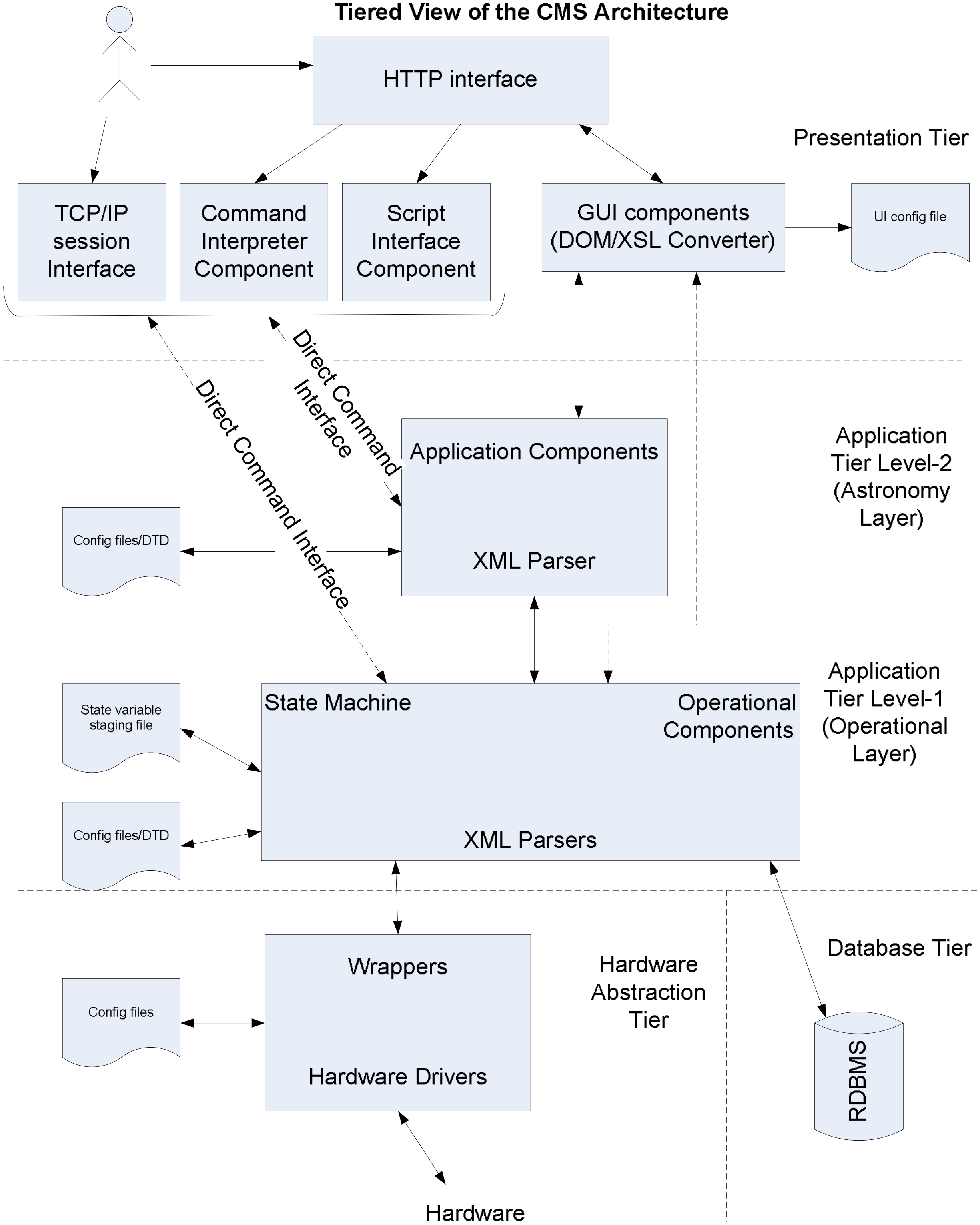
3 System architecture

3.1 High level specification of the system architecture

The CMS should be implemented using a multi-tier architecture reflecting the layers in the previous section. Each tier would interface with the ones above and below, by exchanging information through well defined messages. Every component uses a XML-DTD for defining its behaviour and XML files for information interchange. The details of interfacing the component, XML_DTD and XML file is well explained in the interface requirements section.

This will facilitate the replacement or upgrade of the components of any tier relatively smoothly, and transparent to the other tiers as long as the message interchange protocols are respected by the upgraded system. A graphical description of the proposed architecture is provided in the figure below.

Tiered View of the CMS Architecture

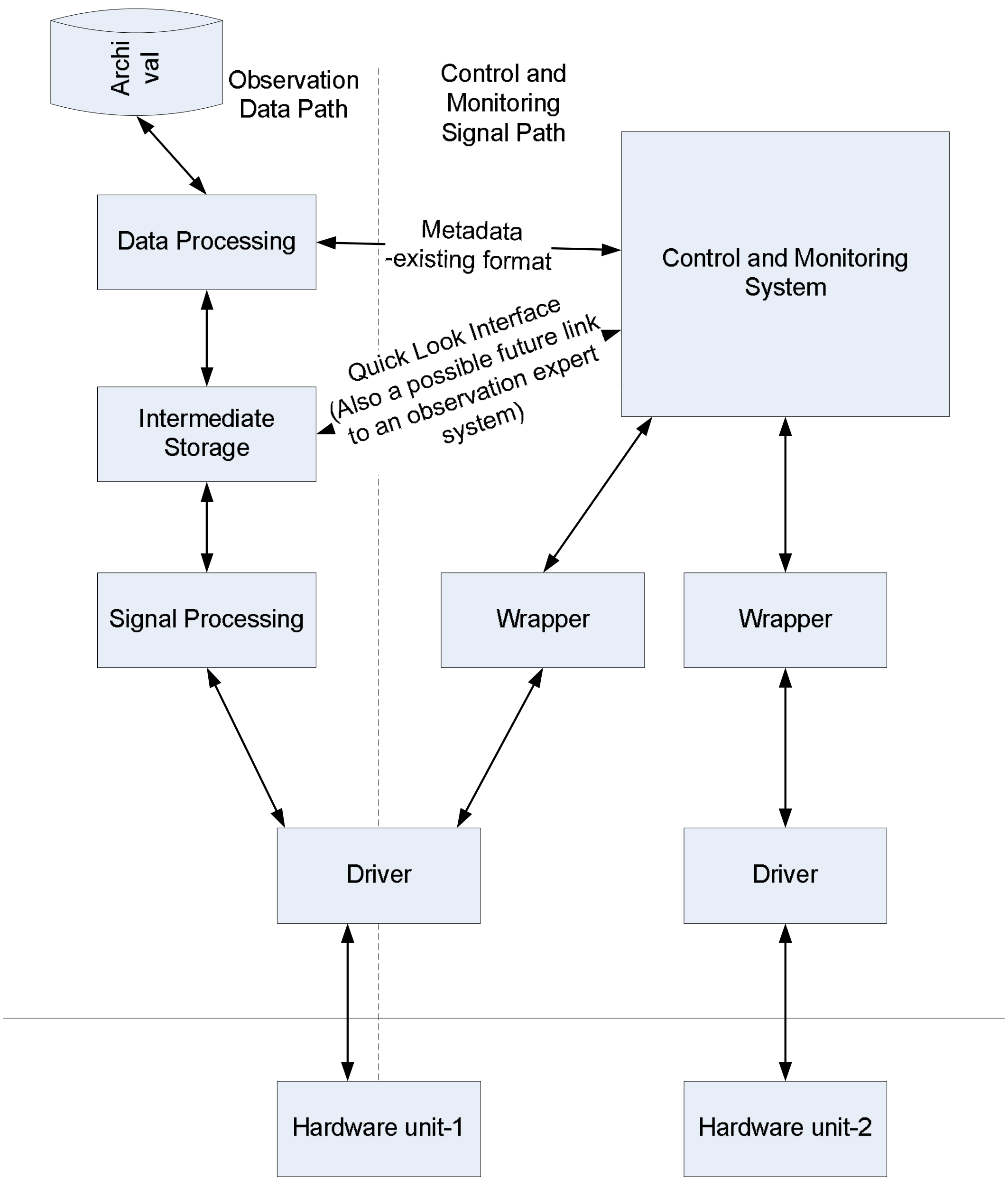


3.2 Separation of observation data and control / status paths

In developing the CMS specification it is proposed to use information flows using message passing, based on XML. Based on the volumes of observation / raw data, it is believed that this would add significant overhead to the data processing and cannot be considered viable. Therefore the architecture should clearly separate the observation data path and the control and monitor signal / event flow. This is indicated in the figure below. The rest of the CMS architecture and requirements specification focuses on the right hand side path, assuming that the existing data flow path is not touched. It should be noted that some of the hardware will have both flows, therefore two separate interfaces are needed.

The interface for the observational data lies outside the scope of the CMS. However, hooks should be provided to launch applications which will make trend plots of observational data etc. (i.e. the “quick look interface” shown in the figure below). The CMS will also supply the required meta-data to the components along the “observation data path” to monitor their health etc. (e.g. back-end as a sub-system).
Similarly the quick look interface could also in future be exploited to provide an input to an “Expert system” which could flag obviously erroneous observations or to guide an observer to take appropriate actions.

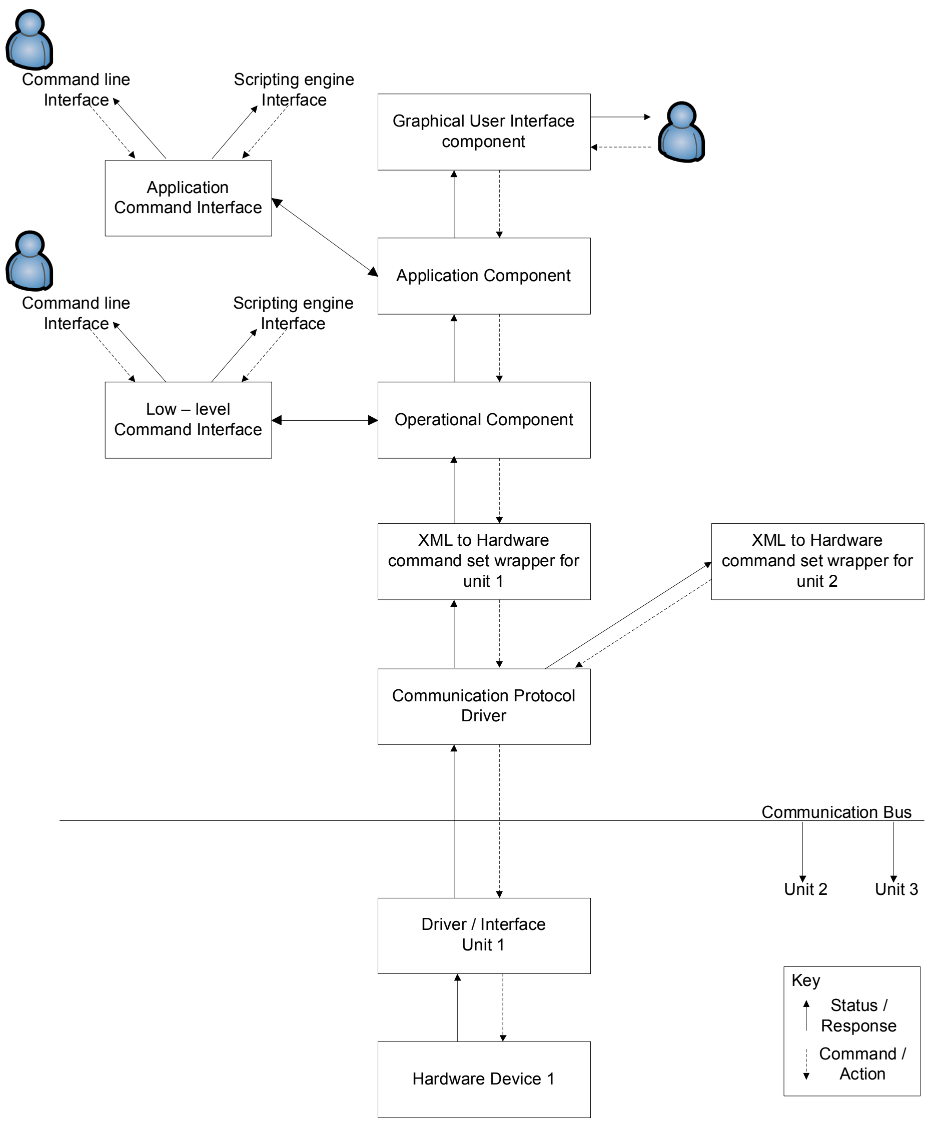
Conceptual Separation Data Path and CMS command/ status flow



3.3 End to end view of command / status flow:

Based on the specified architecture and considering any particular telescope module, the following would be a schematic representation of the control and status / response flow.

End to End Flow of Command & Status



3.4 Physical Deployment of the CMS components:

The physical deployment of the architecture can be carried out as per the figure below. It consists of a front end web server receiving the HTTP traffic, with the CMS application hosted on a Java EE

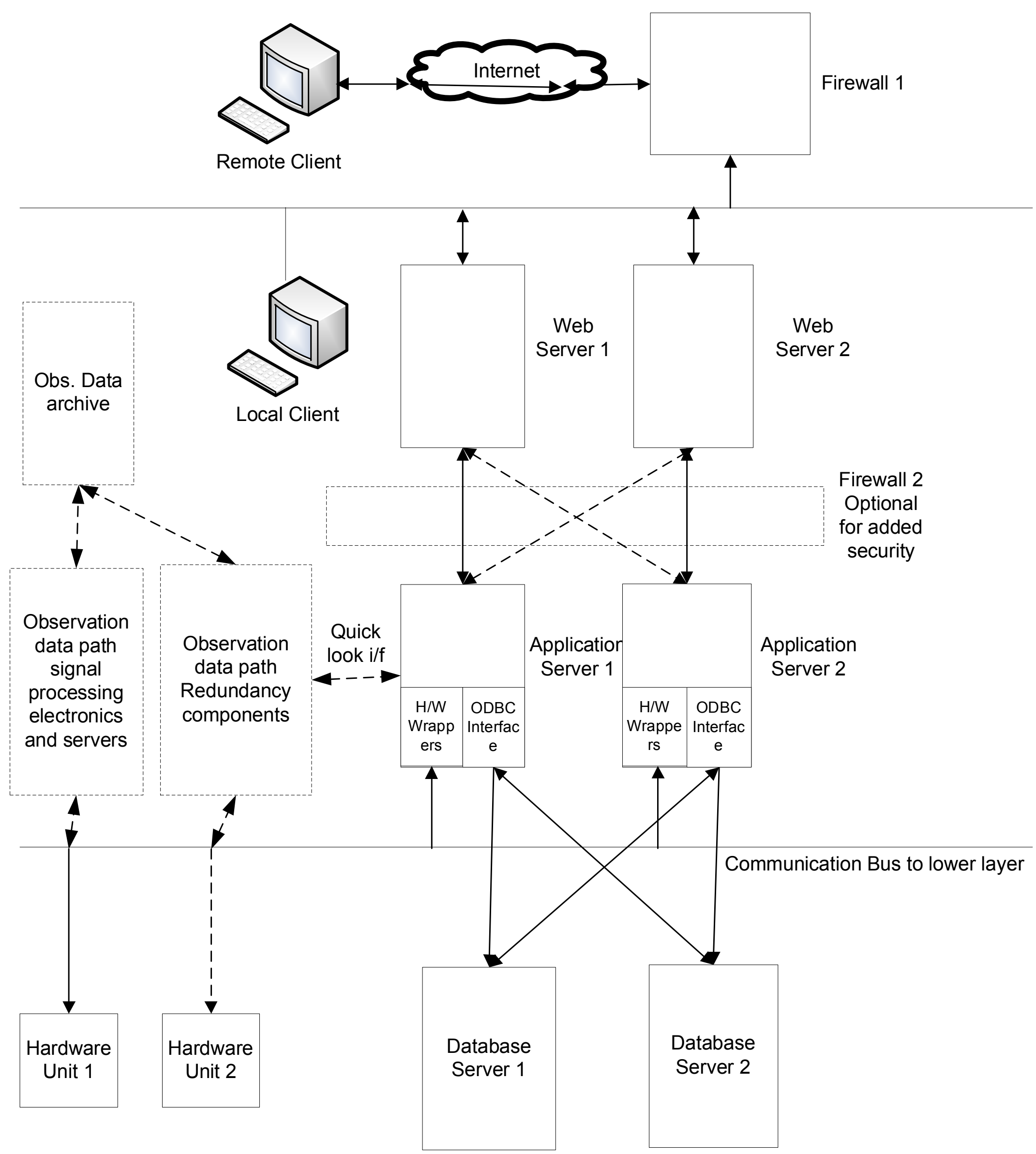
Application server. A separate database is used as a configuration and state variables repository. There are two firewalls for security, the second is optional for added security.

Although the hardware abstraction layer wrappers are not part of the CMS scope; however it is possible to implement them in the CMS application server, or in the hardware. The architecture supports redundancy by providing mirroring at every stage. Further any combination of the web-server, application server and database server can be made active. This would require the implementation of components to support manual changeover of the active servers. The implementation should also support agents which can automatically redirect the connections in case of acknowledgement timeouts or in case failure messages are detected. In order to ensure that states are adequately backed up, the active app server saves state to both the database servers, thus allowing restoring the state and configuration files by making either server active.

However it is not expected that the level of mirroring equals that of a hot standby redundant deployment that can automatically transition in all cases transparent to the user, since even with the mirroring, some of the actions may have to be rolled back and repeated in case of a changeover.

A summarized representation of the observation data path is also shown for completeness; this path also includes some redundancy. However as the observation data path is outside the scope, the details of the component deployment for this path is not shown here.

Physical Deployment with Mirroring



3.5 Benefits of proposed architecture

The architecture proposed meets the major objectives of modularity and allowing easier upgrade of individual hardware or software components with minimal impact to other systems.

This allows the replacement or upgrade of the components of any layer relatively smoothly, and transparent to the other layers as long as the message interchange protocols are respected by the upgraded system.

The salient features of the system are reiterated here:

- Modern and open user interface that is intuitive and easy to use
- Seamless access at local machines and across the web
- GUI presentation and access to functionality that changes as per log-in role and needs
- Architecture that supports
    - System Integration and re-use approach; where existing functional hardware and software can be reused where needed
    - Abstraction – Hide the implementation details from the user
    - Modularity and loose coupling
    - Scalability
    - Open and standards supporting design for interface and data exchange

4 Platform and tool specification

Based on the proposed deployment diagram and architecture, the implementation can use the following software platforms / products which satisfy the project's open source requirement. These are indicative; the implementation vendor may propose other alternatives which meet the overall objectives of the CMS.

i) Java EE 5: This technology is suggested to be used for most of the software development considering the fact that Java EE is well suited for browser based application.

ii) C / C++ for wrappers and components needing a faster response including the state machine component

iii) Application / Web server: Geronimo platform from Apache which consists of an Apache web server and an Application server licensed for Java EE 5 based on Tomcat which has an extensive deployment history and support in the open source community

iv) Relational Database: MySql is a popular open source database which is used to store the information. It is used to store the values of the parameters to be monitored and/or controlled and those values can be used to retain the states of the software in case of power failure.

vi) Tools:

1) Eclipse development environment together with needed plug-ins for Geronimo development / C++ etc.;
2) Abbot – GUI and Performance Testing.

The Abbot framework is a Java library for GUI unit testing and functional testing. It provides methods to reproduce user actions and examine the state of GUI components. The framework may be invoked directly from Java code or accessed without programming through the use of scripts.

5 Software requirements specification; Interface requirements specification

5.1 High level view of the role of the CMS in the telescope operations:

The central control and monitoring system is like the central switchboard for all the command and control of the telescope. It integrates the various distributed controls for the hardware and acts like a master or supervisory control for the same. Further it provides the access to end users like operators and also engineering and maintenance personnel a control and status monitoring view of the system and on going observation. This system thus provides the UI and master control and monitoring of the system and processes the filtered data into reports.

Operational logs, trends and important manuals also need to be linked to from this system. In turn CMS interfaces with the hardware components / sub-systems of the telescope.

5.2 Overview of specifications of the tiers:

i) Hardware abstraction tier: The function of this layer is to abstract the different data representation and command formats from the hardware from the rest of the CMS. The upper interface of the hardware abstraction layer is XML data files. The lower level interface would typically be byte frames or packets.

ii) Wrapper program high level specification: (Note: Writing the wrapper is outside the scope of the CMS implementation, however the interface specification – during the design phase is within the scope). In order to support a faster execution speed, it is suggested that programs in this layer should be programmed in C/C++. Each hardware sub-system interface or driver should have its own wrapper program. This ensures that changes to hardware can be achieved with minimal impact. Further every wrapper program implements the generic XML message to command translation facility and vice-versa; however hardware scale and conversion constants, ranges and the like are stored in separate configuration files which are read by the wrapper program. This adds a further degree of flexibility in making hardware changes without a program impact. In turn this configuration file is updated from the central configuration parameters database.

   Note: While the wrapper logically forms a part of the Hardware Abstraction Tier, they can as well be implemented in the application server, thereby needing no changes to the communications to the hardware. It is also possible to implement them in the ABC, in case of the radio telescope.

iii) Operational tier: This layer houses the major components contributing to the operation of the telescope such as telescope scan / track path updating, data format conversion for quick look presentation, monitoring vital parameters and raising alarms based on limits for parameters received from the lower layer etc.

   This is also the layer that contains the state machine of the telescope. The state machine needs to be carefully designed using OO methodologies of inheritance / overriding etc. to allow generic state templates, sub-system state nesting etc.

iv) Astronomy tier: This is the uppermost logic layer and uses objects that are closest to the astronomy / user domain. The typical components herein include report generation, catalog look up, units conversion etc. This is also the layer that contains the access control / login module.

Application component high level specification: The components in the middleware should be largely coded in Java; interface to the lower layers is through XML files and thus the middleware components include in-built Java parsers. The state machine component should best be coded in C/C++ to allow higher speeds of state storage and retrieval.

v) Presentation tier: This is the tier that the user actually interacts with. The major components herein include the presentation / UI rendering, interface

Presentation tier high level specification: The presentation layer containing UI screens should be developed using Java web objects: The layout and element types should be obtained from XML files / XML-DTDs. Similar to the middleware layer, the other components in the application layer should be largely coded in Java with built in XML parsers;

An important requirement of the CMS is re-emphasized here: In every tier, the overriding design concept is modularity, loose coupling between modules based on message passing, and maximizing the ability to modify and re-configure components using configuration files on the fly without code changes. (This includes the requirement to have easily modifiable GUI screens). A conceptual view of how these could be achieved architecturally is described in further sub-sections.

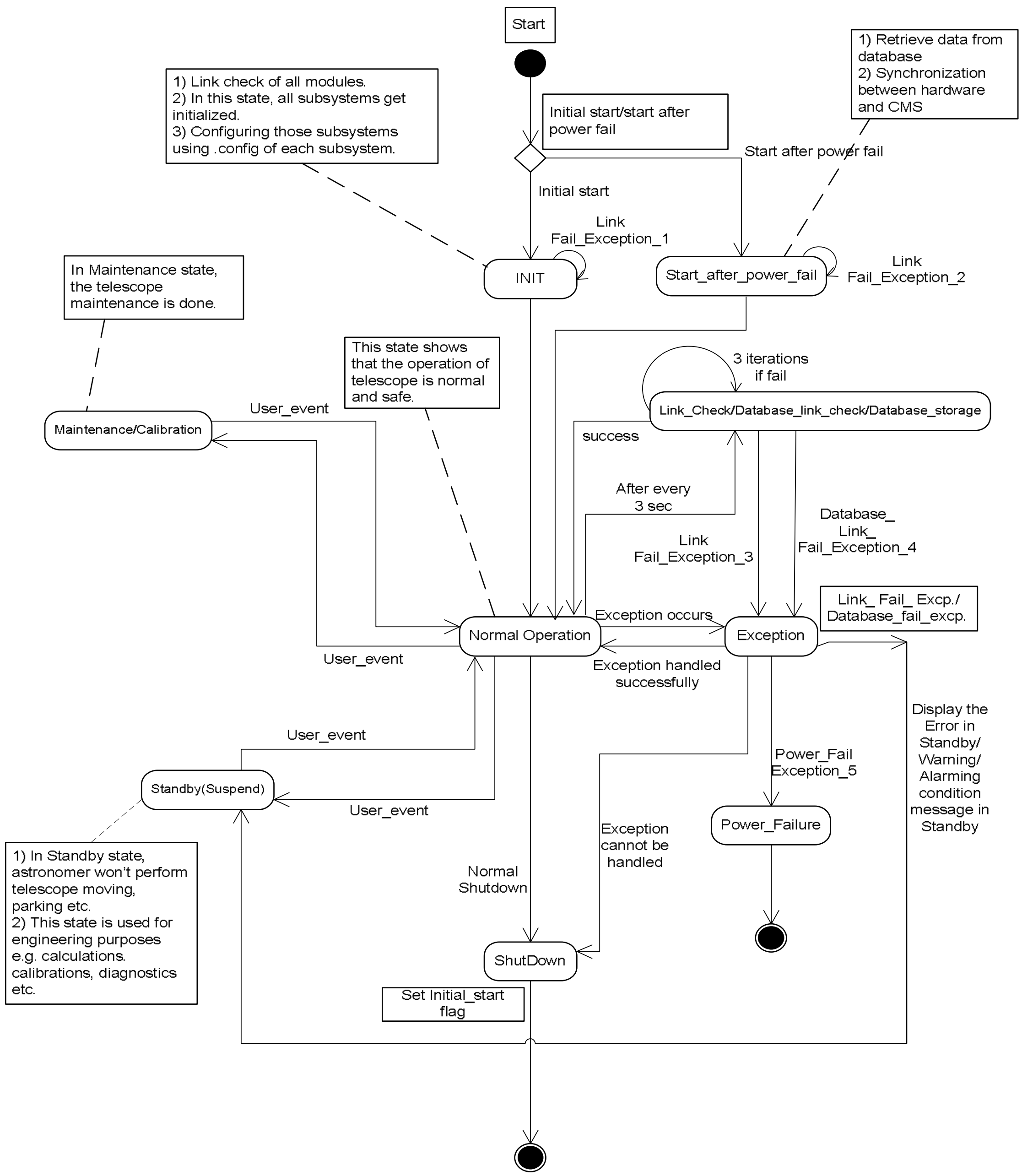
5.3 Summary of typical telescope operations and related CMS functionality:

An overview of the telescope sub-systems / operations is provided below and related to the major CMS functionality needed for the same. The detailed overview of the sub-system control and monitor parameters are described in Section 6.

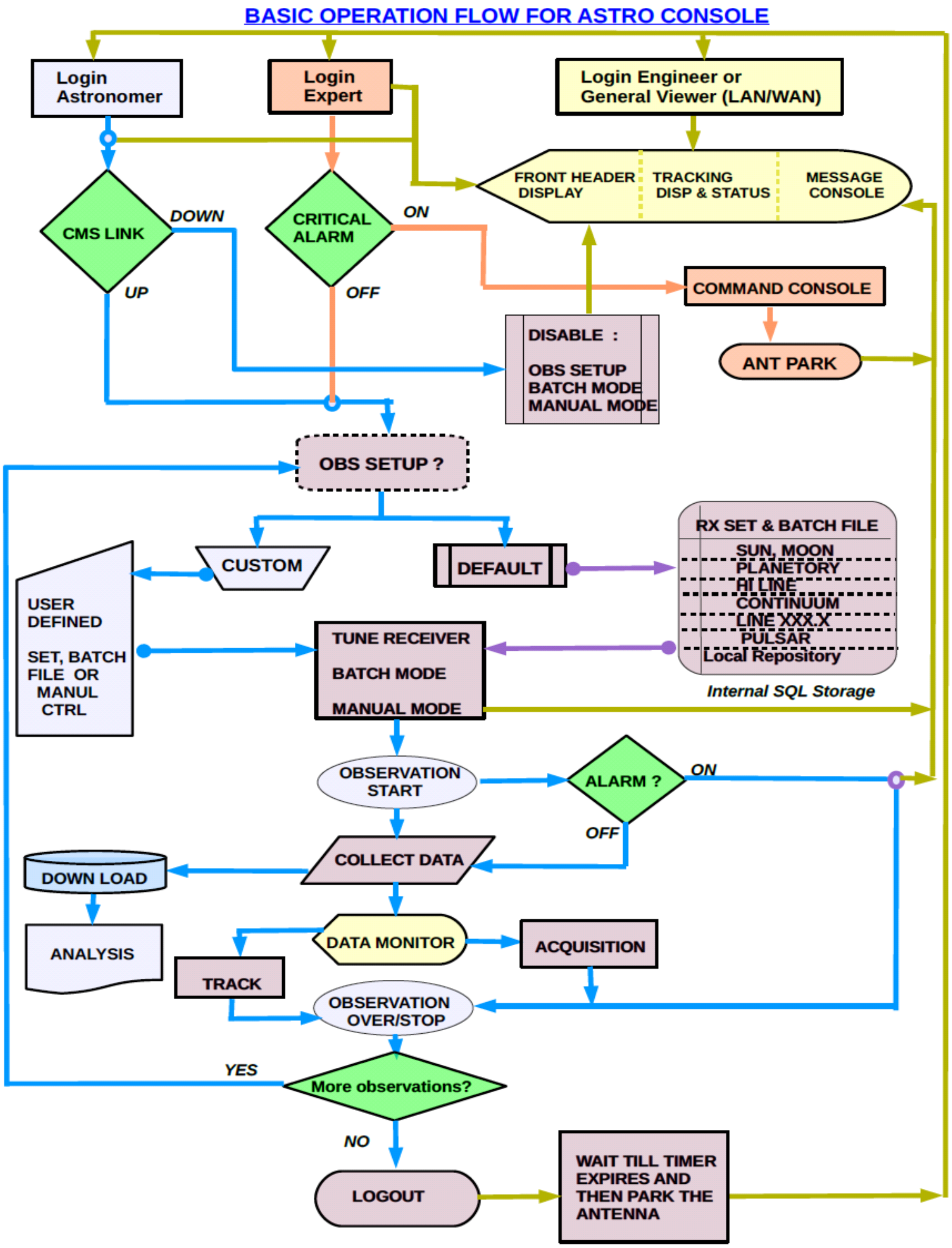
1) An observation can start at any point in the visible sky region and continue to track a celestial object or objects being observed. This involves being able to track the telescope along the expected observation path. This is achieved by the Servo subsystem, and the CMS provides the servo the tracking parameters. The CMS is responsible for ensuring that the needed observation is carried out correctly and aborting it in case of any limits being reached.
2) The data collection consists of choosing an appropriate instrument or feed and configuring the same including applying filters etc. This may be either manual or automated. However the CMS reflects the choice of the right instrument / feed and its configuration.
3) Signal processing is carried out on the raw data and archived, and some of the configuration parameters and meta-data for the same are provided also by the CMS.
4) Ambient parameters like temperature, pressure, wind and turbulence can affect results and need to be logged along with the data. Besides this some of the parameters also form part of the safety shutdown system along with fire and smoke monitoring. Therefore a continuous logging of these parameters is always carried out along with every observation by the CMS
5) The telescope system also requires several facilities (electrical distribution / backup hydraulic systems, heating and cooling of panels and specialized cooling including liquid nitrogen etc. for low noise observations and the like. Managing and controlling all the supporting infrastructure and safety interlocks and shutdown systems is usually achieved through a dedicated PLC or controller. The CMS receives status and alarms in this regard and can abort an observation in case of critical conditions.

6) As per the suggested architecture, the CMS should include independent modules to achieve the needed functionality for every sub-system including initialization, operation, shutdown procedures, alarm conditions etc. Each modules has its own configuration parameters and rules, and diagnostic and exception handling needs to be available for every module
7) While performing any operation, the CMS should also be aware of state changes to the systems and be capable of saving ans restoring state. A detailed view of the State Machine for the CMS is explained in the next sub-section

A conceptual state chart for the CMS



The 'Normal Operation' state can be expanded as follows:



5.4 State machine model description

The state machine shown in the Figure "**A conceptual state chart for the CMS**" represents the overall operation of the control and monitoring system. Thus as shown, the INIT state will be the first state of operation. The 'link_status' is checked in the states wherever necessary. After initialization and configuration the CMS enters in the Normal state in which the normal operation of the telescope can be performed. The normal shutdown can be performed in which the CMS will enter in the ShutDown state.

Whenever there is any exception in the normal operation or in between the other states, these will be handled by the Exception state in the state diagram.

In the normal state of operation, every parameter is periodically stored in the database and these stored values can be used in the case of retaining the state of software in case of power failure and for starting the CMS after power fail, the telescope will be restored to the previous condition by fetching the contents that were previously stored in the database.

The Normal Operation state – Astro Console can be expanded as follows:

Normal Operation State:

In the Figure "Basic Operation Flow for Astro Console", the following convention has been followed:

1) Lines marked with blue color: Astronomer purpose.
2) Lines marked with red color: Expert purpose.
3) Lines marked with yellow color: Display purpose.

The CMS link is always checked whenever any observation is to be performed. The observation setup is selected either as Custom or Default. The Custom observation is user defined and it can include batch file or manual control for the observation. On the other hand, Default setup involves the pre-defined positions like Sun, Moon, Line XXX.X etc.

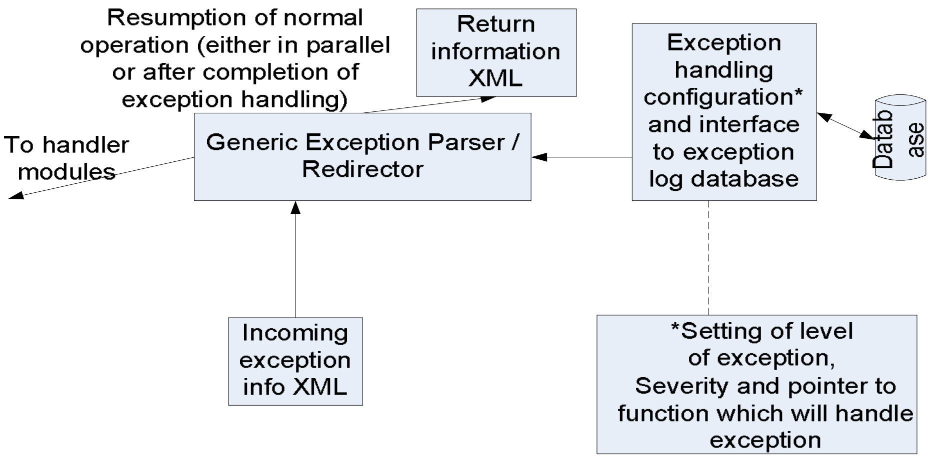
In case of any alarms, the proper message will be given on GUI screens. There are basically three levels of alarms which is also configurable.

i) Critical level alarm: This will happen in cases like any system fails.
ii) Warning: In this level, the no observation can be performed by astronomers.
iii) Information level: This gives information regarding the status of different subsystems. Normal observation can be carried out in this level of alarm.

After this, the data collection can begin. Collected data can be monitored and analysed using the quick look interface. If more observations are to be performed, the whole flow will be executed again. After selecting 'logout' option, the system will wait for some time for any action to be performed. And after expiry of the timer the antenna will be parked. And, login screen will be prompted. Antenna parking is performed by the system in this case. But user and/or an expert also can park the antenna.

Exception handling:

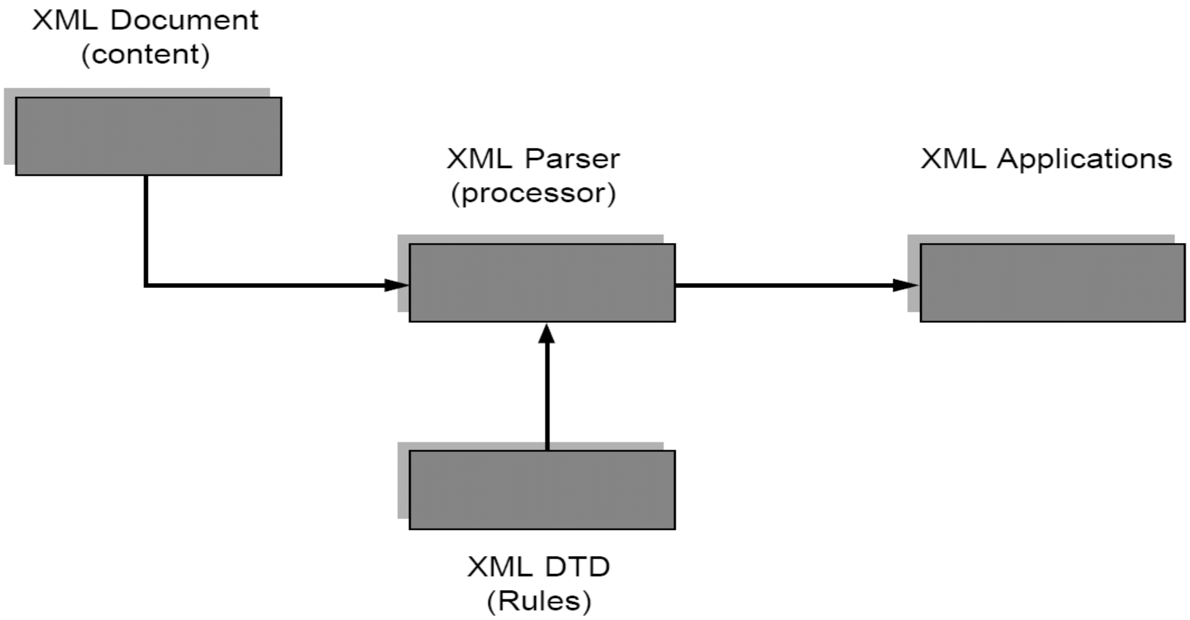
It is proposed that exception handling be a dedicated component in the state machine. The representation of the exception handling flow is depicted in the figure below. It includes a generic exception parser / redirector that handles all exceptions, based on the exception information received and decides the action to be taken based on an exception configuration file which includes action information on all exceptions, and also default information. This also maintains the exception history log. The actual handling of each exception is carried out by a dedicated handler based on the exception. Return to the normal operational flow can happen either in parallel or after the exception handler returns; again this is defined in the exception configuration.



5.5 Overview of internal and external interface mechanism

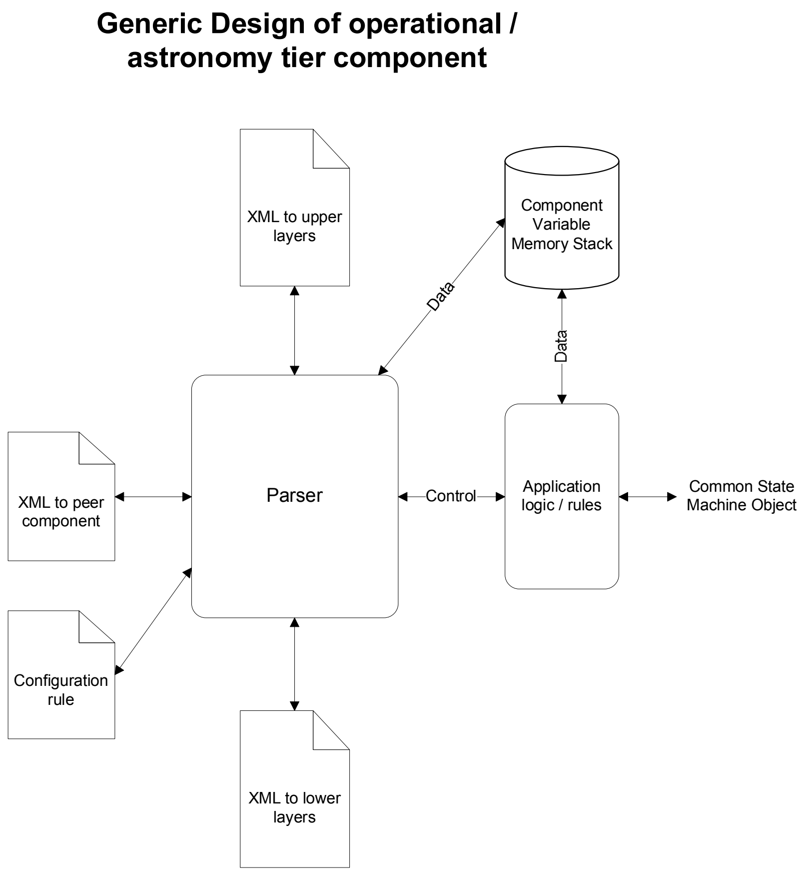
In order to make the system self describing and in conformance with emerging data exchange standards, it is proposed that the messages exchanged between layers should be in XML, and following a well defined XML DTD hierarchy to be developed for this purpose. The overall structure of the interaction between the XML file, XML-DTD and CMS components (XML application) to be implemented is described further.

5.6 XML based component interface overview

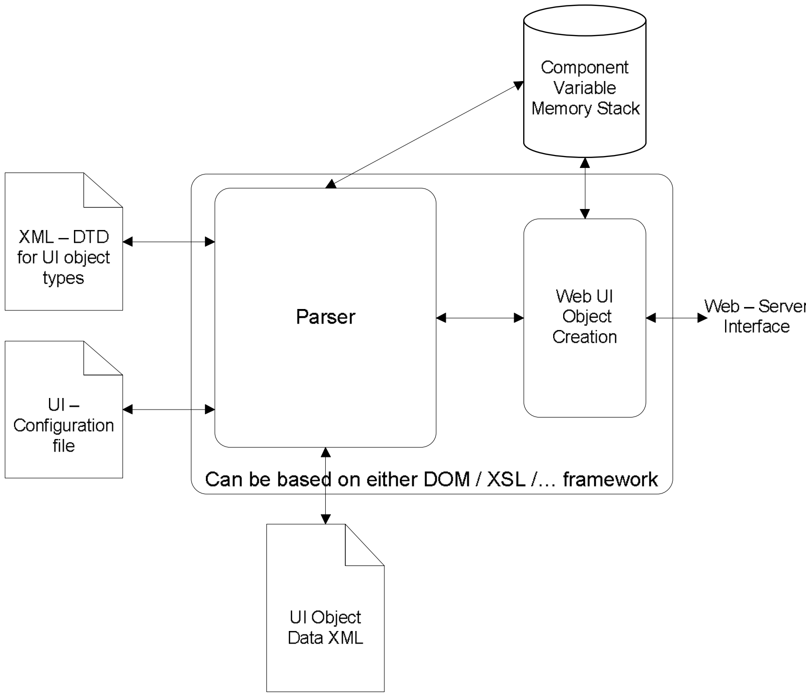


<u>XML based processing – general concept</u>

As shown in the figure, XML file contains the values of the parameters to control or the status data for monitoring. The DTD file contains the definition of each node/tag in XML file. A conceptual view of the implementation of this scheme for the operational / application and presentation components is shown in the figures below.


Generic Design of operational /
astronomy tier component
XML to upper layers
Component Variable Memory Stack
Data
Data
XML to peer component
Parser
Control
Application logic / rules
Common State Machine Object
Configuration rule
XML to lower layers

Generic Design of a GUI Component



5.7 User interface specification

The GUI screens provide interactive screens for every user of the telescope including astronomers, engineers etc. and it can also be used by relative amateurs such as students. Therefore it needs to be quite user friendly.

Conceptually the GUI should be built up of different frames as depicted below

<table>
<tr><td colspan="2">HEADER FRAME – Containing logos, time info and general background like proposal observation ID etc</td></tr>
<tr><td>Main FRAME – Variable based on the user mode, the sub-system being viewed etc. This has various tabs to organize additional information</td><td>Command / response and sub-system specific messages and alarms frame</td></tr>
<tr><td colspan="2">Footer FRAME – Containing important alarms and status messages that are displayed irrespective of the sub-system screen</td></tr>
</table>

The “Main Frame” is where the user interacts with the CMS most extensively. It is context and user specific, e.g. it will change depending on whether on whether the telescope is in a normal mode or maintenance, the user role and access privilege level etc., and also based on the selected subsystem (in the engineering mode for instance). For example, if a servo engineer logs in then he can view the Engineering GUI of servo subsystem in the main frame.

As the main frame is expected to include a lot of functionality, it should include separation of the controls / information display into logically organized tabs.

The header section typically displays the different time formats and a quick look of the summary information such as proposal details etc. status of the telescope. Bottom part contains alarm state, message display for the current state of the telescope etc. The contents of the bottom part and header section remain constant regardless of the subsystem.

Command / response and sub-system specific status and alarm monitoring should implemented on another frame which initially appears in the middle to the right of the main frame. However this is a “dockable frame” which should be capable of being maximized and moved around and even dragged to another screen (monitor) whereas “MAIN FRAME” will be displayed on main screen (monitor).

In order to support the objectives of developing a GUI that is configurable on the fly, the elements / icons on the screen should generally be restricted to:

Control functionality:

i) Alphanumeric Text boxes: For configuration or set point data as well as command messages
ii) Buttons: For activating or triggering actions, submitting data and confirming etc.
iii) Radio buttons and check boxes: For selecting between options

iv) Drop down menu: For selecting between a list of multiple items

Status / alarm functionality:

i) Scroll text boxes: For alarms and response message windows
ii) Pop-up dialog boxes: For status messages or critical alarms requiring activating or triggering actions, submitting data and confirming etc.
iii) Light indicator: Showing on state, activity or parameter within or beyond certain range

Specialized screen features:
Besides this, there would be need for some specialized features in few of the screens for trending and plotting.These can be implemented using custom applets, including 3rd party applets. Java itself provides packages with extensive support for plotting, including the Java 3D API for 3D rendering

A description of a typical baseline implementation is provided in the next section of this document.

5.8 Access control / login component

This component's purpose is to take care of login authentication to the CMS software and also provide the UI access and privileges based on the context of the user and his role. We can use **Java Security** for handling the Authentication and Access Control.
The access control to the CMS should be based on the following hierarchy:

i) Privilege Level: This is the granular level of access control which specifies the actions that may be performed by the user having that privilege. The privileges are objects that include the set of permissible actions and sub-system identification. For example ability to view status is a privilege level that may be available to most users, but only an engineering user may have a System Reset privilege. Further a servo engineer would have full privileges for the servo sub-system but lower privileges for the other systems.

ii) User Role: This is a set of privileges that taken together define a typical user access level. There will be an initial set of user roles based on the current baseline requirements. However the administrator should be able to create new roles, delete, edit and inherit from the existing roles and modify the privileges available.

iii) User-ID: Multiple users can be created for every role and they automatically inherit all the privileges assigned to that role. User-ID accompanied by valid password is the basic attribute identifying the person logged in and distinguishing users from each other and providing audit capability for actions taken.

iv) Current location: For certain actions it may be necessary that the user is in the local location, i.e. in the telescope vicinity. Therefore during login the remote /local attribute for the user will also be distinguished and saved and some of the privileges that are otherwise available may be disabled for that user.

Implementing restrictions based on this parameter is an optional, as it may actually be desirable to permit most maintenance operations etc. remotely, though with appropriate authorizations

Baseline implementation of roles:
After discussion with IUCAA and NCRA, PSL has identified the following baseline roles and their privileges that define the access control needs currently.

Following are the three parent level roles.

1 **Prime User (Astronomer/Co-Astronomer).**

In this role, Following should be the privileges and the authentication recommended.

Login: He should be able to login to CMS Software through LAN/WAN.

Login Authentication: Based on time, proposal and password

Role: Telescope usage for Scientific purposes i.e. Astronomical observations

In this role their might be following child roles depending upon the privileges configured for them

i. **Astronomer**

1) This User should able to access his/her observing schedule. If required, user can do online modification of observing schedule by changing RA, DEC of astronomical object or adding new calibrators or sources, changing receiver tuning, observation mode and reschedule the observation timings within the limit of his/her observing period.
2) This User should able to control telescope units relating to observation only, and only in the time period allocated to him / her.
3) Batch processing and procedure writing from basic operating commands should be possible for observing requirements.
4) User should monitor the progress of observation via basic tools for online monitoring of observational data (time and frequency domains) and other parameters like telescope settings, observing mode etc.
5) There should Provision for feedback form about observation.
6) This User can download the observation data with all his/her observing configuration files and flagging files.

ii. **Co-Astronomer**

All tasks of Astronomer as mentioned above expect 1, 2 and 3

2 **Engineering and Telescope maintenance (Expert/Engineers).**

In this role, Following should be the privileges and the authentication recommended.

Login: He should be able to login to CMS Software only through LAN.

Login Authentication: login and Password.

Role: Telescope usage for maintenance and debug proposes.

In this role their might be following child roles depending upon the privileges configured for them

i. **Expert**

Role: 1) This User should responsible for the operation of telescope by checking the correctness of telescope systems. Also, ensure the correctness of execution of scientific observation.

2) Debug the telescope system on failure.

3) Perform the safety measures on failure of sentinels.

Privileges and Job Execution:

1) All privileges of **Prime user, Engineer and developer.**
2) This User Can Control and configure all the telescope units in order to fulfill the observational requirements by ensuring the correctness of observation (whether given observing parameters by astronomer are set and executed properly) and handling of some basic trouble shoots.
3) This User Can interrupt the observing sessions if critical condition regarding the telescope safety arises due to exceeding of the software/hardware limits of critical systems like temperature, wind, servo encoder/motor currents, local network failure, and some uncertain issues of system failing.
4) An optional / configurable feature is whether or not the expert be able to control the privileges if He/She logs in at a remote location.

ii. Engineer- Telescope Systems.

Role: Debug, maintain, modify, replace or evolve the telescope systems/subsystems & hardware modules in engineering point of view.

Privileges and Job Execution:

1) All privileges of Engineer and developer.
2) This User Can create/edit/modify system configuration database, XML files at GUI and driver module level. Engineer can define the limits or threshold values of systems.
3) This User can access the raw data and monitoring points of systems.
4) This User Should is able to send low level command directly to hardware for debugging purposes.
5) This User should have an access to data plotting tools.

iii. Engineer – Software Support/Developer.

Role: Modify or develop required scripts or programs, maintain the telescope software and install firmware/software of the telescope and new packages on existing/new machines.

Privileges and Job Execution:

1) This user should be able to develop new scripts/programs or modify existing ones in order to fulfill the engineering and scientific requirements
2) Install/Compile any available/new packages on existing or new machines.
3) This User should able to perform basic software maintenance and test jobs like flash EEPROMS or changes in firmware/software versions.
4) This User should able to change basic directory paths and files' names of configuration or data files of telescope software.
5) This User should able to manage basic computer resources relating to telescope software like Serial/Parallel Port Ids or PCI slots and its configuration files.
6) This user should be able to use basic data rendering tools so that one can read binary data and replay/interpret the system status for debugging purposes.

iv. **Administration Engineer – Database/Login**

Role: Ensure about new login-ID with given time duration for each observing project. Archive the astronomical data, settings, logs and error flags files of observation. Maintain the database.

Privileges and Job Execution:

1) On the acceptance of observing proposal create new login ID, ensure observing time for given login. Inform this to user via email for confirmation.
2) Archive and take backup of all files relating to observation like observing plan/command files, Error logs, system settings and astronomical data with standardize names.
3) Provide the observation data/archive on demand to the users.
4) Ensure the disk space requirements.
5) This User should able to assign IP addresses, configuring/updating a sub-net of telescopes.

3 General Observer.

In this role, Following should be the privileges and the authentication recommended.

Login: He should be able to login to CMS Software only through LAN.

Login Authentication: login and Password.

Role: Telescope usage for viewing the data and not for controlling at all.

Privileges and Job Execution: This User should able to monitor the telescope monitoring parameters, observational data etc.

5.9 Trending / plotting and external components

The specifications for the UI described above can handle the basic control, command and status display operations. However there is a requirement for trending and plotting. This can be achieved by either including applets which can be developed for this purpose or imported as third party components. Some of the options for the same are described here:

For plotting real-time data:

http://quies.net/java/math/plot/

JFreeChart:

http://jfree.org

JFreeChart is a popular open source Java charting library that can generate most common chart types, including pie, bar, line, and Gantt charts. In addition, the JFreeChart API supports many interactive features, such as tool tips and zooming. JFreeChart provides an excellent choice for developers who need to add charts to Swing- or Web-based applications.

Java-3D API.

http://java.sun.com/javase/technologies/desktop/java3d/

Java 3D is an addition to Java for displaying three-dimensional graphics.

The Java 3D class library provides a simpler interface than most other graphics libraries, but has enough capabilities to produce good animation. Java 3D builds on existing technology such as DirectX and OpenGL.

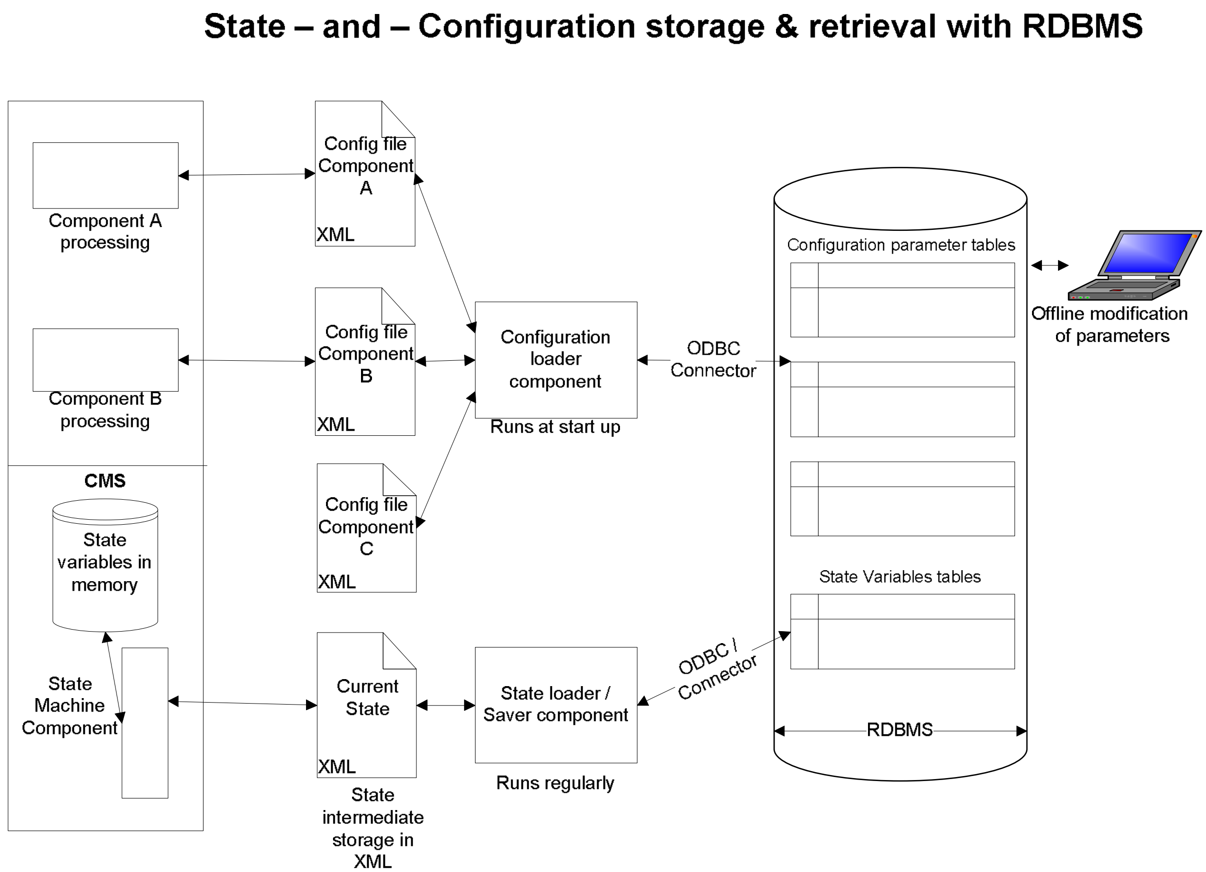
Alternately for applications which are fully executable, a link to launch the same can be provided from the CMS screens and these would be launched in separate windows.

5.10 Relational Database Requirement and use

The CMS needs an RDBMS and a mechanism to link the CMS components to the same. The purpose of the RDBMS for the CMS is two-fold:

- To regularly save the current state, and named standard states; this allows restoring of the system to the last known known state in the event of a loss of power or CMS system crash, and also the ability to automatically set the system to any known standard state without having to command it manually.
- To save the configuration parameters parametrizing the CMS components with constants, limits, scaling factors and the like. These can be loaded at start up; or by triggering a command into XML files to allow faster access and the same can be modified offline to allow for change in these system parameters.

The interface between the database and the CMS should be achieved through loader modules that connect to the database using ODBC and generate the needed XML files. The interaction is schematically represented in the figure below**.** MySql is an open source database which can be potentially used for this purpose.

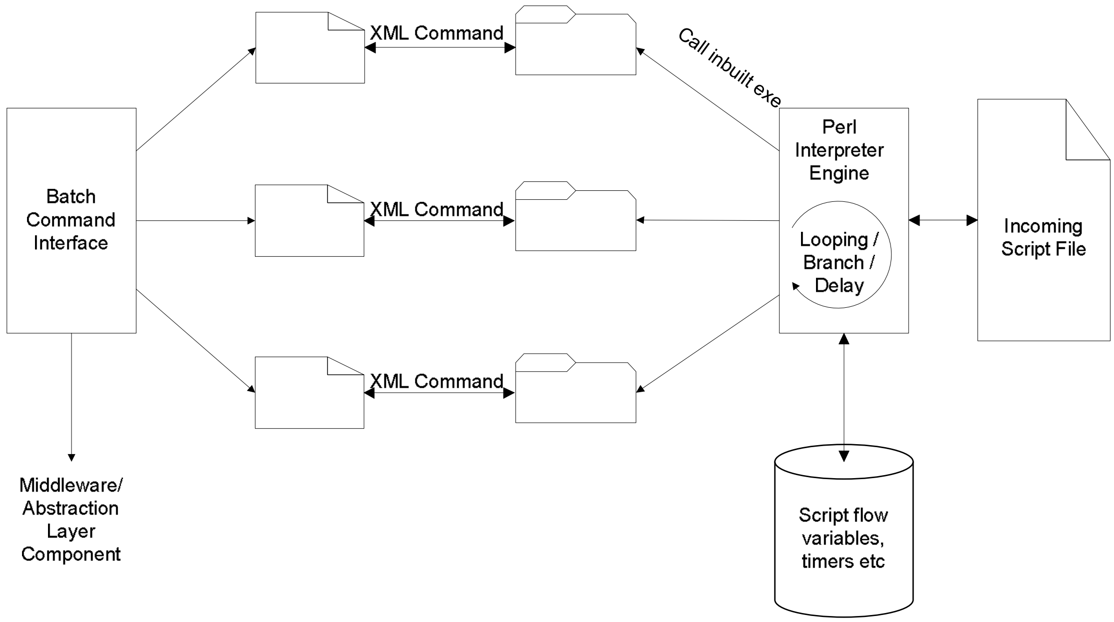


5.11 Support for Batch Mode / Scripting

To support batch mode it is proposed that CMS should incorporate a Perl interpreter engine that can execute scripts where flow control / looping is achieved through Perl syntax and the control actions are triggered through the command set which are interpreted by the Perl engine as system commands calling built in executables and passed to the command environment. A schematic representation of the process is depicted in the figure below.

Although there are recent developments in scripting languages that may also meet the needs, it is felt that Perl offers all the features needed from the batch mode engine and it would be fairly easy to learn for anyone reasonably familiar with C and shell scripting.

The batch mode implementation should also provide some standard template scripts for typical tasks with standard loops, delays etc. which can be modified easily with an intuitive user interface without detailed Perl knowledge.



Batch mode implementation

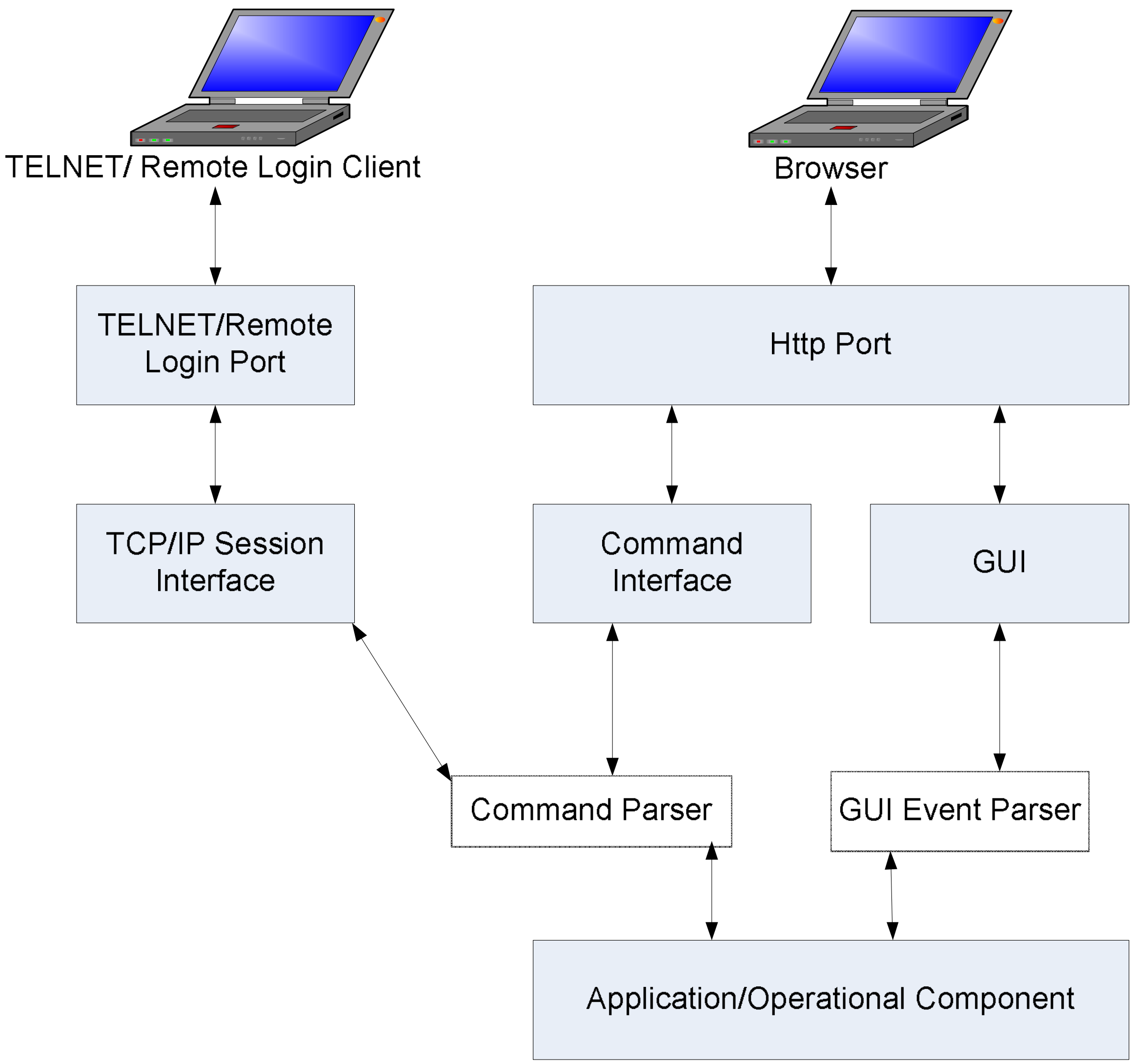
5.12 Help Module

The help module links to CHM files, FAQs and existing manuals and cookbooks. The updated versions of most of these would be stored on other servers in NCRA / IUCAA / CMS, therefore the help module in the CMS should simply implement a linking mechanism.

5.13 Provision for direct TCP/IP session access

Viewing and controlling the CMS using the UI (including the command line which is actually part of the UI) are accessed via the http protocol and reach a web-server. It is a requirement that in addition to this interface, there should be a possibility of directly accessing the CMS executable commands over a TCP/IP session (using rlogin or telnet for example) which bypasses the Web server. A user choosing this mode would view the command prompt on the screen. All the executable commands exposed by the CMS will be available based on user privilege, and there will be no intervention of the GUI. A schematic view of this access is provided in the figure below

TCP/IP Session Remote Access



6 Baseline requirements for the CMS system

6.1 Purpose and outline of this section

The architecture and the interface mechanism described in the previous two sections will largely allow any hardware module to be interfaced to CMS and its needed control and status elements represented in the operator or engineering UI. There is also a significant flexibility changing of the fields on the screens and the number and types of data elements represented. Yet the first implementation of the CMS and XML file definition must incorporate the baseline screens and data type requirements. The purpose of this section is to describe these initial requirements.

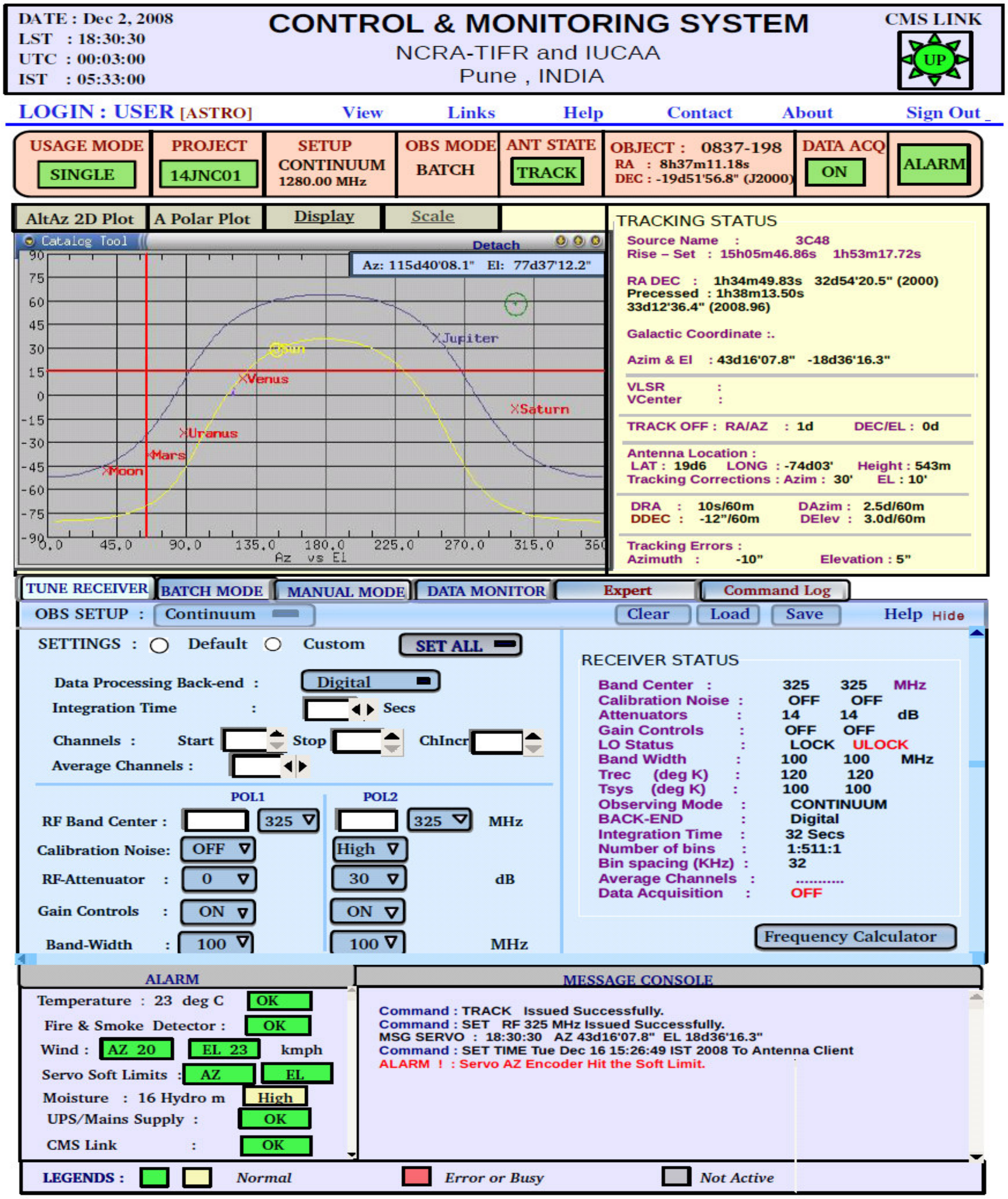
The next section 6.2 specifies the look and feel of the GUI, especially from a normal operational perspective for an operator / observational user.

Section 6.3 provides guidelines for implementing the engineering functionality with some sample GUI prototypes. Sections 6.4 to 6.7 provide guidelines for the needed baseline sub-system level functionality and GUI implementation for the optical and radio telescope. Following a high level description of the optical and radio telescope modules, tables are provided for each module which describe the control actions, commands, parameters to be provided to the hardware controllers together with the privilege levels as well as the monitor data / status messages. The details in these tables should be used for the creation of the XML message and config file structure within the CMS and correspondingly the initial control / monitor UI.

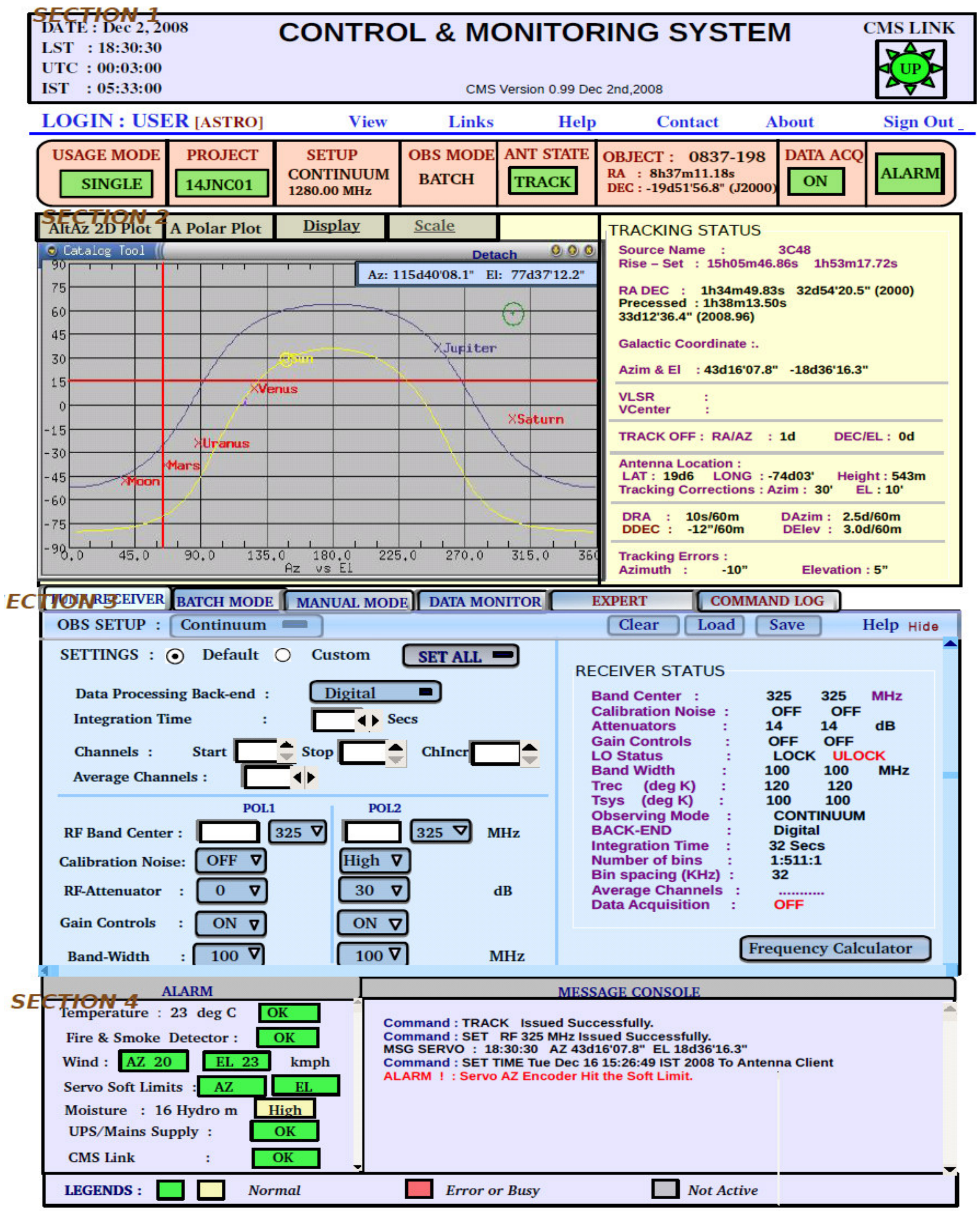
NOTE: All the GUI prototypes depicted further are representative; the actual look and feel of the individual elements and the screen organization is expected to be slightly different when implemented in a Java Web framework.

6.2 Baseline GUI specification from an operational /observational perspective

The IUCAA / GMRT team has suggested the following as the baseline prototype requirements from the GUI:



This GUI is divided into four sections as follows:



The explanation is given section-wise as follows:

Section 1 : HEADER DISPLAY

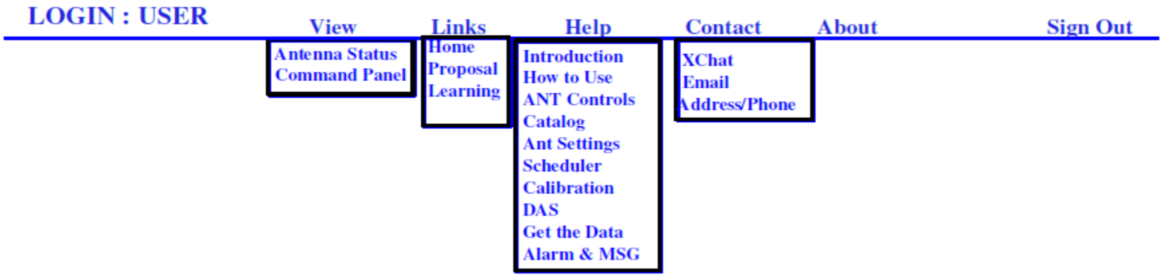
Field 1 : Time : Authentication : To all user
Aim : This field reflects ONLINE Real time of CMS
LST : Local Sidereal Time, IST – Indian Standard Time

DATE : Dec 2, 2008
LST : 18:30:30
IST : 03:33:00

Field 2 : CMS LINK : Authentication : To all user
Aim : This indicates a link status between CMS Server which controls the antenna and client application console.
Status :
Link enable : color – GREEN : "UP"
Link disable : color – RED : "DN" (Down).

CMS LINK

Section 1.1 : Task Bar : Authentication : To Expert , Prime User Astronomer
Aim : Describe which category of user logged in CMS Console. Help the user to introduce, get information in emergency and learn about how to use NCRA 15m Radio Telescope or Optical telescope .
Task bar should Pop-up Menu driven list by clicking on one of the menu from task bar.
About : CMS Software version,Brief Introduction, Equipment required Platform and libraries and Credits.



Section 1.2 : Front Header Display : Authentication : Expert , Prime User Astronomer, Engineers and general viewers

Aim : Display the telescope activity and status at a one glance.

Field 1 : USAGE MODE : SINGLE : color – GREEN : Telescope being used remotely for scientific project/observation. In this mode Expert only can login. Other users who already have logged in will be informed that 'Expert has logged in'. An expert can view other logged-in users' status. An expert can log-out other users also. In such as a case other users will be informed that an expert has taken a control and they cannot perform any operation till expert finishes his/her task. An expert can log-in from local or remote location.
MULTI : color – YELLOW : Telescope being used locally from WAN by NCRA STAFF. The users' list can appear in this as a drop-down. The number of users and their privileges are configurable.

Field 2 : PROJECT : Scientific usage : color – GREEN : This field identify the scientific observation being executed presently by displaying the PROJECT CODE (Maximum 8 characters). The project code is a unique key for each scientific observation/proposal.

Field 3 : SETUP : The setup field gives information about current configuration of receiver chain of the telescope.
The receiver chain of telescope can be configured in one of following default setups : (i) Continuum (ii) HI (iii) Pulsar (iv) Planetary (v) SUN (vi) MOON (vii) LINE etc.
This field also suffix one additional field about central frequency of the band (in MHz).

Field 4 : OBS MODE : This mode tells whether telescope being used under
BATCH mode : Automatic command execution from BATCH file.
MANUAL mode : Telescope being controlled manually by the astronomer/user.

Field 5: ANT STATE : Servo status of antenna :
(i) TRACK : color – GREEN : Antenna is tracking on the given target field.
(ii) POSN : color – YELLOW : Antenna is positioning toward given target field or given ` target for Elevation/Azimuth or both axis.
(iii) STOP : color – YELLOW : Antenna axis are in hold mode, i.e. Not tracking.
(iv) SCAN : color – GREEN : Antenna is scanning across the source.
(v) ERROR : Servo system error flag is ON, antenna not tracking.

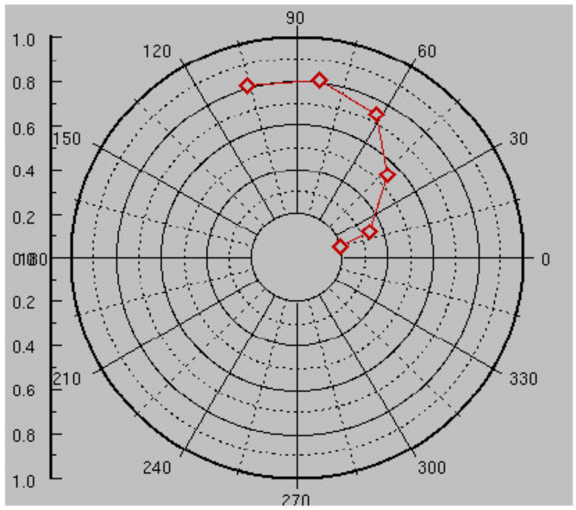
Field 6 : OBJECT : Display the current astronomical target source submitted by the user (In batch/manual mode) for the antenna tracking. Object field contains : source name (Maximum 12 characters) and equatorial coordinates (Right Ascension , Declination) with Epoch date.

Field 7 : DATA ACQ : This field gives information whether astronomical data is being recorded or not.
ON : color – GREEN : Astronomical data recording/logging is ON.
OFF : color – RED : Astronomical data recording/logging is OFF

Field 8 : ALARM : This indicates over all status of enabled alarms of the telescope.
OK : color – GREEN : All alarm flags are off, i.e. Telescope status is normal.
Alarm : color – RED : Any one or multiple alarm is ON.

Section 2 : TRACKING STATUS OF ANTENNA

Section 2.1 : Astronomical source catalog display : Authentication : Can be viewed by all category of users.
Aim : Graphically view the status of target fields and actual antenna tracking in user friendly manner.
The default catalog sources should be automatically updated in plot, also users can add their own catalogs. In case of optical telescope DS9 will be used. DS9 is an astronomical imaging and data visualization application.
Fields-1 AltAz display : shows the Cartesian co-ordinates (Elevation Vs Azimuth) for all target astronomical sources from catalog and position of antenna (i.e. real time status of azimuth and elevation of antenna).
Fields-2 A Polar plot : Shows a real-time positions of astronomical sources and Azimuth and Elevation of antenna.



A Polar plot : An Example of Polar Plot where Vertical Scale Can be Elevation from 0 to 90 deg and Azimuth from 0 deg (North) to 360 deg in astronomical coordinate..

Fields-3 Display : Display Should have following buttons for menu and Legends On off.

#	BUTTON TYPE	BUTTON NAME :	ARGUMENTS :	ACTION
Button 1 :	Normal Button : (with file-browser on click)	Display Catalog :	Name of Catalog file	: Astronomical source from the given : Catalog should be added to the graph.
Button 2 :	menu Driven :	Scale :	RA Vs DEC Altitude Vs Azimuth Galactic Latitude Vs Longitude	: Plot the X-Y Coordinates In a given Scale
Button 3 :	Radio Buttons :	Set Grid :	ON/Off	: Grid On Plot Becomes Visible/Invisible
Button4,5 :	Input Box with Spin-Button :		Ranges for X-Y Tics	: Draw given ticks for X-Y/polar plot
Buttons :	Check Boxes :	Legends :	Alt-Azimuth of Antenna Tracking offsets from Target Source Target Source name with RA-REC etc.	: Depending upon Check-Box : Selected or Not Legends on graph : will become ON/OFF

Fields-4 Scale : User Should able to scale the graphics like selecting ranges for X-Y axis, Zoom into it.

Section 2.2 TRACKING STATUS panel :Authentication : Can be viewed by all category of users.
This Panel shows a real time display of antenna status, and brief the details of antenna tracking and current astronomical object tracked by antenna.
Aim : User should convinced from this display that antenna is tracking right source with given correct precessed coordinates to the present epoch date. In addition , display gives useful information about antenna location, galactic latitude and longitude of the object, tracking error, loaded corrections for antenna tracking etc.

Section 3 :OBSERVING PANEL

***Aim :** The GUI of this section used to control, set and monitor the antenna systems' , astronomical observation and data collection. For this purpose the specific functionality are grouped under each tab panel which are described under following subsections with authenticity of users.*
Using this panel a general astronomer can use default setup as well as an experienced astronomer should able to choose specific custom settings to setup telescope receiver chain and controls of antenna tracking.

Section 3.1 TAB : TUNE RECEIVER : Authentication : Prime Astronomer and Expert

***AIM :** This tab panel mainly designed to setup the telescope receiver chain.*
***Authentication :** Astronomer-I, Expert.*

Tools bar contains two kind of setup- 'radio buttons' - one is DEFAULT and other is CUSTOM.
(i) DEFAULT setup : Depending upon 'OBSERVING SETUP' menu button selection ("Continuum", "Spectral-line (HI)" , "Planetary" , "SUN/MOON", and "pulsar" observation), receiver chain menus should be automatically setup to the defaults; There will be one default SETUP file for each Observing setup.
Note : User is not allowed to change any settings in default setup.
(ii) In CUSTOM setup, user can tune the coarse desired settings by choosing various given option and input buttons for various parameters (like back-end, channel, data integration selection and receiver parameter selection like Band-Width, gain controls, Attenuators for both polarization of antenna.

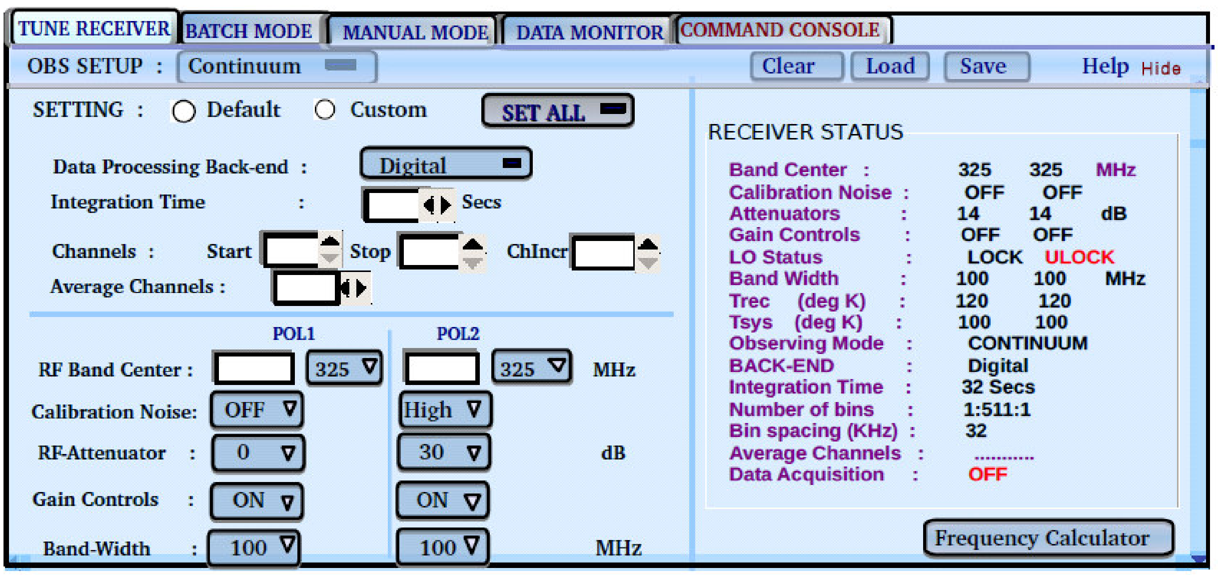
(iii) CLEAR – Reset all menus in panel to default setup.
SAVE – Save antenna settings in a user file with '<filename>.SETUP' extension (A standard file browser can be used for this),
LOAD – Load saved user's file '<filename>.SETUP' in the panel.

(iv) HELP – Information about various default setups, how to set antenna receiver chain etc. A separate window should pop-up on clicking help menu.

(v) Hide – A complete tab panel should be minimized by clicking upon Hide button.

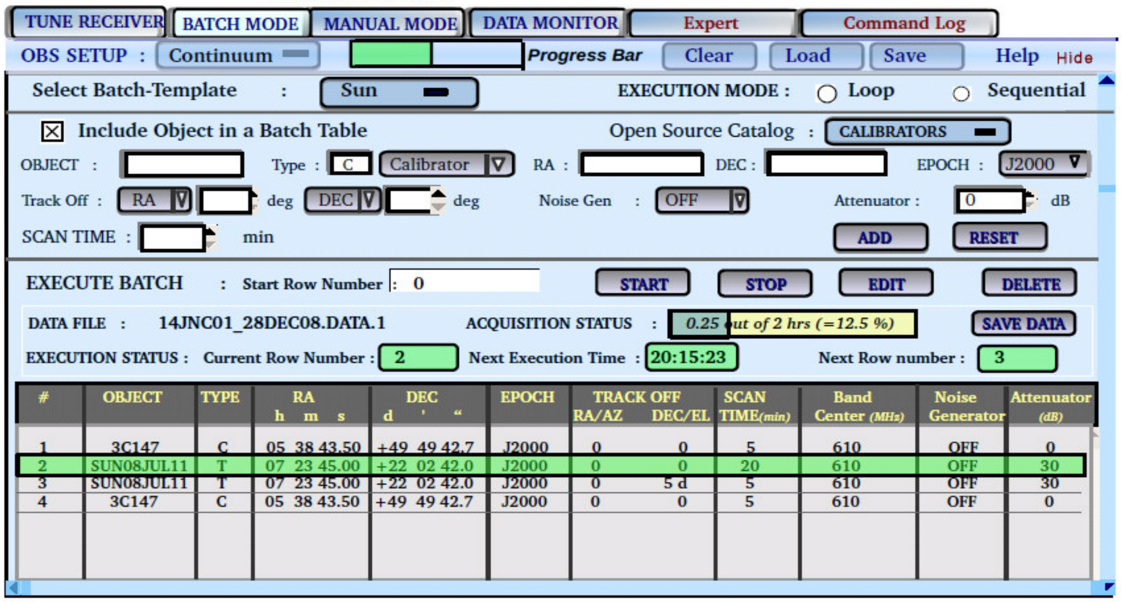
***SET** : Drop-down : 'action' menu button : Using this button either user can choose 'SET ALL' settings or one can select to configure specific system like SET-RF, SET-BACKEND, SET-GAIN CONTROLS etc.*

***RECEIVER STATUS** : This panel inform about state of settings configured at the receiver chain of antenna.*



Section 3.2 TAB : BATCH MODE : Authentication : Prime Astronomer and Expert.

AIM *: This tab panel mainly designed to execute astronomical observation through a batch file.*
Authentication : Expert, Astronomer-I
Comment : (i) There can be number of batch-file templates specific to each observational setup.
(ii) During the execution of batch file user can not control the telescope manually unless 'STOP' Batch execution button is clicked.
i.e. On the contrary, BATCH MODE should disabled when data acquisition in MANUAL MODE start.
(iii) In this mode, astronomical data acquisition is done automatically once the execution BATCH-TABLE starts



#	OBJECT	TYPE	RA h m s	DEC d ' "	EPOCH	TRACK OFF RA/AZ	TRACK OFF DEC/EL	SCAN TIME(min)	Band Center (MHz)	Noise Generator	Attenuator (dB)
1	3C147	C	05 38 43.50	+49 49 42.7	J2000	0	0	5	610	OFF	0
2	SUN08JUL11	T	07 23 45.00	+22 02 42.0	J2000	0	0	20	610	OFF	30
3	SUN08JUL11	T	07 23 45.00	+22 02 42.0	J2000	0	5 d	5	610	OFF	30
4	3C147	C	05 38 43.50	+49 49 42.7	J2000	0	0	5	610	OFF	0

The buttons on tool-bar are described with their functionality as follows :

(i) Clear : Blank 'Batch Table', and all fields relating to OBJECT.

(ii) Load : Using a file browser, load user's Batch File '<filename.BATCH>' in a Batch table.

(iii) Save : Save rows from a Batch-table in a user specified file-name with '.BATCH' extension.

(iv) HELP – Information about various default batch file, how to load batch-table, start batch execution, data acquisition status, batch execution status, and batch table etc.

(v) Hide – A complete tab panel (section-3) should be minimized by clicking upon Hide button.

There are four main subsections in 'BATCH MODE' tab panel :

Section 3.2.1 : This sub-section selects a batch file according to observational Setup or select a 'custom' menu in which user adds astronomical object and observing plan in the Batch-table.

This section contains two menus :
(a) Select Batch-Template :
Button Type : Menu-Driven Button.
Option : "SUN", " JUPITER", "MOON", "HI-OBS", "Line", " PULSAR", " CUSTOM".
Action : Default predefined template file for SUN, Planet or HI observation should be automatically loaded in the Batch table. In CUSTOM mode, user can create new entries for BATCH-TABLE using the 'Include Object in a Batch Table' menu.
Progress Bar: This shows the progress of the issued command. From this, an astronomer/ an engineer can come to the status of issued command and he/she can decide for the further sequence of operation.
(b) EXECUTION MODE : Only one of the two mode for execution can be selected by radio buttons :
(a) Loop : Rows in a Batc h table will be continuously executed in a loop.
(b) Sequence : Batch execution will be over after the executing the last row from the Batch-Table.

Section 3.2.2 : This sub-section of GUI helps to include a astronomical source in the BATCH-TABLE for batch execution.

(i) The Check-Button 'Include Object in a Batch Table' Enables all the field of Astronomical Source (OBJECT). With the help of catalog menu-driven button user can load these fields to default by selecting a astronomical source from specific source catalog (like catalogs of calibrator, pulsar, strong radio sources etc.).

(ii) The Object field contains :

(a) Object name

(b) Object Type : One character identification code can be assigned to the astronomical object which can be used for post-analysis of the data (e.g. T-Target, C-Calibrator, P-Pulsar, H- HI sources near galactic latitude).

(c) RA , DEC, EPOCH : Right ascension, Declination and Epoch date of Object

(d) Track Off : One can track the antenna off from the astronomical source (OBJECT) by giving offsets in RA/Azimuth and DEC/Elevation.

(e) Noise Generator : User Can switch NG while executing the batch file for calibration purpose. Noise generator can have specific values like Extra-High, High, Medium and Low cal.

(f) Attenuators : While moving on strong source, signal attenuator can be put to avoid the saturation of the antenna signals.

(e) SCAN-TIME : This field specify the total time for data acquisition on given astronomical object.

(iii) Normal Buttons : (a) ADD : Once the valid data entered in all the fields of the astronomical object, one can submit the object to a Batch-Table.

(b) RESET : By clicking this button user can reset and clear values in all field of object.

Section 3.2.3 : This sub-section of GUI mainly concern about execution of Batch-Table, astronomical/scientific data acquisition and gives information about the status of same. The buttons along with their functionality are described as follows :

(i) EXECUTE BATCH : This row contains buttons and menus which can used to execute the Batch-Table.

(a) Start Row Number : Input box : Default value : 0. When 'Start' button clicked, batch-execution will start from the row number given in this field.

(b) Normal button : START : Start the execution of 'Batch-Table'. Once START button clicked, the astronomical data collection on given source will start automatically in a data file named with <PROJECT-CODE_DATE>.DAT extension. The acquisition status bar will become active and will indicate the status of data collection.

(c) Normal button : STOP : Stop the execution of 'Batch-Table' . The astronomical data acquisition should also stopped once 'STOP' button clicked.

Note : Unless STOP button clicked and batch-execution stopped, one can not EDIT or DELETE a rows from BATCH-TABLE.

(d) Normal button : EDIT : By double clicking the row in a batch table, or single click on row of batch table and EDIT button, one should able to modify the field in BATCH-TABLE.

(e) Normal button : DELETE : Delete a selected row/rows of Batch-Table. A row of batch table can be selected by single click on row.

(ii) STATUS : Batch-Table execution and astronomical data acquisition.

In this group box, user will get information about

(a) DATA FILE : The name of data file in which observational data on source will be acquired. In BATCH-MODE, the name of data file will be decided by PROJECT-CODE, Observing DATE (e.g. 14JNC01_28DEC08.DAT).

(b) ACQUISITION STATUS : progress bar : This gives information about how much percentage data (out of Total time allocated by SCAN TIME of each source in a BATCH-TABLE) has been acquired.

(c) SAVE DATA : User should able to save the astronomical data on their PC at any time, whenever he/she want to confirm/post-analyze it.

(d) Execution Status : This row gives information about status of execution of BATCH-TABLE by providing information on current row number under execution, time of next-row execution and next-row to be executed.

(iii) BATCH-TABLE : Since all the information described under section 3.2 gets filled in BATCH-TABLE, BATCH-TABLE should be self explanatory.

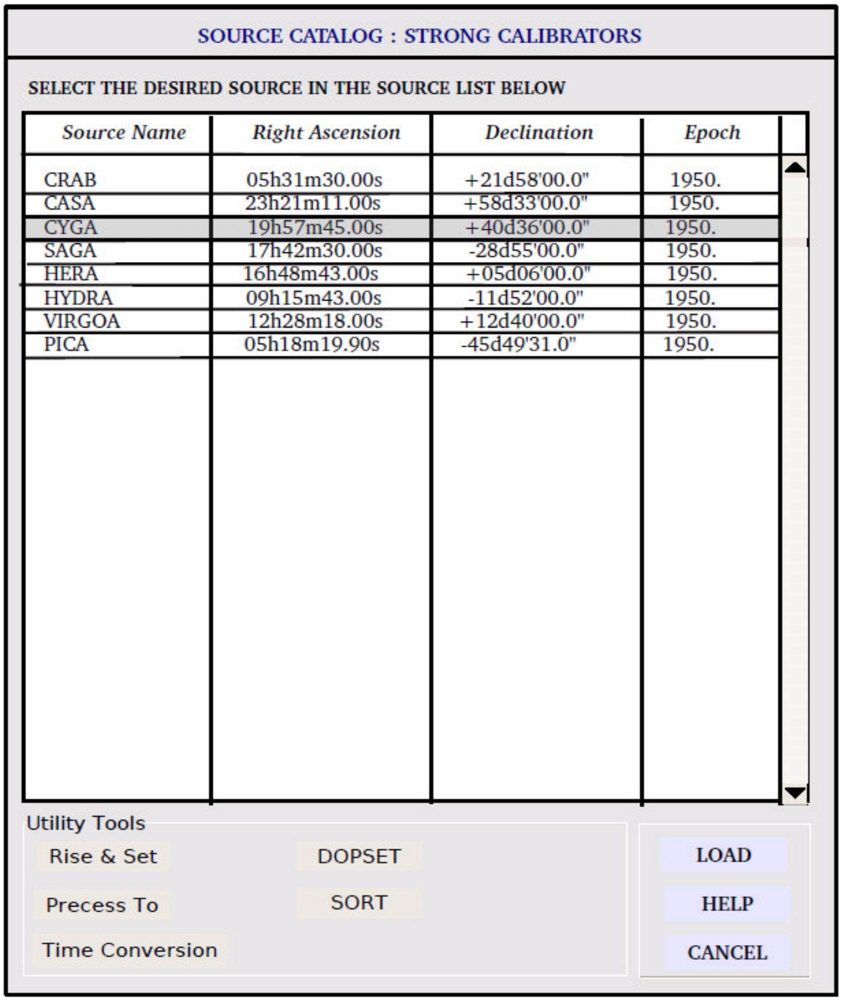
(a) The GREEN color of the row indicates the current BATCH-TABLE command execution.

(b) If the BATCH-TABLE execution is stopped by STOP button, the single click on any row should make it faint gray so that it can be either edited or deleted. Otherwise it is not allowed to edit/delete a row in the BATCH-TABLE if the BATCH execution is going on.

Section 3.2.4 : Source catalog panel : This new window will pop-up upon clicking the 'Open Source Catalog' menu driven button of sub-section 3.2.1 GUI. User can select a variety of catalog using the menu driven button, the catalog types can be "Calibrators" , "Strong Radio Sources", "HI Sources", "Pulsars", "Planetary Object", "Custom". If "Custom" menu selected, user can load his/her own catalog to 'Source catalog panel' with the help of File-Browser.

Aim : With the help of source catalog, user can select a sources from specific catalog of his/her interest.

Action : 'LOAD' button on panel should fill the astronomical source with it's type , equatorial co-ordinates and other details in fields of section 3.2.2 GUI or directly to a BATCH-TABLE where user can specify SCAN-TIME for selected object and other parameters details if needed like NOISE GENERATOR etc..



SOURCE CATALOG : STRONG CALIBRATORS

SELECT THE DESIRED SOURCE IN THE SOURCE LIST BELOW

Source Name	Right Ascension	Declination	Epoch
CRAB	05h31m30.00s	+21d58'00.0"	1950.
CASA	23h21m11.00s	+58d33'00.0"	1950.
CYGA	19h57m45.00s	+40d36'00.0"	1950.
SAGA	17h42m30.00s	-28d55'00.0"	1950.
HERA	16h48m43.00s	+05d06'00.0"	1950.
HYDRA	09h15m43.00s	-11d52'00.0"	1950.
VIRGOA	12h28m18.00s	+12d40'00.0"	1950.
PICA	05h18m19.90s	-45d49'31.0"	1950.

Utility Tools

Rise & Set DOPSET

Precess To SORT

Time Conversion

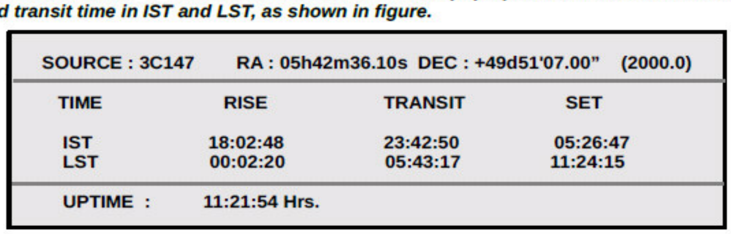
LOAD

HELP

CANCEL

Source Catalog contains a few utility tools which can run on the selected astronomical source from source catalog panel (On Single click – source should get selected from catalog with indication of gray box e.g. In catalog panel figure, 'CYGA' is selected). The Utility tools are described below with their functionality.

(a) Normal button : Rise & Set : The click on this button should pop-up a new window which can show source rise, set and transit time in IST and LST, as shown in figure.

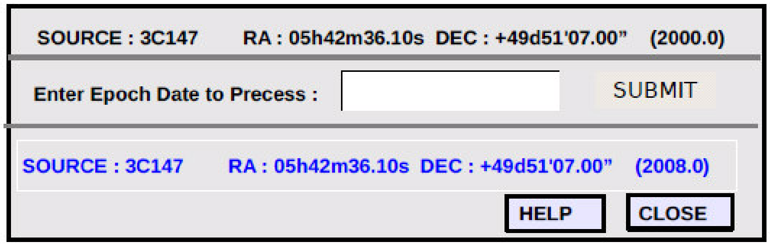


SOURCE : 3C147 RA : 05h42m36.10s DEC : +49d51'07.00" (2000.0)

TIME	RISE	TRANSIT	SET
IST	18:02:48	23:42:50	05:26:47
LST	00:02:20	05:43:17	11:24:15

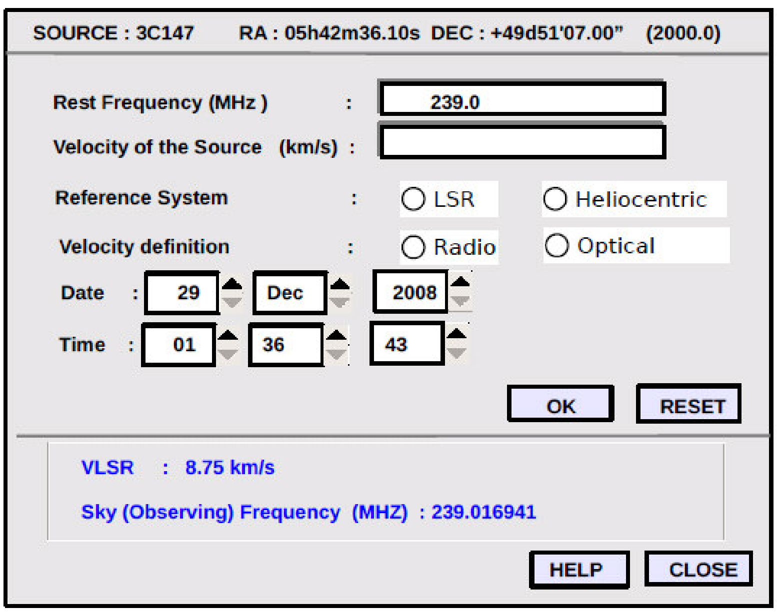
UPTIME : 11:21:54 Hrs.

(b) *Normal button : Precess To : This button precess the source co-ordinates (RA, DEC) to the present epoch or any desired epoch.*



(c) *Normal button : Time Conversion : This button help user to convert time from one format to other.*
FORMATS are :
IST : Indian Standard Time
LST : Local Sidereal Time
UT : Universal Time
GMT : Greenwich Meridian time
JULIAN DATE : A continuous count of days and fractions since noon Universal Time on January 1, 4713 BCE) : cf http://aa.usno.navy.mil/data/docs/JulianDate.php for further informations.

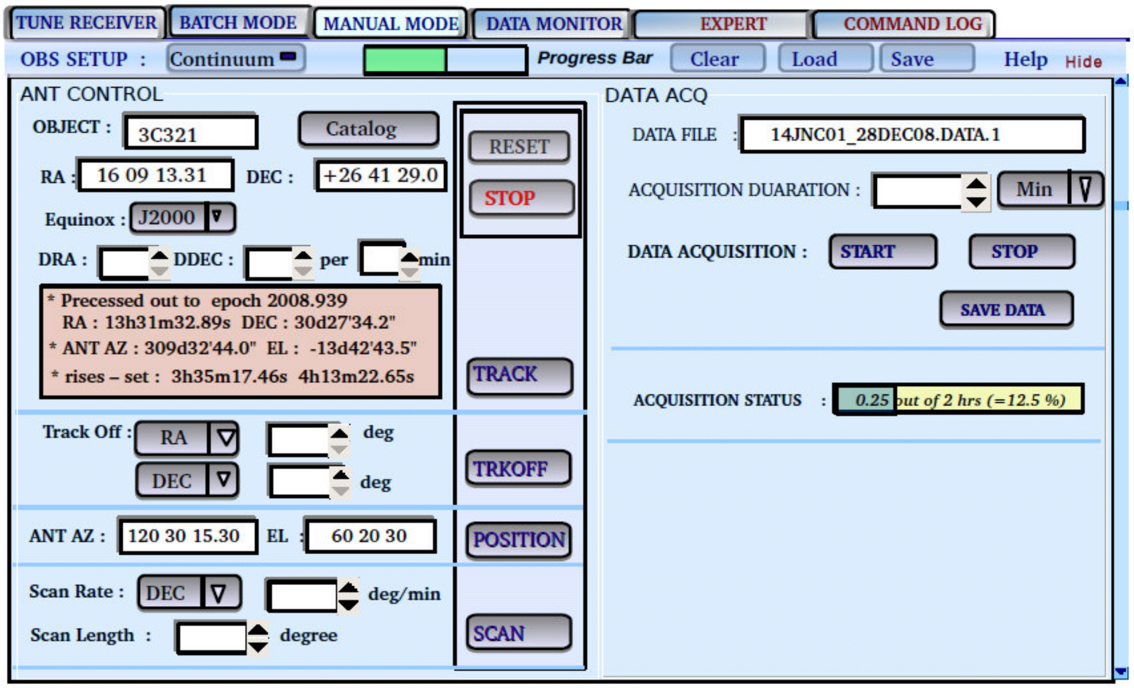
(d) *Normal button : DOPSET : Spectral Line observation requires some care for setting the observing frequency because due to Doppler tracking effect actual spectral-line frequency shift to other value. Hence observing frequency (sky frequency) of the telescope need to set to the changed value.*



(e) *Normal button : SORT : SORT the Source Catalog according to Right Ascension , Declination, and Total time above the horizon etc.*

Section 3.3 TAB : MANUAL MODE : Authentication : Prime Astronomer and Expert.

AIM : This tab panel designed to perform an astronomical observation, calibration using manual control . Also, user can become familiar to the system by rotating antenna, monitoring and acquiring the test data .
Authentication : Astronomer-I, Expert, Servo Engineer.
Comment : (i) Data logging has to be done manually.
(ii) During the execution of batch file user can not control the telescope manually unless 'STOP' Batch execution button is clicked.
i.e. MANUAL MODE Option should not be selectable (gray) when batch mode acquisition is going on.
Progress Bar: This shows the progress of the issued command. From this, an astronomer/ an engineer can come to the status of issued command and he/she can decide for the further sequence of operation.



The buttons on tool-bar are described with their functionality as follows :

(i) Clear : Clear all fields in ANT CONTROL and DATA ACQ Frame.
Note : This button should be disabled when the data acquisition/tracking is in execution.

(ii) Load : Can be disabled (gray) ?or Using a file browser, load user's control File '<filename.CTRL>' ..

(iii) Save : Can be disabled (gray) ?or Save all control and data acquisition variables in <filename.CTRL>' File..

(iv) HELP – Information about Antenna Control buttons with how to submit and track source, TRKOFF, POSITION, SCAN and STOP button. Also How to acquire and save astronomical data.
(v) Hide – A complete section-3 tab panel should be minimized by clicking upon Hide button.

There are two main subsections in 'MANUAL MODE' tab panel :

Section 3.3.1 : This sub-section allow user to control telescope manually, as well as get a familiarity of telescope controls.

This section contains five main control buttons :
(a) TRACK :
Button Type : Normal Button.
Action : Track the antenna on OBJECT with given parameters of RA, DEC, and EPOCH (1950, 2000.0). With the help CATALOG button user can get the Source catalog (refer section 3.2.4) and select a source for OBJECT, RA, DEC, EPOCH fields. Automatically user gets the information on source rise, set, current Alt-Azimuth and Precessed co-ordinates for the selected source in a faint-red box.
The RESET button should clear all fields of the OBJECT.
(b) TRKOFF :
Button Type : Normal button
Action : Allow user to track the antenna on the source with customized offsets either in "RA/AZIMUTH" and "DEC/ELEVATION" using two menu-driven button.

(c) POSITION :

Button Type : Normal Button.

Action : Allow Astronomer/Expert to position the antenna at desired AZIMTUTH and ELEVATION by giving the antenna co-ordinates in deg:min:second for both axis.

(d) SCAN :

Button Type : Normal button

Action : Take a scan across the given source (OBJECT field with RA, DEC and epoch) in "RA" , "DEC" , "AZIMUTH" or "ELEVATION" with specified rate in degree/min. And total length of scan in degrees.

(e) STOP :

Button Type : Normal Button

Action : Stop the antenna tracking, positioning or scanning activity.

Section 3.3.2 : This sub-section allow user to acquire the astronomical data for specified duration, and save it on the local disk.

This section contains following buttons :

(a) DATA FILE :

Button Type : Input Button.

Action : User can specify here a name of the data file in which astronomical data will be collected.

(b) ACQUISITION DURATION :

Field : Input Spin Button and Menu driven button - "Hour", "Min", "Second".

Action : This field allow user to mention a duration of total time for astronomical data he/she needs.

(c) DATA ACQUISITION :

Button Type : Normal Buttons : START and STOP .

Action : Start button start the actual data acquisition for given duration. Once a data acquisition accomplished for a specified type, automatically data acquisition should stop. 'START' button should become gray during the astronomical data acquisition. User can interrupt the data acquisition using a 'STOP' Button.

(d) DATA ACQUISITION STATUS :

Button Type : Progress Bar.

Information : With the help of progress bar user will know the data acquisition status. Comment : During the data acquisition, BATCH-MODE should be disabled (Gray).

Section 3.4 TAB : DATA MONITOR : Authentication : Astronomer, Expert, Engineer, Viewer.

AIM *: Display the astronomical data using plots with various plotting input parameters given by user. In case of optical telescope DS9 application will execute the available data.*
Authentication : All types users can see data monitoring except general viewer using WAN.

On Data monitor tool bar there are symbolic icons and action buttons which are explained below :

(i) Plot – button type : Menu driven action button
Action : Using Plot option on tool bar, user can generate three types of displays :
"Chart recorder" - showing Real time display of Power Vs Time
"Spectral Line" - shows real time discrete spectrum plot as well as accumulated plot.
"Pulsar" - real time pulsar signal in time domain, current and folded profile and spectrum plot. (Note : "pulsar" plot is proposed only, yet to be finalized.

(ii) Start – Icon Button
Action – Start the real time display when user press this icon.

(iii) Stop – Icon Button
Action – Stop the real time display when user press this icon.

(iv) Set preferences – Icon Button
Action – Open a preference dialog where user can set the plotting parameters relevant to individual types of astronomical display.

(v) Help – Normal button
Action – Briefs the general idea about each plot with their purpose and how to use.

(vi) Hide – User can hide the section-3 tabs.

All types of plot displays share a common tool bar which is described as follows :

(i) Scale – Icon Button
Action - User can zoom in/Out or change the scale of axis, steps etc.

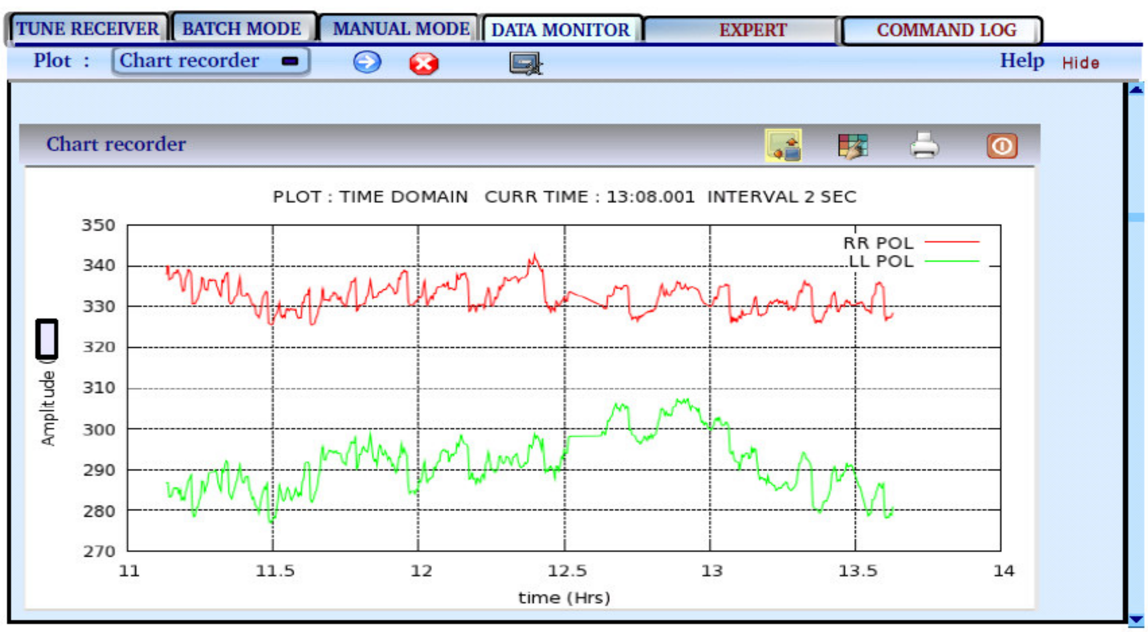
(ii) Set color – Icon Button
Action – Set a color of lines , grid or legend etc.

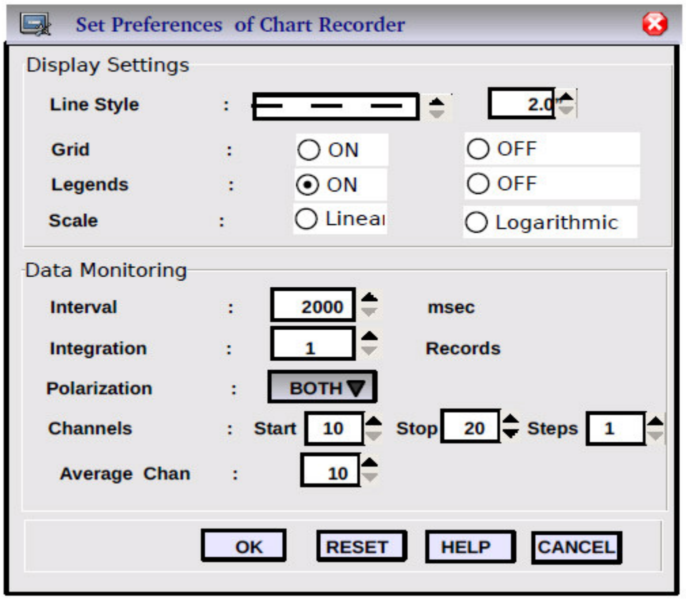
(iii) Print - Icon Button
Action – Print a landscape or portrait types of plot on user's printer.

(iv) Quit - Icon Button *Action – Quit the plot.*

The three types of display and preferences/settings dialog panel with respect to each display are described under three sections.

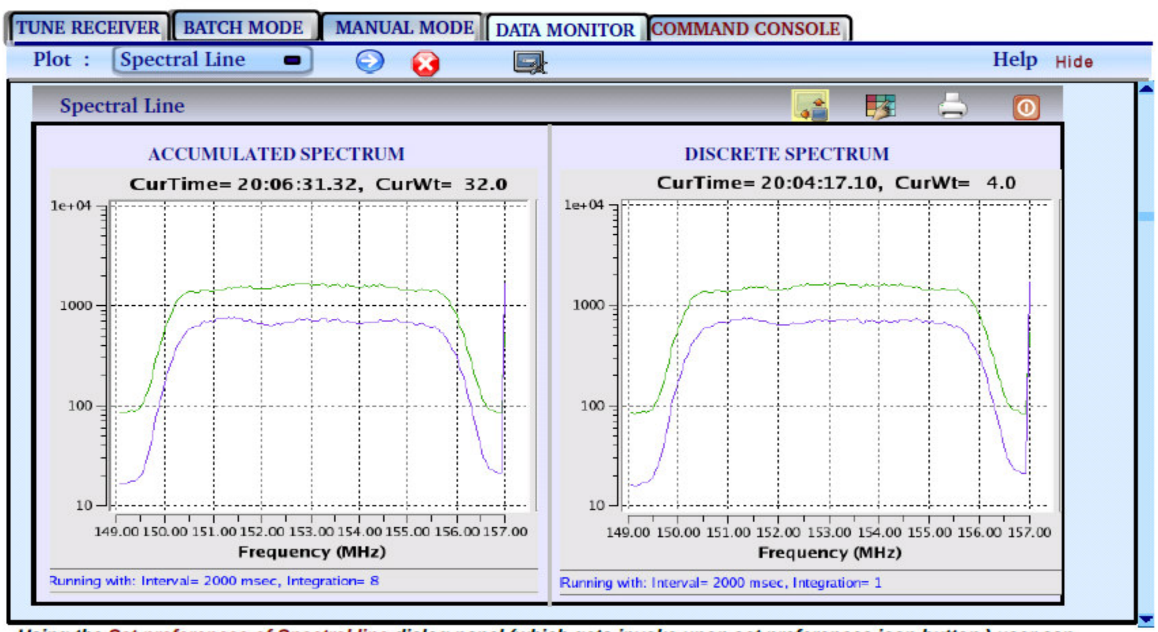
Section 3.4.1 : ***Chart recorder*** *: This shows real time plot of relative power of the signal received Vs time (Hrs) with informative title bar about update time interval, current record time.*



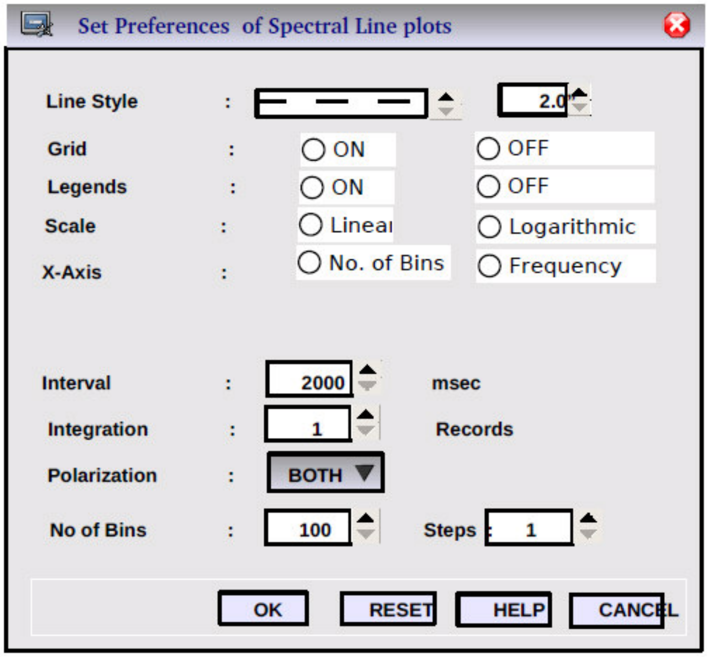


Using the Set preferences of Chart recorder dialog panel (which gets invoke upon set preferences icon button) user can customize chart recorder display along with Data monitoring parameters like update interval, integration, polarization (“BOTH”, “RR”, “LL”) and number of channels to average.
Currently the data stored in case of GMRT is in FITS (Flexible Image Transport System) format. So whole file should be transmitted at a time. Currently in case of optical telescope changes can be sent regardless of update of whole file. However all this is configurable.

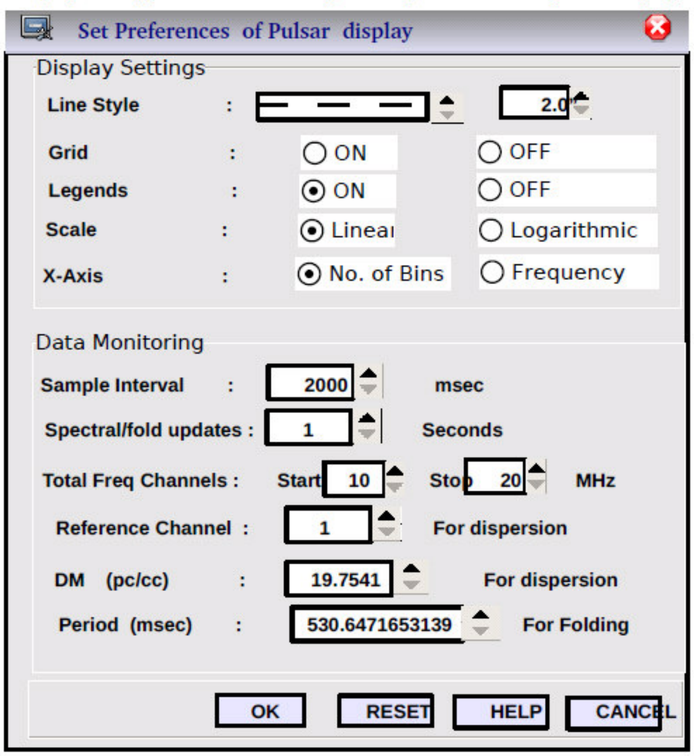
Section 3.4.2 : Spectral line display : ***This display contains two kind of spectrum, one is real time 'discrete spectrum' as it finishes the user input span in MHz and another is 'accumulated spectrum' which shows accumulated spectra since the selected observation begin. The title bar of each display gives important information like current time, update interval and total integration time. The y-axis is total power and X-axis can be either frequency or channel numbers depending upon the spectral line preferences settings.***



Using the Set preferences of Spectral line dialog panel (which gets invoke upon set preferences icon button) user can customize spectral line display along with Data monitoring parameters like update interval, integration, polarization (“BOTH”,
“RR”, “LL”) and Scale of X-axis (Frequency or Number of bins).



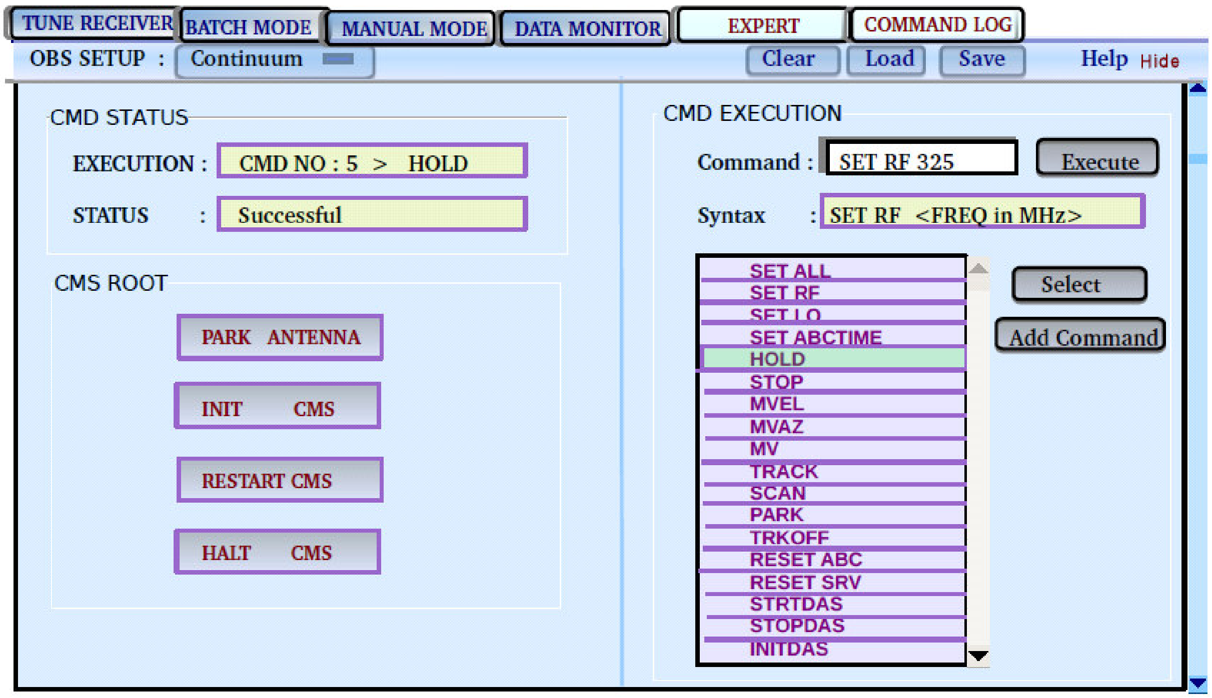
Section 3.4.2 : Pulsar display : This display contains chart recorder, current profile of the pulsar, folded/accumulated profile of pulsar and band shape spectrum. The pulsar visualization/monitoring is specifically depend upon sampling interval, Dispersion measure and period of pulsar at given frequency. These variables along with display settings can be done using the set preferences of pulsar display.



Section 3.5 TAB : Expert .

Aim : *This tab mainly used by an expert to give commands in text mode to set receiver settings or control the antenna. Also, expert can park the antenna using park button in case of emergency or antenna security related issues.*

Authentication : Expert only.



There are three main sub group in EXPER Ttab panel :

(i) CMD EXECUTION : *Expert can either issue a command to CMS master(PC) through the Command input box and using Execute button, or can select the desired command from the List Box using the select button. The command selected by "Select" button will appear in command input box with required syntax for the execution of that command in 'Syntax Box'. One can built a batch of commands and submit it to the list box using Add Command button.*
__Note :__ Execute button should validate the command syntax, and upon occurring of syntax error pop-up menu about command not correct should appear at user desk and command should automatically get canceled.

(ii) CMD STATUS : *This box will show the current command executed by Execute button and its status : "Successful", "Failed", "Time Out".*

(iii) CMS ROOT : *These are buttons which required either in emergency and critical cases which are related to security and CMS functioning of antenna. __This button should ask password/confirmation while executing the button.__*

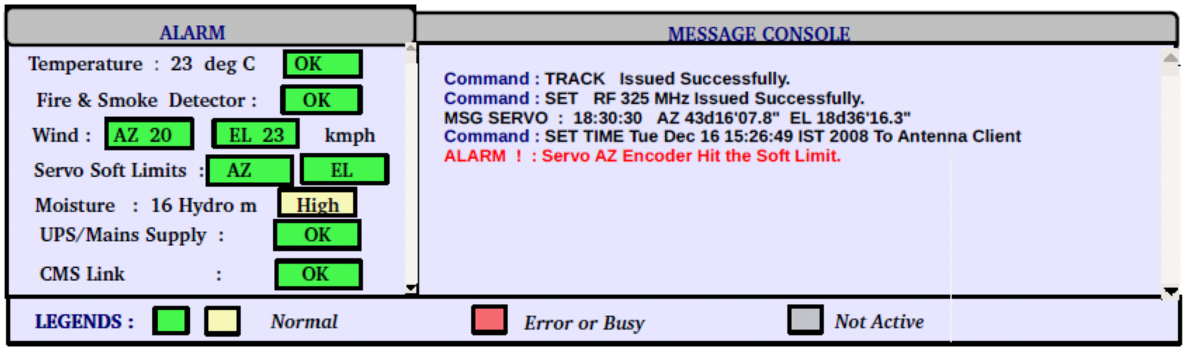
a. PARK ANTENNA : Stop the on-going activity (observation, etc.) and park antenna immediately.
b. INIT,RESTART,HALT CMS : These commands are mainly aim to initialize, restart or halting of CMS software (Computer?).

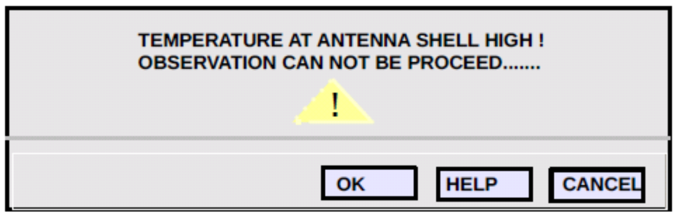
Section 3.6 TAB : COMMAND LOG .

Aim : *This tab mainly used by view the log. An expert can view logs of all users. Also, each user can view his/her log for the operations performed in particular time. This log can be saved in the form of a file.*

Authentication : Expert and astronomer.

Section 4 : Alarm and message console





Aim : *This section contains a display for critical alarms for the telescope and message console for user for detail information. This section should be common to all console of the telescope.*

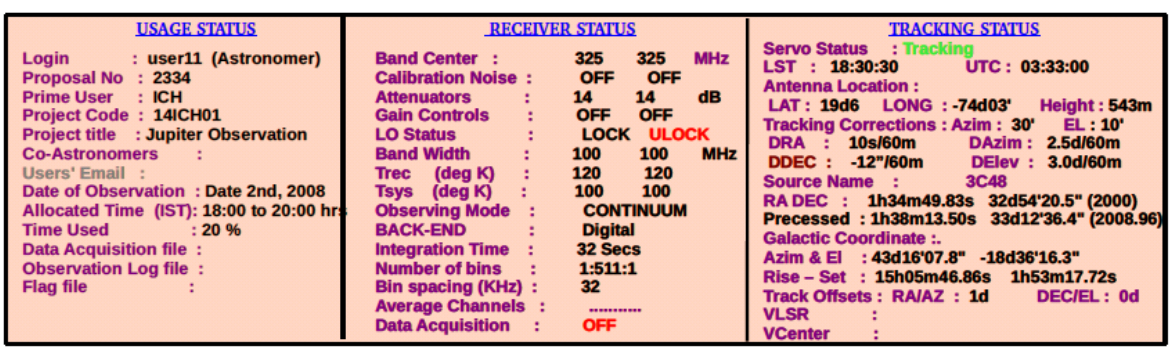
Authentication : *ALL users.*

Section 4.1 : ALARM : *This display contains critical alarms for the telescope, exceeding upon the set threshold value, it should pop-up a alarm indicator with buzzer and intimate a user about the interrupt the observing session.*

Comment : There can be a one file in which alarms and its threshold values can set for NORMAL (GREEN), Problem but can proceed the observation (YELLOW), and observation/telescope control can not be done (RED) .

Section 4.2 : MESSAGE CONSOLE : *Message console inform users about the state of telescope, i.e.*

a. command execution and its status (Success/fail/time out).
b. Status of servo computer and antenna tracking.
c. Alarm for exceeding the set threshold value, changing the status of systems.
d. Time out from various systems or link.



6.3 Engineering GUI

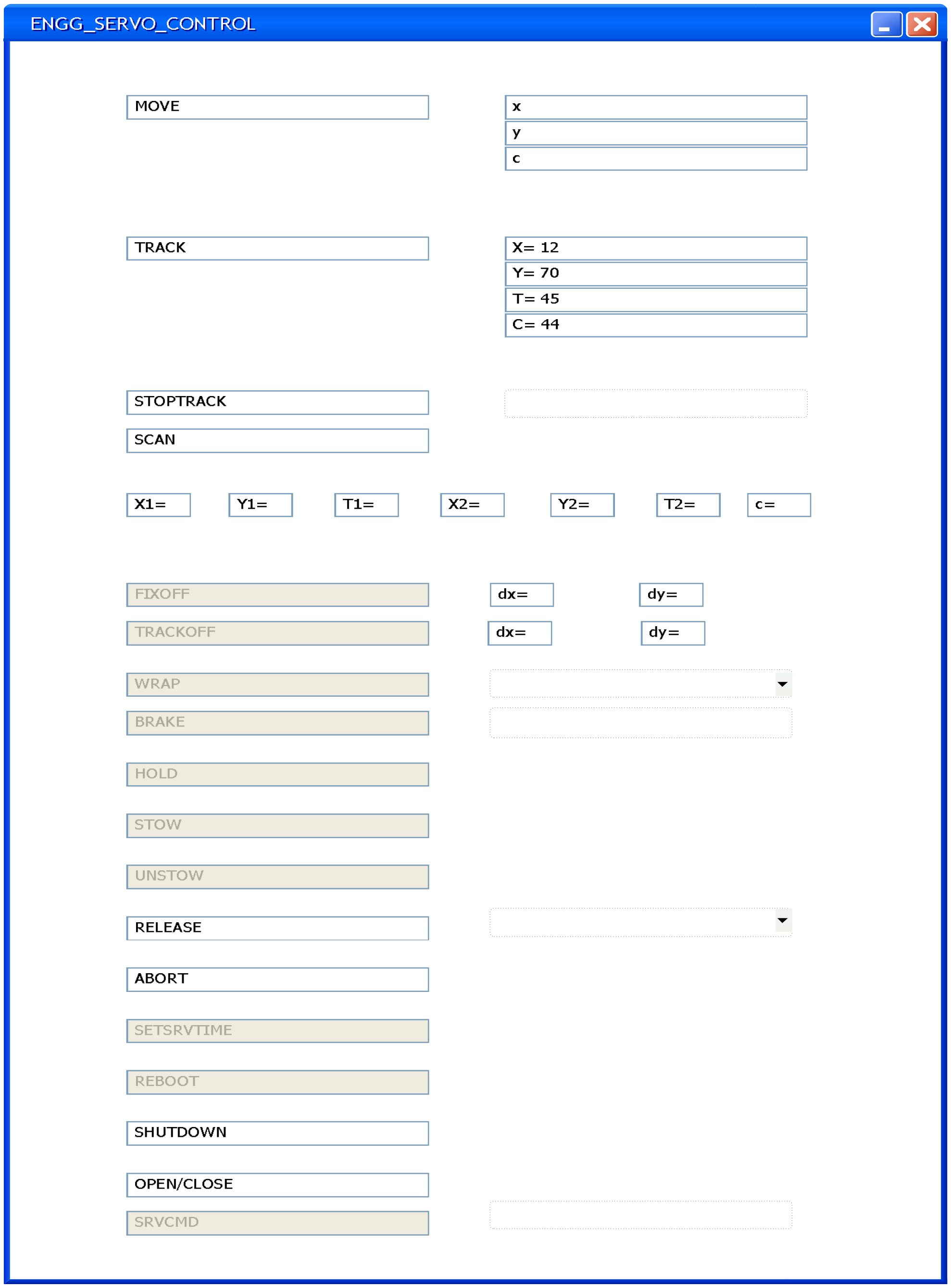
The Engineering GUIs are intended for the use of the telescope engineers for maintenance, troubleshooting or advanced configurations of a telescope sub-system which is beyond the purview of the operator or astronomer. Therefore, these will typically provide more lower level access than the user / operator GUI. Some of the expected ways in which the engineers may need to use the CMS are

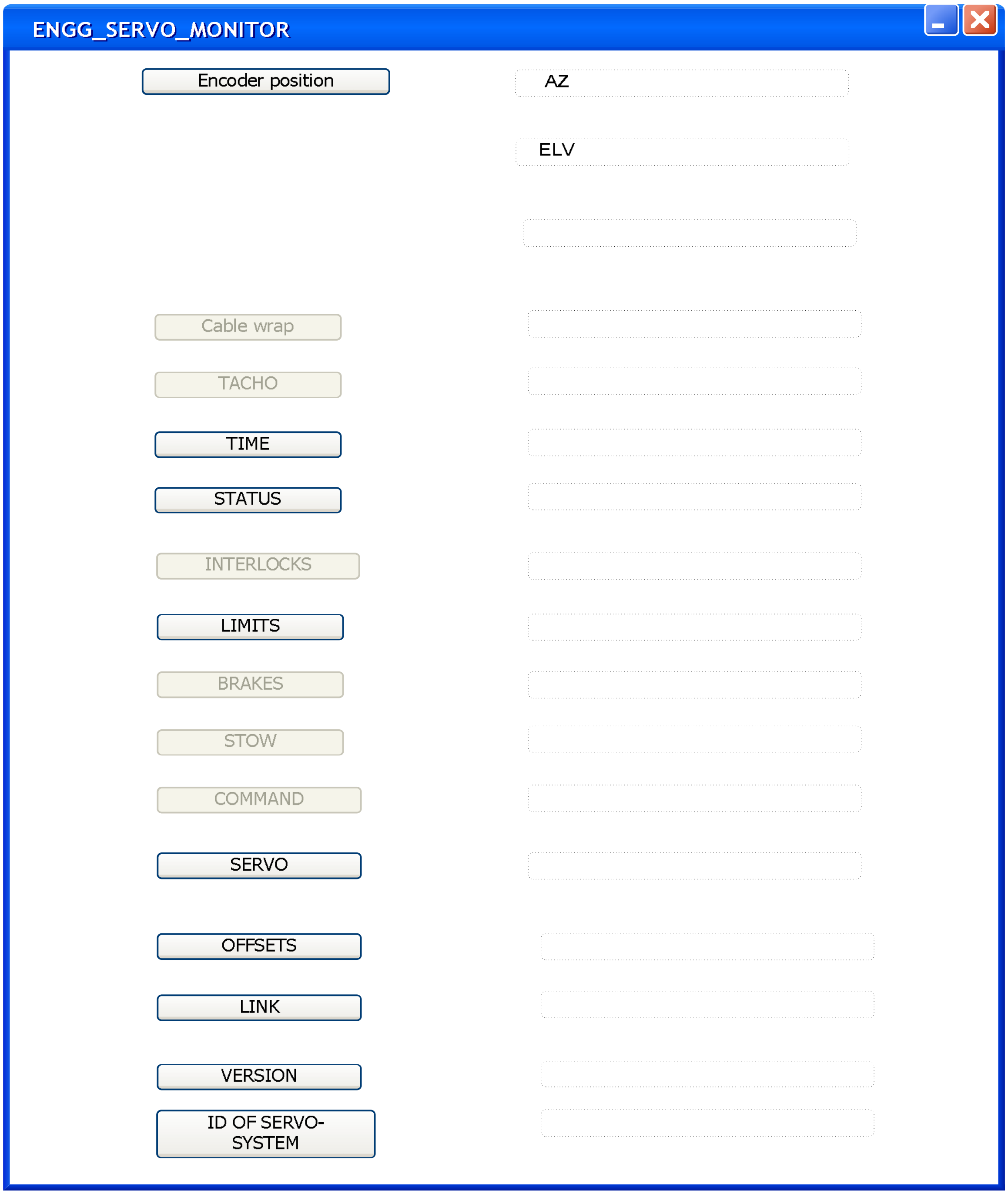
- Suspending / halting, shutting down or restarting a sub-system
- Reading configuration parameters or flags of a sub-system
- Setting or resetting configuration parameters or flags
- Sending a particular raw command sequence / message (typically ASCII) to the hardware
- Triggering pre-defined actions on the hardware through a button

These are indicative and the actual functionality exposed through the engineering GUIs are very sub-system specific. As a general specification, the Engineering GUI screens consist of various command buttons, status indicators – both binary and alpha numeric text boxes, test boxes or drop downs for raw commands. These controls should be achieved using the same principles described earlier: not hard coded but capable of being changed through simple configuration file changes. The Engineering GUI thus displays the commands / parameter values are entered through the GUI. After that, these are stored in the XML file and then sent to the particular subsystem.

For the baseline implementation, Sections 6.4 to 6.7 contain tables that provide the needed controls / information display etc. and access privilege levels, by sub-system for each telescope type. However, some level of confirmation would be needed during implementation as there could be changes for some of the sub-systems.

For illustrative purposes only, shown below are prototype representations for a typical Engineering GUI screen layout for servo sub-system for radio telescope. There are separate screens for control and monitor purposes. The gray field shows the restricted access to the user logged in and changing status during measurement.


ENGG_SERVO_CONTROL
MOVE
x
y
c
TRACK
X= 12
Y= 70
T= 45
C= 44
STOPTRACK
SCAN
X1=
Y1=
T1=
X2=
Y2=
T2=
c=
FIXOFF
dx=
dy=
TRACKOFF
dx=
dy=
WRAP
BRAKE
HOLD
STOW
UNSTOW
RELEASE
ABORT
SETSRVTIME
REBOOT
SHUTDOWN
OPEN/CLOSE
SRVCMD



6.4 Optical Telescope – High level overview of the modules

There are four main modules in Optical Telescope.

i) CCU (Common Control Unit)

ii) TCS (Telescope Control System)
iii) I-FOSC
iv) ECS (Enclosure Control System)

6.4.1 CCU Module.

CCU is the embedded control system of the UNICS. It is designed and developed so that it can take various commands and instructions from the user through GUI via RS232 communication protocol and execute them. Its power requirements are also provided inside the enclosure box using switch mode power supplies. The basic architecture of CCU is built around Atmega 128 micro-controller and the design is based on the Master-Slave back- plane architecture. One of the most important initial steps during the development of the CCU was to design a system that requires minimum backups. So it was decided to adopt the industry standard while designing the PCB and the enclosure box for the CCU. The industry standard 3U size is used for the PCB designing while the enclosure box is of the dimension of 4U X 84 T **.**Further keeping the same idea in the mind it was decided to have only one type of card that will contain all the required functionality. Thus the CCU contains 6 identical cards and each one of them can be configured as the Master Card or Slave Card by changing the settings of few jumpers on enable pin is disconnected to the card by a jumper. Also since only the Master can communicate with the PC, two jumpers are used to make (in master mode) or break (in slave mode) the connections with the PC on the back-plane card.
For detail about protocol and the software architecture please refer the "**CCU_draft**" pdf document.

6.4.2 TCS Module

This is a TTL 2.0m Telescope is primarily a research telescope that can also be used for teaching purpose.
The Telescope incorporates motion control system to accurately position the telescope in altitude and azimuth. Additionally an auto guider is provided, which uses start image centroid in the telescope field of view to obtain the telescope position and track image. The motion control system uses embedded x86 based industrial PCs both on and off the telescope mount. Control data and status are distributed in real time throughout the system using internet protocol over switched network.

For getting more details please go through "OPERATOR's MANUAL" of 2.0m TELESCOPE.

6.4.3 ICS/ECS/MCS Module

This defines the interface between the Master Control System (MCS), which is the supervisory software system of the overall telescope computing systems, and the Enclosure Control System (ECS). There will be a single RS232 interface for sending enclosure requests (i.e. The Request Interface). This 'Request Interface' is used by the MCS to make asynchronous requests to the enclosure control system and by the enclosure control system to respond to the MCS. The command set described in this document is a GENERIC set of commands and responses based upon a virtual enclosure model that can accommodate both a conventional rotating dome and a non-rotating open-air enclosure. If a mechanism is not present on a particular enclosure then its status will be reported as not applicable (see Section 3). The OSI seven-layer network model is used to define the interface between the MCS and the ECS. For detailed understanding please go through the "MCS-IUC-Encl-ICD-V11 2" pdf document

6.4.4 I-FOSC Module

The FOSC motor controller box, contains 5 stepper motor controllers and 1 multiplexer PCB. The Multiplexer PC board distributes commands to the 5 stepper motor PC boards.
The 5 stepper motor PC boards are identical, containing one microprocessor system with non-volatile program memory, 5 phase stepper motor power drives and DC motor drives. All board have a same software in EPROM for driving all motors The identification of which type of movement is used on a particular board, is determined by the slot in which card is inserted, The PC board has a RS232 line to a terminal ,PC or workstation. It receives commands and distributes them to the drive board. The communication is RS232, 9600 baud 8bit, 1stop bit type.

6.5 Optical Telescope Command Set

Please consider the following points while considering the command set of optical telescope.

*** The following is used for setting the parameters

PRIVILEGE CLASSES:

1. Observer (OBS)
2. Operator (OPR)
3. Engineer (ENG)
4. Developer (DEV)
5. Admin / Super User (SU)

ACCESS MODES:

1. Local - When someone physically present in the observatory is issuing these commands
2. Remote - When someone outside observatory premises is issuing these commands
3. Session - Person may or may not be in the observatory (eg: through TCP/IP)

The privilege class is mentioned in **ascending** order of privileges, i.e. Admin / Super User have the highest privilege, followed by Developer and so on. The privilege class which is **NOT** allowed to execute the command is mentioned in the column.
Eg: "ENG" entry implies that Observer, Operator and Engineer should **not** be allowed to issue the command whereas Developer and Super User will have the privilege to do so.
All controlling parameters will have **ONLY** local mode access whereas all monitoring parameters will have **both** local and remote mode access
"Blank entries in Privilege to execute command" column implies all privilege classes are allowed to execute the command

Following are the types of alarms

ALARM STATUS

1. Level 1
2. Level 2
3. Level 3

By default all commands will have Level 1 alarm status which indicates error in a single command transaction. When there is persistent command failure, the alarm status should move to Level2. If there is any other problem which leads to hardware error (e.g. hardware limit hit), the alarm status should move to Level 3, Level4, Level5 etc depending on the seriousness of the failure.

Privilege classes, access modes and Alarm status should be dynamically configurable.

ALL the above information is to be treated as a baseline implementation guideline of the current use case.

The command set for each module is explained in detail as follows:

6.5.1 CCU Module:

SR NO	**PARAMETRES OF THE SUBYSTEM**	**INPUT PARAMTER**	**PRIVILEGE TO EXECUTE THE COMMAND***	**Controlling Parameters**	**Monitoring Parameters**	**TYPE OF THE PARAMETER**	**SIZE OF THE PARAMETER**	ALARMS/STATUS ###	**REMARKS**
1	**INIT**- This is the first command from the application program to the system and it is only for master subsystem. By the execution of this command the Master will know the available system IDs (i.e. the Slave subsystems that are functioning properly) and will keep trace of them.	SystemID	OPR	YES		Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
2	**MOVE** - Command to move an axis,	SystemID, No of Steps and	OPR	YES		Positive Integer, Positive	1 byte, 4bytes, 1 byte	LEVEL 3	

	which is attached to a subsystem, <No:steps> towards <Direction>.	Direction				Integer,Integer	(Total: 6 bytes)		
3	**MOVE WITH CONSTANT SPEED** - Command to move an axis attached to a system with constant speed. The system executes this command only if the axis is not already moving. Otherwise the system will ignore this command.	SystemID, No of Steps and Direction	OPR	YES		Positive Integer, Positive Integer,Integer	1 byte, 4bytes, 1 byte (Total: 6 bytes)	LEVEL 3	
4	**STOP**-This command will stop the movement of axis smoothly a moving axis will gradually reduce its speed and at last stops. If the axis is moving with constant speed or moving towards reference then the axis will stops immediately	SystemID	OPR	YES		Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
5	**HALT MOTION** - Command to immediately stop the motion of an axis.	SystemID	OPR	YES		Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	

6	**GOTO REFERENCE** - This command moves the axis towards <Direction> until it reaches at any of the extreme point or at the home.	SystemID, Direction	OPR	YES		Positive Integer, Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 3	
7	**SLEEP MOTOR** - To minimize power consumption when motor is not in use.	SystemID, ON/OFF	OPR	YES		Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
8	**DELAY** - Command to delay the subsystem from the executing the next ACTION command. The system ensures that there will be <Delay-time> milliseconds gap between every ACTION commands.	SystemID, Delay-time	OPR	YES		Positive Integer, Positive Integer	1 byte, 4 bytes (Total: 5 bytes)	LEVEL 1	
9	**SWITCH** - Command to turn On (Off) a switch. This command also contains SwitchID , which indicates the switch.	SystemID, SwitchID*, ON/OFF	OPR	YES		Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	
10	**PERIOD BLINK** - Command to blink a switch for <No:cycles>. The switch turn on	SystemID, SwitchID,ONduration, Offduration ,No of cycles.	OPR	YES		Positive Integer, Positive Integer, Positive Integer,	1 byte, 1 byte, 4 bytes, 4 bytes, 4 bytes (Total:	LEVEL 1	

	for <On-duration> (in milliseconds) and turn off for <Off-duration> (in milliseconds)and so on for <No:cycles>.					Positive Integer, Positive Integer	14 bytes)		
11	**HALT SWITCH** - Command to turn On (Off) a period switch that was currently Off (On) for a period.	SwitchID,SystemID	OPR	YES		Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	
12	**DEFINE MACRO** - The DEFINE MACRO will be used to define a set of commands to be executed one after another. Each group of commands is assigned a <Macro-Identifier>.The <Command-1>…………..<Command-n> are the commands with their respective command codes, Parameters size and Parameters.	SystemID, Macro identifier,Command 1-N	OPR	YES		Positive Integer, Positive Integer, (command structure of respective commands)	Macro (0-4) - upto 255 bytes, Rest - upto 64 bytes	LEVEL 1	
13	**EXECUTE MACRO** -The EXECUTE MACRO will be used to execute a macro which	SystemID, Macro-Identifier	OPR	YES		Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	

	has already been defined.								
14	**TRANSFER DATA** The <byte-1><byte-2> …..<byte-n> will be sent to some external instrument/unit through the Common Control Unit using its one of the two UARTs which is set for this purpose. The respective master/slave subsystem will transfer these bytes to the external instrument/unit and wait for the <Response-time> (in milliseconds) so that it can get the response back from the external instrument/unit.	SystemID, Response-time,Byte1-N	OPR	YES		Positive Integer, Positive Integer, Byte-1, Byte-2 …….. Byte n	Variable	LEVEL 1	
15	**SET MOTION PROFILE** - Command to set the motion profile (start speed, maximum speed, acceleration and deceleration) of an axis. The system will ignore this command if the	SystemID, Start-speed,Max-speed,Accelaration,deceleration	OPR	YES		Positive Integer, Positive Integer, Positive Integer, Positive Integer, Positive Integer	1 byte, 2 bytes, 2 bytes, 2 bytes, 2 bytes (Total: 9 bytes)	LEVEL 1	

	axis is already moving.								
16	**SET CURRENT POSITION** - Command for labelling the current position of axis as <Position>. The system will ignore this command if the axis is already moving.	SystemId, Position	OPR	YES		Positive Integer, Integer	1 byte, 4 bytes (Total: 5 bytes)	LEVEL 1	
17	**SET MICRO STEPS** - Command to set the micro steps for the stepper motor, which is responsible for the movement of axis. The system will ignore this command if the axis is already moving	SystemID, Microsteps	OPR	YES		Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	
18	**SET CONSTANT SPEED** - Command to set the constant speed for an axis, which is attached the specified system. The system will ignore this command if the axis is already moving. The constant speed must be	SystemId,speed	OPR	YES		Positive Integer, Positive Integer	1 byte, 2 bytes (Total: 3 bytes)	LEVEL 1	

	greater than or equal to 4 and less than or equal to 1000.								
19	**GET MOTION STATUS** - Command to get the status of an axis. The status can be one of the following.	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
20	**GET MOTION PROFILE**- Command to get the motion profile of an axis. Motion profile contains Start speed, Max Speed, Acceleration, and Deceleration.	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
21	**GET MICRO STEPS** Command to get the micro steps for an axis.	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
22	**GET CONSTANT SPEED** - Command to get the constant speed.	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
23	**GET CURRENT POSITION** Command to get the current position of an axis.	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	
24	**GET LIMIT STATUS** Command to check whether	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	

	the axis is at any of the extreme points or not. (Alternatively this command can be used to get back the status of two digital feedbacks (DF).)								
25	**GET SWITCH STATUS** Command to find whether a switch is On or Off. If the switch is a period switch the acknowledgement also contains the time interval.	SystemID, SwitchID	OPR		YES	Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	
26	**BLINK STATUS** Command to get the details of a blinking switch. For a blinking switch the acknowledgment gives the on duration and off duration. If the switch is blinking for a period the acknowledgement also contains the number of blinking left.	SystemID, SwitchID	OPR		YES	Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	
27	**GET ADC OUTPUT** Command to find the value of a given sensor (ADC).	SystemID, ADC-ID	OPR		YES	Positive Integer, Positive Integer	1 byte, 1 byte (Total: 2 bytes)	LEVEL 1	

28	**GET EXTERNAL RESPONSE** - Through this command the respective master/slave subsystem will send the response (i.e. the byte packet received from the external instrument/unit) back to PC. This byte packet was obtained by the master/slave subsystem during the execution of 'TRANSFER DATA' command (This has already been described in this document earlier).	SystemID	OPR		YES	Positive Integer	1 byte (Total: 1 byte)	LEVEL 1	

6.5.2 TCS Module:

SR NO	PARAMETRES OF THE SUBYSTEM	INPUT PARAMETER /KEYWORDS	PRIVILEGE TO SET THE PARAMETER(Please see footer of the table marked as ***)	Controlling Parameters	Monitoring Parameters	TYPE OF THE PARAMETER	SIZE OF THE PARAMETER	ALARMS/STATUS(Please see footer of the table marked as###)	REMARKS
1	**ACKNOWLEDGE** -Turn off the	LIMIT,ENGINEERIN		**yes**		**STRI**		LEVEL 1	

	engineering mode and/or software limit alarms	G,ALL				**NG**			
2	**ADD** -Write the current contets of the edit source data area as a new entry in the user catalogue	NONE		**yes**		**-**		LEVEL 1	
3	**AGCENTROID -** Centroid on current guide source then return pixel full width half maximum and stellar magnitude	NONE		**yes**		**-**		LEVEL 1	
4	**AGFILTER -**Set (Current) autoguider filter position.	IN/OUT		**yes**		**Boolean**		LEVEL 3	
5	**AGFOCUS** -Set (Current) autoguider filter position.	(0.00 TO 25.00)MM		**yes**		**Float**		LEVEL 3	
6	**AGMOVE** - Move autoguider probe mirror to a radial postion ,rotator to a mount postion angle and set autoguider pixel on which to guide.	Position, angle ,pixel_x,pixel_y		**yes**		**Float ,Float,Float,Float**	**Total : 16 bytes**	LEVEL 3	
7	**AGRADIAL**- Move autoguider probe mirror to a radial postion.	Postion,CENTRE		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1	
8	**AGSELECT** - Slect the autoguider to	CASSEGRAIN		**yes**		**-**		LEVEL 1	

	use.								
9	**AGVIEW** - Perform an aperture offset to move an image onto the autoguider CCD.	NONE		**yes**		**-**		LEVEL 1	
10	**AGWAVELENGTH**- Enter the value of the of the wavelength of light used in the atmospheric refraction calculation for the autoguider.	0.3 to 35(Microns)		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1	
11	**ALTITUDE-** Move to specified altitude and stop	20.00 to 90 degree		**yes**		**Float**	**Total : 4bytes**	LEVEL 1	
12	**APERTURE**- Offset the telescope so that an object moves by a vector fixed in the focal plane.	Number(0to10)		**yes**		**Integer**	**Total : 2 bytes**	LEVEL 1	
13	**AUTOGUIDE-** Lock or unlock the CCD autoguider loop.	state,agmode		**yes**		**String (TBC)**	**variable, mix of char and int (TBC)**	LEVEL 1	
14	**AZIMUTH** - Move to the specified azimuth and stop	angle		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1	
15	**BEAMSWITCH** - To offset the telescope from the refrence	opffset_x, offset		**yes**		**Float , Float**	**Total : 8 bytes**	LEVEL 1	

	postion so that an image moves to a given postion in the foacl plane.								
16	**BLIND_OFFSET** - Offset between a refrence and a faint target object.	name		**yes**		**String**	**TBC**	LEVEL 1	
17	**CALIBRATE** - Update a subset of the coefficients in the pointing model.	solution[mode]	OBS	**yes**		**String (TBC)**	**char (TBC)**	LEVEL 1	
18	**DARKSLIDE**- Open/Close dark slide if available.	OPEN/CLOSE		**yes**		**Boolean**		LEVEL 1	
19	**DEC** -Enter the declination in the edit source data area	dec_degrees,dec_minutes,dec_seconds		**yes**		**Integer, Positive Integer, Float**	**Total : 8 bytes**	LEVEL 1	
20	**DFOCUS**- Offset the focus to compensate for additional optical elements	focus_offset		**yes**		**Float**	**Total : 4Bytes**	LEVEL 1	
21	**DIFF_RATES** - Enter diffrential tracking rate in right ascension and declination	rate_in_ra, rate_in_dec		**yes**		**Float, Float**	**Total : 8 bytes**	LEVEL 1	
22	**DISPLAY** - Change the coordinate system of the information display.	coordinate_system		**yes**		**String (TBC)**	**Variable, mix of char and int**	LEVEL 1	

23	**DOME** - Move the dome to the specified azimuth and stop.	az_angle,el_angle		**yes**		**Float, Float**	**Total : 8 bytes**	LEVEL 3	
24	**ELEVATION** - Same as ALTITUDE			**yes**		**Same as Altitude**	**Same as Altitude**	LEVEL 1	
25	**ENCLOSURE** - Open/Close the enclosure mechanism	mechanism,state,pos		**yes**		**String, String (TBC), Float**	**TBC**	LEVEL 1	
26	**ENTER_OFFSET**-Set up aperture and postional offstres for repated use.	keyword , number xpos,ypos		**yes**		**String (TBC), Positive Integer (TBC), Float**	**Variable, mix of char and int / float**	LEVEL 1	
27	**EPOCH** - Enter a value for the epoch of the postion into the edit source data block	date		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1	
28	**EQUINOX**- Enter a value for the equinox of the postion into the edit source data block	equinox		**yes**		**String (TBC)**	**variable, mix of char and int**	LEVEL 1	
29	**ERASE** - Erase all entries from the current user catalogue	NONE		**yes**		**-**	**-**	LEVEL 1	
30	**EXIT** - Exit TCS user session	NONE	OBS	**yes**		**-**	**-**	LEVEL 1	

31	**FIND** - Retrieve data for the named source from the user or system catalogues and place them in the edit source data block.	source-name		**yes**		**String (TBC)**	**Total (max) : 20 characters (TBC)**	LEVEL 1	
32	**FOCUS** - Drive the focus to a specified setting and stop it	Postion,CENTRE		**yes**		**Float**	**Total : 4 Bytes**	LEVEL 3	
33	**GOCAT** - Retrieve the entry for the named source from the user or system catalogues and then send the telescope to track that source	name		**yes**		**String (TBC)**	**Total (max) : 20 characters (TBC)**	LEVEL 1	
34	**GOTO** - Move the telescope to a new source and track it	name,right_ascension,declination,equinox,epoch,diff_ra, diff_dec, pm_ra, pm_dec parallax,rad_vel		**yes**		**String (TBC)**	**Total (max) : 20 characters (TBC)**	LEVEL 3	
35	**HANDSET** - Place the TCS user interface in handset mode	NONE		**yes**		**-**	**-**	LEVEL 1	
36	**HELP**- Provide the information about the commands available from the TCS user	topic, subtopic		**yes**		**String (TBC)**	**-**	LEVEL 1	
37	**HUMIDITY** - Enter the value	value		**yes**		**Float**	**Total :**	LEVEL 1	

	of the relative humidity used in the calculation of refraction						**4Bytes**		
38	**INCLUDE** - Read in a text format source catalogue	name		**yes**		**String (TBC)**	**Total (max): 132 characters(TBC)**	LEVEL 1	
39	**INSTRUMENT**- Tell the TCS which instrument is in use. This command will set the focal station,rotator,focus offset, fold	instrument		**yes**		**String (TBC)**	**Total (max) : 20 characters (TBC)**	LEVEL 1	
40	**MARK** - Store the current postion of the telescope as a named catalogue entry	Source name		**yes**		**String (TBC)**	**Total (max) : 20 characters (TBC)**	LEVEL 1	
41	**MIRROR_COVER**- Open /Close mirror cover.	OPEN/CLOSE		**yes**		**Boolean(TBC)**		LEVEL 3	
42	**MOON** - Display the geocentric and topocentric apparent right ascension and declination of the moon.	NONE		**yes**		**-**	**-**	LEVEL 1	
43	**MOVE_FOLD** - Move fold mirror to a predetermined position.	state (1-4)		**yes**		**String (TBC)**		LEVEL 3	

44	**NETWORK** - Enable or disable the network command interface.	ENABLE/ DISABLE	OBS	**yes**		**Boolean (TBC)**		LEVEL 1	
45	**NEXT** - Send the telescope to track the source whose data are in the edit source block	NONE		**yes**		**-**	**-**	LEVEL 1	
46	**OFFBY** - Offset the telescope from refrence postion so that the image moves by a specified distance in the sky or tangent plane.	offset-mode,offset_ra.offest_dec		**yes**		**String, Float, Float**		LEVEL 1	
47	**OFFTO** - Offset the telescope from the refrence postion to a given postion.	ra_hours,ra_minutes,re_seconds,dec_degress,dec_minutes,dec_seconds, equinox		**yes**		**Positive Integer, Positive Integer, Float, Integer, Positive Integer, Float and equinox parameter type (TBC**	**TBC**	LEVEL 1	

						)			
48	**OPERATIONAL** - Prepare the telescope for observing or request the telescope stand-down at the end of an observing section	ON/OFF	OBS	**yes**		**Boolean (TBC)**		LEVEL 3	
49	**OUTPUT**-Write out the current user catalogue in text format to the printer ,TCS User interface or to disk file.	PRINTER, TERMINAL,FILE			**yes**	**String**		LEVEL 1	
50	**PARALLAX**- Enter a parallax into the edit source data area	value(0.0000 to 10.0000)		**yes**				LEVEL 1	
51	**PARK** - Park Telescope	Position, (Details can be seen in TCS Configuration file)		**yes**		**String (TBC)**		LEVEL 3	
52	**PM** - Same as PROPER_MOTION.			**yes**		**Same as PROPER_MOTION**	**Same as PROPER_MOTION**	LEVEL 1	
53	**POINT**- Log, in TPOINT format, the present position of the telescope as	file status		**yes**		**String (TBC)**		LEVEL 1	

	read on the encoders.							
54	**POLE**- Input values of polar motion	x_postion, y_postion		**yes**		**Float, Float**	**Total : 4 Bytes**	LEVEL 1
55	**POSTION** - Move the telescope by a previously stored offset. Tracking then continues as the offset postion.	number		**yes**		**Positive Integer**	**Total : 2 bytes**	LEVEL 1
56	**PRESSURE**- Enter the value of the barometric pressure used in the calculation of refraction	value		**yes**		**Float**	**Total : 4 Bytes**	LEVEL 1
57	**PROPER_MOTION** - Enter proper motions into the edit source data area	ra,dec.		**yes**		**Float, Float**	**Total : 8 bytes**	LEVEL 1
58	**RA** -.Enter a right ascension in the edit source data area	ra_hours,ra_minutes, ra_seconds		**yes**		**Positive Integer, Positive Integer, Float**	**Total : 8 bytes**	LEVEL 1
59	**RADIAL_VEL**- Enter a radial velocity in the edit source area	velocity(km/sec)		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1
60	**RECALL**- Recall previous command	command name		**yes**		**String**		LEVEL 1

61	**REMOVE** - Remove the entry for the named source from the user catalogue	name		**yes**		**String**	**Total (max): 20 characters**	LEVEL 1
62	**RMS** - Turn calculation of RMS errors on or off	error, state		**yes**		**String, Boolean (TBC)**		LEVEL 1
63	**ROTATOR** - Move the rotator to the specified mount or sky position angle or change its mode of operation	rotmode,angle		**yes**		**String, Float**		LEVEL 3
64	**RV** Same as RADIAL_VELOCITY			**yes**		**Same as RADIAL_VELOCITY**	**Same as RADIAL_VELOCITY**	LEVEL 1
65	**SHOW** - Display data on the topic indicated by the keyword	topic			**yes**	**String**		LEVEL 1
66	**SOURCE**- Enter new source data into the edit source data area	name,ra_hours,ra_minutes,ra_seconds,dec_degrees,dec_minutes,dec_seconds,equinox		**yes**		**String of upto 20 chars, Positive Integer, Positive Integer, Float ,**		LEVEL 1

						Integer, Positive Integer, Float, equinox parameter type(TBC)		
67	**STOP**- Stop the named mechanism	mechanism		**yes**		**String**		LEVEL 1
68	**STORE_OFFSET** - Store aperture and positional offset.	type, number		**yes**		**String, Positive Integer**		LEVEL 1
69	**TEMPRATURE** - Enter the value of the outside temperature used in the calculation of atmospheric refraction	value		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1
70	**TRACK** - Turn the focus ,dome, rotator or telescope tracking on or off	mechanism, OFF/ON		**yes**		**String, Boolean (TBC)**		LEVEL 1
71	**TWEAK**- Apply a given aperture offset to align a field on an instrument.	x_offset,y_offset,rotation		**yes**		**Float, Float, Float**	**Total : 12 bytes**	LEVEL 1

72	**UNWRAP** - Rotate either Azimuth axis or the current rotator by 360 degree from its current postion,if this is possible.	mechanism	OBS	**yes**		**String**		LEVEL 1
73	**UT1UTC**- Set the current UT1-UTC within the control system pointing calculations.	correction	OBS	**yes**		**Float**	**Total : 4 bytes**	LEVEL 1
74	**WAVELENGTH** - Enter the value of the effective wavelength of light used in the calculations	wavelength value		**yes**		**Float**	**Total : 4 bytes**	LEVEL 1

6.5.3 I-FOSC:

SR NO	**SETTABLE PARAMETERS OF THE SUBSYSTEM**	**INPUT PARAMTER /KEYWORDS**	**PRIVILEGE *****	**Controlling Parameters**	**Monitoring Parameters**	**TYPE OF THE PARAMETER**	**SIZE**	**ALARMS/STATUS(Please see footer of the table marked as###)**
	APERTURE COMMANDS							LEVEL 1
1	**AI -** Initialize position to hall switch position + the aperture_zero offset	NONE	OPR	YES		**String**		LEVEL 3
2	**AGnnnnnn[cr] Goto postion nnnnn.nnnnn is decimal,finish with carriage return.**	value in decimal	OPR	YES		**String**		LEVEL 3

3	**AM[s]nnnnnn[cr]** Move relative nnnnnn.A+ or - may precede nnnnn. The drive takes the shortest path to the postion.	value	OPR	YES		**String**		LEVEL 1
4	**AP** Send (read) postion 'nnnnn[cr]'	value	OPR		YES	**String**		LEVEL 1
5	**A1,A2,---A8** - Goto one of 8 positions(n*40000 steps),corresponding to a centred aperture plate opening.(Note that I-FOSC has 7 positions ,so these commands do not work correctly ,use AG or AM instead.)		OPR	YES		**String**		LEVEL 3
6	**Aq** - Quit the current operation 'A' and 'q' must come fast after each other.	NONE	OPR	YES		**String**		LEVEL 1
7	**a** - Check the driver ready. A 'y' or 'n' is returned for ready to accept commands or not ready. This does not indicate that the aperture wheel is not initialized, only that it is ready for commands,	NONE	OPR		YES	**String**		LEVEL 1
8	**AX** -Goto hall switch init position(Without moving to aperture_zero)	NONE	OPR	YES		**String**		LEVEL 3
9	**Aidfoc** - Set the aperture_zero position to this position , Stores the aperture_zero value in non-volatile eeprom. When using the AP command ,this position is reported as position to 0(zeroes)	NONE	OPR	YES		**String**		LEVEL 1
10	**Azdfoc** - Set the apaerture_zero value to 0(zero)		OPR	YES		**String**		LEVEL 1

	FILTER COMMANDS							
1	**FI -** Initialize position to hall switch position + the aperture_zero offset	NONE	OPR	YES		**String**		LEVEL 3
2	**FGnnnnnn[cr]** Goto postion nnnnn.nnnnn is decimal,finish with carriage return.	value in decimal	OPR	YES		**String**		LEVEL 3
3	**FM[s]nnnnnn[cr]** Move relative nnnnnn.A+ or - may prece de nnnnn. The drive takes the shortest path to the postion.	value	OPR	YES		**String**		LEVEL 3
4	**FP** Send (read) postion 'nnnnn[cr]'	value	OPR		YES	**String**		LEVEL 1
5	**F1,F2,---F8** - Goto one of 8 positions(n*40000 steps),corresponding to a centred aperture plate opening.		OPR	YES		**String**		LEVEL 1
6	**Fq** - Quit the current operation 'F' and 'q' must come fast after each other.	NONE	OPR	YES		**String**		LEVEL 1
7	**f** -Check the driver ready. A 'y' or 'n' is returned for ready to accept commands or not ready. This does not indicate that the aperture wheel is not initialized, only that it is ready for commands,	NONE	OPR		YES	**String**		LEVEL 1
8	**FX** -Goto hall switch init position(Without moving to aperture_zero)	NONE	OPR	YES		**String**		LEVEL 3
9	**Fidfoc** - Set the aperture_zero position to this position , Stores the aperture_zero value in non-volatile	NONE	OPR	YES		**String**		LEVEL 1

	eeprom. When using the AP command ,this position is reported as position to 0(zeroes)							
10	**Fzdfoc** - Set the apaerture_zero value to 0(zero)		OPR	YES		**String**		LEVEL 1
	Grism COMMANDS							
1	**GI -** Initialize position to hall switch position + the aperture_zero offset	NONE	OPR	YES		**String**		LEVEL 3
2	**GGnnnnnn[cr] Goto postion nnnnn.nnnnn is decimal,finish with carriage return.**	value in decimal	OPR	YES		**String**		LEVEL 3
3	GM[s]nnnnnn[cr] Move relative nnnnnn.A+ or - may prece de nnnnn. The drive takes the shortest path to the postion.	value	OPR	YES		**String**		LEVEL 3
4	GP Send (read) postion 'nnnnn[cr]'	value	OPR		YES	**String**		LEVEL 1
5	G1,G2,---G8 - Goto one of 8 positions(n*40000 steps),corresponding to a centred aperture plate opening.		OPR	YES		**String**		LEVEL 3
6	Gq - Quit the current operation 'G' and 'q' must come fast after each other.	NONE	OPR	YES		**String**		LEVEL 1
7	g Check the driver ready. A 'y' or 'n' is returned for ready to accept commands or not ready. This does not indicate that the aperture wheel is not initialized, only that it is ready for commands,	NONE	OPR		YES	**String**		LEVEL 1
8	**GX** -Goto hall switch init position(Without	NONE	OPR	YES		**String**		LEVEL 3

	moving to aperture_zero)							
9	Gidfoc - Set the aperture_zero position to this position , Stores the aperture_zero value in non-volatile eeprom. When using the AP command ,this position is reported as position to 0(zeroes)	NONE	OPR	YES		**String**		LEVEL 1
10	Gzdfoc - Set the apaerture_zero value to 0(zero)		OPR	YES		**String**		LEVEL 1
	Shutter COMMANDS							
1	**SI -** Initialize position to hall switch position + the aperture_zero offset	NONE	OPR	YES		**String**		LEVEL 3
2	**SEnnnnnn[cr] Start exposure for nnnnn*0.1 seconds.minimum is 4(400ms)**	value in decimal	OPR	YES		**String**		LEVEL 1
3	SS -Stop exposure	NONE	OPR	YES		**String**		LEVEL 1
4	SR- Resume exposure for the rest of the seconds started with SE.	value	OPR	YES		**String**		LEVEL 1
5	SAnnnnn[cr]- Add nnnnn*0.1 seconds to the exposure time started with SE. Can be done both when shutter open and closed. The number can be negative, but if it is larger than the exposure time left, exposure time left is set to 0,and the shutter is closed.		OPR	YES		**String**		LEVEL 1

6	SL - Get the exposure time left in nnnn*100ms	NONE	OPR	YES		**String**		LEVEL 1
7	SO-force shutter open	NONE	OPR	YES		**String**		LEVEL 3
8	**SC- Force shutter closed**	NONE	OPR	YES		**String**		LEVEL 3
9	SGnnnn[cr] - Goto position nnnnn.nnnnn is decimal, finish with carriage return	Value	OPR	YES		**String**		LEVEL 3
10	SM[s]nnnnn[cr] - Move relative nnnnn A + or - may prece de nnnnn The drive always take the same direction.		OPR	YES		**String**		LEVEL 3
11	SP- Get position 'C' for closed ,'O' for open		OPR		YES	**String**		LEVEL 1
12	S1,S2,S3,S4 - Goto one of 4 position (n*2000 steps),corresponding to an open or closed position.		OPR	YES		**String**		LEVEL 3
13	Sq - Quit the current operation must come first after each other.		OPR	YES		**String**		LEVEL 1
14	s - Check driver ready. A 'y' or 'n' is returned for ready to accept commands or not ready.This does not indicate that shutter is initialized, only that it is ready for commands.		OPR		YES	**String**		LEVEL 1
	Focus COMMANDS							
1	**CI -** Initialize position to init switch position then move to focus_zero	NONE	OPR	YES		**String**		LEVEL 3
2	**CGnnnnnn[cr]** Goto postion nnnnn.nnnnn is decimal,finish with carriage return.	value in decimal	OPR	YES		**String**		LEVEL 3

3	**CM[s]nnnnnn[cr]** Move relative nnnnnn.A+ or - may prece de nnnnn. The drive takes the shortest path to the postion.	value	OPR	YES		**String**		LEVEL 3
4	**CP** Send (read) postion 'nnnnn[cr]'	value	OPR		YES	**String**		LEVEL 1
5	**CX** - Initialize position to init switch position.		OPR	YES		**String**		LEVEL 3
6	**Cq** - Quit the current operation 'C' and 'q' must come fast after each other.	NONE	OPR	YES		**String**		LEVEL 1
7	**c** -Check the driver ready. A 'y' or 'n' is returned for ready to accept commands or not ready.	NONE	OPR		YES	**String**		LEVEL 1
8	**Cidfoc** - Set the focus_zero to this position .	NONE	OPR	YES		**String**		LEVEL 1
9	**Czdfoc** - Set the apaerture_zero value to 0(zero)	NONE	OPR	YES		**String**		LEVEL 1
	DIRECT COMMANDS TO MOTOR (for testing**)**							
	Step motor driver board commands							
1	**SM-**Turn power on step motor. Type + or - to single step. 'q' quits		OPR	YES		**String**		LEVEL 1
2	**SU or SD** Run forward or backward . Accelerate with '+',decelerate with'-' .Quit with 'q'.		OPR	YES		**String**		LEVEL 3
3	**MO[s]nnnnn**- Move motor +/- nnnnn steps		OPR	YES		**String**		LEVEL 3
4	**PO**- Send position back.		OPR			**String**		LEVEL 1
5	**RP** Reset postion to 0		OPR	YES		**Strin**		LEVEL 1

						g		
6	**IN** - Got hall init switches and set postion to 0		OPR	YES		**String**		LEVEL 1
7	**GI**- Goto hall init switches , do not set position to 0		OPR	YES		**String**		LEVEL 1
8	**GP[s]nnnnn** Goto position +/- nnnnn		OPR	YES		**String**		LEVEL 3
9	**RW** - Make a number of random turns.		OPR	YES		**String**		LEVEL 1
			OPR					
	Shutter Commands							
1	**SI**- Run shutter to hall switch position		OPR	YES		**String**		LEVEL 1
2	**Si** -As SI but does not set position to 0		OPR	YES		**String**		LEVEL 1
3	**Gannon**- Goto absolute position[0,7999].		OPR	YES		**String**		LEVEL 3
4	**SH** - Move shutter 90 degrees = 2000 steps		OPR	YES		**String**		LEVEL 3
5	**SX**- Make continuous 'SH' commands until 'q' is pressed.		OPR	YES		**String**		LEVEL 1
6	**PO** -send position		OPR		**YES**	**String**		LEVEL 1
	Focus commands							
1	**sm,su,sd** - Same as SM,SU,SD but for 2 phase focus motor		OPR	YES		**String**		LEVEL 1
2	**gpnnnn[cr]** Goto position nnnn		OPR	YES		**String**		LEVEL 3
3	**monnnn[cr]** Move nnnn steps		OPR	YES		**String**		LEVEL 3
4	**inNNNN[cr]** Initialize to init switch. Then move+NNNN steps		OPR	YES		**String**		LEVEL 1
5	**po** -send position		OPR		**YES**	**String**		LEVEL 1

6.5.4 <u>ECS Module:</u>

SR NO	PARAMETRES OF THE SUBYSTEM	INPUT PARAMTER	PRIVILEGE ***	Controlling Parameters	Monitoring Parameters	TYPE OF THE PARAMETER	SIZE OF THE PARAMETER	ALARMS/ STATUS###	REMARKS
	COMMANDS FROM MCS TO ECS								
1	**DOME**- To move the dome opening to a given azimuth and elevation position. This command is enclosure specific and is used to position the dome/shutter(s) into designated positions either manually or whilst tracking. Command Syntax: @ length DOME az-angle el-angle checksum Response Syntax: @ length DOME ---- response-data checksum	Azimuth angle (4 decimal digits), Elevation angle (4 decimal digits)	OPR	**YES**		String	23 bytes. (This will be written in the message as two hexadecimal digits (17).)	LEVEL 3	
2	**APERTURE**- To open/close the enclosure aperture fully or to a desired position. This command is enclosure specific and is used to open/close the enclosure aperture or to control individual aperture shutters (if available). Command Syntax: @ length APERTURE mechanism request-data checksum Response Syntax: @ length APERTURE mechanism response-data checksum	Mechanism(APER, SHU 1 or SHU 2), Request-data(OPEN or CLSD-4 decimal digits each)	OPR	**YES**		String	23 bytes. (This will be written in the message as two hexadecimal digits (17).)	LEVEL 3	

3	**STOP**- To request any one mechanism, or all mechanisms together, to stop. Command Syntax: @ length STOP mechanism ---- checksum Response Syntax: @ length STOP mechanism response-data checksum	Mechanism (ALL-, DOME, APER (i.e. all shutters), SHU 1 or SHU 2.)	OPR	**YES**		String	23 bytes. (This will be written in the message as two hexadecimal digits (17).)	LEVEL 1	
	STATUS REQUEST FROM MCS TO ECS								
1	**STATUS- To request the status of the enclosure. This command is sent periodically by the MCS (nominally once every second) to request the status of the various mechanisms of the enclosure. The ECS will respond by reporting the status (including fault conditions) back to the MCS.** **Command Syntax: @ length STATUS ---- ---- checksum**	None			YES	String	i) This will always be 23 bytes for a request. ii) The response shall be 66 bytes for a succes	LEVEL 1	

	Actioned Syntax: @ length STATUS ---- response-data status-information checksum						sful response or 23 bytes for an unsuccessful respons		

6.6 Radio telescope modules – overview

Modules in the data path:

Front end: Section immediately downstream of the antenna feed. Consists of two channels, for each polarization sensed of the incoming signal. It has gain / attenuator controls and the ability to swap the channels. Further the front end electronics can be disconnected from the feed and terminated into a standard signal for testing purposes.

Sig-con: It consists of a pre-attenuator, filter, local oscillator (LO) which is used to select the observation bandwidth by heterodyning, Automatic Level Controller. As the Local oscillator frequency is dynamic, it is important that the LO be locked at the set frequency for the data to be valid.

Sigtrans: Transports the signal from the antenna to the control room over optical fibre using modulation and demodulation. The transport is analog for the current 45m dishes at GMRT, but with the advances in processing speed for the optoelectronic circuitry, it is proposed to be implemented in digital domain for the 15m dish.

Back-end: This stage digitizes the signal and stores it in a staging file with some of the meta-data.

Modules not in the data path:

Servo: Controls the antenna position through servo motor / servo drive / controller. It is responsible for performing positioning and tracking. The servo module works on earth co-ordinates and higher level modules are responsible for translating celestial co-ordinates into earth co-ordinates.

Sentinel: Responsible for monitoring environmental and safety parameters such as wind speed, temperature, smoke etc. and provide the CMS with the logs and generate alarms if necessary. Also controls some of the emergency response systems such as fire-extinguishers, power shut-down etc.
ABC: This is the antenna based computer that is located at the antenna shell and is responsible for passing on the CMS data to the other antenna hardware and controls the telemetry operations.

6.7 Radio Telescope Baseline Parameters

The command set for each subsystem/module of the Radio Telescope has been explained here. The privileges for every parameter are mentioned. The parameters are identified as an alarm or status parameter. It should be represented in the GUI appropriately.

6.7.1 Servo subsystem:

Note: This command set for the servo motor is based on the existing Servo controller at GMRT and is not updated for the proposed Baldor drive / servo controller for the 15m telescope and can be considered as a guideline only. Once the command set for Baldor is ready, the same would need to be updated

i) Controlling Parameters of Servo Module (ENGINEER GUI):

SR. NO.	PARAMETERS TO SET	TYPE OF PARAM.	SIZE OF PARAM.	PRIVILEGES	ALARMS /STATUS	REMARK
1	MOVE (x,y,c) move antenna to position (x,y). (x,y) could be in various co-ordinate systems (e.g. Alt-Az or Ra-Dec) and a variety of units e.g. (deg, min,sec), (hr,min,sec), radians etc. The argument c specifies the co-ordinate system. Position could also be given as the name of a celestial source (e.g. 3C123) in which case the co-ordinates are looked up from a data base. This range of possibilities for specifying positions should be possible for all commands which take position as an argument. Positions should be checked before the command is executed, if the position lies outside the permitted range (the “software limits”) the command should be rejected in the general mode. In a restricted “expert” mode though one could go beyond the software limit till the hardware limit. It would also be desirable to have special variables e.g. $x, $y which the system replaces with the current antenna position.	float,float,char	1, 1, 128	both,user,engineer	1, 2,3	cable wrap, software limit, hardware limit

2	TRACK(x,y,t,c) Track a position in the sky for time t	float, float, float, char	1, 1, 1, 128	user, engr		
3	STOPTRACK stop the current track, if any	nil	nil	user, engr	nil	
4	SCAN(x1,y1,t1,x2,y2,t2,c) Move the antenna at a uniform speed from co-ordinates (x1,y1) to (x2,y2). Start the motion at time t1 and stop at time t2.	float,float,float, float,float,float, char	1,1,1 1,1,1, 128	user, engr	nil	
5	FIXOFF(dx,dy) add a fixed "pointing offset" (dx,dy) to the Altitude and Azimuth. (Correction for mechanical alignment errors).	float, float	1, 1	user, engr		
6	TRACKOFF(dx,dy) Add offsets (dx,dy) to the current tracking position.	float, float	1, 1	user, engr		
7	WRAP(n) set the cable wrap to the specified argument.	integer		user, engr		
8	BRAKE(arg) apply brakes in azimuth/elevation/both	char	128	user, engr		
9	HOLD hold the antenna in the current position			user, engr		Alt/Az/both
10	STOW park the antenna in a specified safe position, applying stowing pins etc.	nil	nil	user, engr		
11	UNSTOW take the antenna out of the stow position	nil	nil	user, engr		
12	RELEASE(L) release specified interlock. The interlocks are 1. Azimuth Limit1 reached 2. Azimuth Limit2 reached 3. Elevation Limit1 reached 4. Elevation Limit2 reached 5. Wind speed exceeded. Interlocks can be released only in a restricted "expert" mode.	char	128	engr		
13	ABORT- abort current command and HOLD the antenna in the current position	nil	nil	User. Engr		
14	SETSRVTIME set the time on the SERVO clock	char	128	engr		
15	REBOOT/RESET reboot the servo computer (power on if shut	nil		engr		

	off)					
16	SHUTDOWN -shutdown the servo computer	nil		engr		
17	CLOSE - close the link to the servo system (to prevent accidents during maintenance)	nil		engr		
18	OPEN -open the link to the servo system	nil		engr		
19	SRVCMD(x) send specified string to the servo	char	2048	engr		
20	INIT- Releases the stow and hold the position	nil		user, engr		
21	PARK- This means Az= 0 deg. And ELV= 90 deg.	nil		user, engr		actual park position set by Engr.

ii) Monitoring Parameters of SERVO Module (ENGINEER GUI):

SR .NO	PARAMETR TO MONITOR	TYPE OF PARAM.	SIZE	PRIVILEGES	ALARMS/STATUS	REMARK
1	ENCODER POSITION current encoder position, this should be returned in any of the co-ordinate systems and units described above (trend plot option needed). Options for display with or without correction for the specified offsets.	float,float,char	1, 1, 128	user,engr		
2	COMMANDED POSITION and RATE	float,float,char	1, 1, 128	user,engr		
3	MOTOR CURRENTS currents in the azimuth and elevation motors (trend plot option needed)	float,char	1, 128	engr		
4	WRAP current cable wrap	Integer	2	user,engr		
5	TACHO readings from Elevation and Azimuth motors	float,char	1, 128	engr		
6	TIME time on the servo computer	float, char	1, 128	user,engr		
7	STATUS status of the servo computer (on/off)	boolean	Logic 1/0	engr		1- SSSC 'ON', 2 -

						SSC 'OFF'
8	INTERLOCKS status of the different interlocks, (elevation soft limit reached, azimuth soft limit, elevation hard limit, azimuth hard limit, cable wrap limits) , level 1 alarm if soft limit is reached, level 2 alarm if hard limit is reached	char	128	engr		
9	LIMITS current software	float, char	1, 128	user,engr		
10	BRAKES status of the brakes (applied/released)	boolean	Logic 1/0	engr		
11	STOW stow status, i.e. stowed or unstowed	boolean	Logic 1/0	user,engr		
12	COMMAND last command being executed and its status (success/failure/inprogress)	Integer	3	user,engr		
13	SERVO servo status i.e. Moving, Tracking, Holding,Scaning	Integer	4	user,engr		
14	OFFSETS current Fixed and Tracking offset values	float, float, char	1, 1, 128	user,engr		
15	LINK status of link to servo (open/close)	boolean	Logic 1/0	engr		
16	VERSION servo software version	float, char	1, 1, 128	engr		
17	ID of Servo subsystem	float, char	1, 1, 128	engr		ew feature

6.7.2 SIGCON subsystem:

i) <u>Controlling Parameters of SIGCON Module (ENGINEER GUI):</u>

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	ALARMS/ STATUS	REMARK

1	PREATTEN(x) set the pre filtering attenuation to the specified value (dropdown button in GUI)	DOUBLE	1	all	status	value in dB
2	LO(x) set the local oscillator frequency to the specified value	DOUBLE	1	all	Status-Alarm if LO is in unlock	value in MHz
3	FILTER(n) select the specified filter (dropdown button in GUI)	INTEGER	2	all	status	
4	POSTATTEN(x) set the post filtering attenuation to the specified value (dropdown button in GUI)	DOUBLE	1	all	status	value in dB
5	ALC(on/off) switch on/off the Automatic Level Controller (dropdown button in GUI)	BOOLEAN	1	all	status	

ii) **Monitoring Parameters of SIGCON Module (ENGINEER GUI):**

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARAM.	SIZE OF PARAM.	PRIVILEGE TO SEE THE PARAMETER	REMARK
1	Preattenuation value set	DOUBLE	1	all	dB
2	Status of the LO, i.e. locked/unlocked	BOOLEAN		all	alarm if NOT locked
3	Frequency that the LO is at if locked	DOUBLE	1	all	MHz
4	Filter selected	INTEGER	2	all	status
5	Postattenaution value set	DOUBLE	1	all	dB
6	ALC status (on/off)	BOOLEAN		all	status
7	POWERSUPPLY (status of power supply)	DOUBLE	1	all	ALARM if NOT WORKING
8	Monitor parameters from the GPS system	string		all	ALARM if NOT locking
9	LO Power	FLOAT	1	all	dB out of range – alarm
10	INPUT SIGNAL	FLOAT	2	all	dB out of range –

	POWER(Set of two)				alarm
11	OUTPUT SIGNAL POWER(set of four)	FLOAT	4	all	dB out of range – alarm

iii) Controlling Parameters of SIGCON Module (USER GUI):

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET PARAMETER	ALARMS/STATUS	REMARK
1	PREATTEN(x) set the pre filtering attenuation to the specified value	DOUBLE	1	all	status	value in dB
2	LO(x) set the local oscillator frequency to the specified value	DOUBLE	1	all	Status-Alarm if LO is in unlock	value in MHz
3	FILTER(n) select the specified filter (dropdown button in GUI)	INTEGER	2	all	status	
4	POSTATTEN(x) set the post filtering attenuation to the specified value	DOUBLE	1	all	status	value in dB
5	ALC(on/off) switch on/off the Automatic Level Controller (radio button in GUI)	BOOLEAN	1	all	status	

iv) Monitoring Parameters of SIGCON Module (USER GUI):

SR. NO.	PARAMETERS TO BE MONITORED	PRIVILEGE TO SEE THE PARAMETER	Type of parameter	Size of Parameter	REMARK
1	Pre attenuation value set	all	DOUBLE	1	dB
2	Status of the LO, i.e. locked/unlocked	all	BOOLEAN		alarm if NOT locked
3	Frequency that the LO is at if locked	all	DOUBLE	1	MHz
4	Filter selected	all	INTEGER	2	status

5	Postattenaution value set	all	DOUBLE	1	dB
6	ALC status (on/off)	all	BOOLEAN		status
7	POWERSUPPLY (status of power supply)	all	DOUBLE	1	ALARM if NOT WORKING
8	Monitor parameters from the GPS system	all	string		ALARM if NOT locking
9	LO Power	all	FLOAT	1	dB out of range – alarm
10	INPUT SIGNAL POWER(Set of two)	all	FLOAT	2	dB out of range – alarm
11	OUTPUT SIGNAL POWER(set of four)	all	FLOAT	4	dB out of range – alarm

6.7.3 Frontend subsystem:

i) <u>Controlling Parameters of Frontend Module (ENGINEER GUI):</u>

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	ALARMS/STATUS	REMARK
1	ENABLE switch on the front end control card	Boolean	Logic 1/0	Engineer	Alarm if ON	Default OFF mode during Normal observation
2	DISABLE switch off the front end control card	Boolean	Logic 1/0	Engineer	Alarm if ON	Default OFF mode during Normal observation
3	POWER(on/off) switch on/off the entire front end system (for safety in case of very adverse weather conditions).	Boolean	Logic 1/ 0	Engineer	HIGH for OFF(Alarm)	Auto shutdown if High Temp and Manual OFF for maintenance

4	TERMINATE give a standard internal calibration signal to the input of the amplifiers instead of the astronomical signal from the antenna. NOTE: each front end has two channels ("polarizations") so most of the commands below and in the monitor section have to be channel specific, i.e. one can do the setting/monitor the parameters independently for each channel.	Boolean	Logic 1 for Termination	Engineer	Alarm if ON	Default is OFF
5	ATENN(x) set the attenuation to the specified value (administrator should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	Engineer	Status	Each bit combination set different values of attenauation
6	GAIN(x) set the gain to the specified value(administrator should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	Engineer	Status	
7	FILTER(x) turn on the specified filter . (Administrator should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	Engineer	Status	
8	CALSEL(x)	Boolea	Logic	Engin	Alarm if	By default -OFF.

	Turning calibration ON/OFF.	n	1/0	eer	ON	
9	CALTYPE(n) select the calibration type (e.g. none, Duty cycle of 5%, 10%,25% and 100%. Engineer/Astronomer should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	Engin eer	Status	
10	SWAP/UNSWAP swap the two polarization channels	Boolea n	Logic 1 / 0	Engin eer	Alarm if SWAP= ON	Default is OFF
11	Walsh(n) set the Walsh pattern	Integer	Intege r	Engin eer	Status (pattern number)	Not required
12	SAVE(a) Save current frontend settings into “SET A”. Could have N possible sets	String	String	Engin eer	NA	Could give names for each setting
13	RESTORE(a) Restore front end settings from “SET A”.	String	String	Engin eer	NA	Call By name given above
14	SETID_LNA/SETID_ ATTENUATOR Set the ID for LNA as well as Attenuator	String	String	Engin eer	Status	ID given at the time of Installation.
15	WALSH (Enable/Disable)	Boolea n	Logic 1 / 0	Engin eer	Status	default?

ii) <u>Monitoring Parameters of Front-End Module (ENGINEER GUI):</u>

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARA METER	SIZE OF PAR AME TER	PRIVILE GE TO SEE THE PARAME TER	ALAR MS/ST ATUS	REMARK
1	FRONT END ID	String	String	Engineer	Status	
2	CONTROL State of the control card	Boolea n	logic 0/1	Engineer	Alarm if ON	

	(enabled/disabled)					
3	POWER State of front end system (power on/power off)	Boolean	logic 0/1	Engineer	Alarm if OFF	
4	TERMINATION State of the termination (on/off)	Boolean	logic 0/1	Engineer	Alarm if ON	
5	ATTENUATION Attenuation value	Integer		Engineer	Status	
6	FILTER Filter selected	string		Engineer	Status	
7	CAL ON/OFF Calibration ON/OFF	Boolean	logic 0/1	Engineer	Alarm if ON	
8	CALTYPE Calibration type (5%,10% etc) selected	string	String	Engineer	Status	
9	RFPOWER Signal power level at the output of the amplifier (trend plot option and out of range alarm needed)	Float	Float	Engineer	Status	
10	POWERSUPPLY power supply current and voltage	Float	Float	Engineer	Status	
11	SWAP Swap status	Boolean	logic 0/1	Engineer	Alarm if SWAP= ON	
12	TEMPERATURE of FrontEnd	Float	Float	Engineer	Alarm if HIGH & Status	Normally status and if limit is exceeded then ALARM.
13	WALSH (Enable/Disable)	Boolean	logic 0/1	Engineer	Status	

iii) <u>Controlling Parameters of Front-End Module (USER GUI):</u>

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	ALARMS/STATUS	REMARK
1	ENABLE switch on the front end control card	Boolean	Logic 1 / 0	All	Flash if ON	
2	DISABLE switch off the front end control card	Boolean	Logic 1 / 0	All	Alarm -If tried to issue communica	

					te in off condition	
3	POWER(on/off) switch on/off the entire front end system (for safety in case of very adverse weather conditions).	Boolean	Logic 1 / 0	Auto / All	Alarm if OFF	BY default AUTO.
4	TERMINATE give a standard internal calibration signal to the input of the amplifiers instead of the astronomical signal from the antenna. NOTE: each front end has two channels ("polarizations") so most of the commands below and in the monitor section have to be channel specific, i.e. one can do the setting/monitor the parameters independently for each channel.	Boolean	Logic 1 / 0	All	Flash if ON	default OFF.
5	ATENN(x) set the attenuation to the specified value (administrator should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	All	Status	If ALC is ON then ATTN is in AUTO.
6	GAIN(x) set the gain to the specified value(administrator should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	All	Status	If ALC is ON then GAIN is in AUTO.
7	FILTER(x) turn on the specified filter . (Administrator should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	All	Status	
8	CALSEL(x) Turning calibration ON/OFF.	Boolean	Logic 1 / 0	All	Alarm if ON	default OFF.

9	CALTYPE(n) select the calibration type (e.g. none, Duty cycle of 5%, 10%,25% and 100%. Engineer/Astronomer should be able to configure system to specify permitted values for each frontend)	Byte	1 Byte	All	Status	
10	SWAP swap the two polarization channels	Boolean	Logic 1 / 0	All	Flash if ON	default OFF.
11	SAVE(a) Save current frontend settings into "SET A". Could have N (=4?) possible sets	String	String	All	NA	(Common for all subsystems.)
12	RESTORE(a) Restore front end settings from "SET A".	String	String	All	NA	(Common for all subsystems.)
13	SETID_LNA/SETID_ATTENUATOR Set the ID for LNA as well as Attenuator	String	String	NOT FOR ASTRONOMER	Status	
14	WALSH (Enable/Disable)	Boolean	Logic 1 / 0	All	Status	default?

iv) **Monitoring Parameters of Front-End Module (USER GUI):**

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SEE THE PARAMETER	ALARMS/STATUS	REMARK
1	FRONT END ID	String	String	All	Status	
2	CONTROL State of the control card (enabled/disabled)	Boolean	logic 0/1	All	Alarm if ON	
3	POWER State of front end system (power on/power off)	Boolean	logic 0/1	All	Alarm if OFF	
4	TERMINATION State of the termination (on/off)	Boolean	logic 0/1	All	Alarm if ON	
5	ATTENUATION Attenuation value	Integer		All	Status	

6	FILTER Filter selected	string		All	Status	
7	CAL ON/OFF Calibration ON/OFF	Boolean	logic 0/1	All	Alarm if ON	
8	CALTYPE Calibration type (5%,10% etc) selected	string	String	All	Status	
9	RFPOWER Signal power level at the output of the amplifier (trend plot option and out of range alarm needed)	Float	Float	All	Status	
10	POWERSUPPLY power supply current and voltage	Float	Float	All	Status	
11	SWAP Swap status	Boolean	logic 0/1	All	Alarm if SWAP= ON	
12	TEMPERATURE of FrontEnd	Float	Float	All	Alarm if HIGH & Status	Normally status and if limit is exceeded then ALARM.
13	WALSH (Enable/Disable)	Boolean	logic 0/1	All	Status	

6.7.4 Back-End Subsystem:

i) <u>Controlling Parameters of Back-End Module (ENGINEER GUI):</u>

SR NO	PARAMETERS/COMMANDS TO SET	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGES	ALARMS /STATUS	REMARK
1	FILE name of the file into which astronomical data is being recorded	STRING	2048	all	status	
2	START start recording of the astronomical data. There should be an interlock which starts the recording only when the antenna has started executing the specified TRACK or SCAN commands.	STRING	2048	all	Status- alarm if rec not started	file name with source name and other parameters

	The current antenna position, and selected monitor parameters from the SERVO, FrontEnd, SigCon and BackEnd systems has to be sent along with the start command.					
3	STOP stop recording of the astronomical data	STRING		all	Status-alarm if rec not stopped	
4	CONFIG FILE-CONFIGURING PARAMETERS	STRING	2048	all		
	1) SAMPLING RATE-CLOCK RATE-drop down gui	DOUBLE	1	all		value in MHz
	2) adc Mode-drop down gui	char	1	all		Mode Name, selected from drop down list
	3) SAMPLING GAIN-drop down gui	DOUBLE	1	all		value in dB, selected from drop down lost
	4) NO OF CORRELATIONS-drop down gui	INTEGER	2	all		selected from drop down list
	5) CORRELATION TYPE-drop down gui	char	1	all		selected from drop down list
	6) NO OF CHANNELS-drop down gui	INTEGER	2	all		selected from drop down list
	7) INTEGRATION TIME-drop down gui	INTEGER	2	all		
	SET CONFIGURING PARAMETERS-button	boolean		all	status	
5	META DATA			all		
	OBSERVER NAME	STRING	2048	all		
	PROPOSAL ID	STRING	2048	all		
	SOURCE NAME	STRING	2048	all		
	SOURCE CO-ORDINATES[N]	STRING	2048	all		hh:mm:ss
	ANTENNA CO-ORDINATES	STRING	2048	all		dd:mm:ss
	BANDWIDTH	STRING	2048	all		value in MHz

	CENTER FRQUENCY	STRING	2048	all		value in MHz
	FLAGS-ERROR CONDITION	STRING	2048	all		
	INSTRUMENT ID	STRING	2048	all		

ii) Monitoring Parameters of Back-End Module (ENGINEER GUI):

SR. NO.	PARAMETER TO MONITOR	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE	ALARMS/STATUS	REMARK
1	Attenuation values which is set	FLOAT	1	all	status	value in dB
2	ADC mode set	STRING	16	all		Mode Name
3	Configuration file name	STRING	2048	all		
4	File name of the astronomical data file	STRING	2048	all		
5	Status of recording (start/stop)	boolean		all	Status-alarm if not started or stopped	
6	Current scan number in the astronomical file	INTEGER	2	all		
7	Various trend plots of the astronomical data being acquired			all		
8	ADC Sampling rate	DOUBLE	1	all	status	value in MHz

iii) Controlling Parameters of Back-End Module (USER GUI):

SR NO	PARAMETERS/COMMANDS TO SET	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGES	ALARMS/STATUS	REMARK
1	FILE name of the file into which astronomical data is being recorded	STRING	2048	all	status	

2	START start recording of the astronomical data. There should be an interlock which starts the recording only when the antenna has started executing the specified TRACK or SCAN commands. The current antenna position, and selected monitor parameters from the SERVO, FrontEnd, SigCon and BackEnd systems has to be sent along with the start command.	STRING	2048	all	Status-alarm if rec not started	file name with source name and other pars
3	STOP stop recording of the astronomical data	STRING		all	Status-alarm if rec not stopped	
4	CONFIG FILE-CONFIGURING PARAMETERS	STRING	2048	all		
	1) SAMPLING RATE-CLOCK RATE-drop down gui	DOUBLE	1	all		value in MHz
	2) adc Mode-drop down gui	char	1	all		Mode Name, selected from drop down list
	3) SAMPLING GAIN-drop down gui	DOUBLE	1	all		value in dB, selected from drop down lost
	4) NO OF CORRELATIONS-drop down gui	INTEGER	2	all		selected from drop down list
	5) CORRELATION TYPE-drop down gui	char	1	all		selected from drop down list
	6) NO OF CHANNELS-drop down gui	INTEGER	2	all		selected from drop down list
	7) INTEGRATION TIME-drop down gui	INTEGER	2	all		
	SET CONFIGURING PARAMETERS-button	boolean		all	status	
5	META DATA			all		
	OBSERVER NAME	STRING	2048	all		

	PROPOSAL ID	STRING	2048	all		
	SOURCE NAME	STRING	2048	all		
	SOURCE CO-ORDINATES[N]	STRING	2048	all		hh:mm:ss
	ANTENNA CO-ORDINATES	STRING	2048	all		dd:mm:ss
	BANDWIDTH	STRING	2048	all		value in MHz
	CENTER FRQUENCY	STRING	2048	all		value in MHz
	FLAGS-ERROR CONDITION	STRING	2048	all		
	INSTRUMENT ID	STRING	2048	all		

iv) Monitoring Parameters of Back-End Module (USER GUI):

S R. N O.	**PARAMETER TO MONITOR**	**TYPE OF PARAM ETER**	**SIZE OF PARAM ETER**	**PRIVI LEGE**	**ALARMS/STA TUS**	**REMA RK**
1	Attenuation values which is set	FLOAT	1	all	status	value in dB
2	ADC mode set	STRIN G	16	all		Mode Name
3	Configuration file name	STRIN G	2048	all		
4	File name of the astronomical data file	STRIN G	2048	all		
5	Status of recording (start/stop)	boolean		all	Status-alarm if not started or stopped	
6	Current scan number in the astronomical file	INTEG ER	2	all		
7	Various trend plots of the astronomical data being acquired			all		
8	ADC Sampling rate	DOUBL E	1	all	status	value in MHz

6.7.5 SIGTRANS Subsystem:

i) Controlling Parameters of SIGTRANS Subsystem (ENGINEER GUI):

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	REMARK
1	POWER switch on/off the power to the SIGTRANS system (in case of adverse weather conditions)	Boolean	Logic 1 / 0	Engineer	Auto in case of emergency and Manual during maintenance
2	ENABLE enable the SIGTRANS control card	Boolean	Logic 1	Engineer	
3	DISABLE disable the SIGTRANS control card	Boolean	Logic 0	Engineer	
4	BAND(n) select the band for the SIGTRANS system (would ideally like a default selection based on the FrontEnd ID).	Integer	1 Byte	Engineer	
5	ALC(on/off) set the ALC (Automatic Level Control) on or off	Boolean	Logic 1 / 0	Engineer	By default it is ON
6	SAVE(a) Save current SIGTRANS settings into "SET A". Could have N (may be >4) possible sets	String	String	Engineer	When said save it should ask for setting name so that we could call the setting by name.
7	RESTORE(a) restore SIGTRANS settings from "SET A".	String	String	Engineer	When said save it should ask for setting name so that we could call the setting by name.

ii) Monitoring Parameters of SIGTRANS Subsystem (ENGINEER GUI):

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARAMETER	Alarm/Status	PRIVILEGE TO SEE THE PARAMETER	REMARK
1	STATUS status of the SIGTRANS system (power on/off)	Boolean	Flash if OFF	All	

2	CONTROL status of the control card (enabled/disabled)	Boolean	Flash if OFF and command is sent	All	
3	iii. POWERSUPPLY currents and voltages from the powersupply (trend plot option and out of range alarm needed)	Float	Alarm if Voltage and Current High	All	
4	BAND band selected	String	Status	All	
5	ALC Automatic Level Control status (on/off)	Boolean	Status	All	
6	BIAS bias current for the Laser Diode (trend plot option, and out of range alarm needed)	Float	Alarm if Bias current drops	All	
7	OPTICPOWER optical power level at the photodiode (trend plot option and out of range alarm needed).	Float	Alarm if No power	All	
8	Temperature monitor	Float	Alarm if temperature is high	All	

iii) Controlling Parameters of SIGTRANS Subsystem (USER GUI):

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	REMARK
1	POWER switch on/off the power to the SIGTRANS system (in case of adverse weather conditions)	Boolean	Logic 1 / 0	Not for user(astronomer)	By default in Auto mode
2	ENABLE enable the SIGTRANS control card	Boolean	Logic 1	Not for user(astronomer)	
3	DISABLE disable the SIGTRANS control card	Boolean	Logic 0	Not for user(astronomer)	
4	BAND(n) select the band for the SIGNTRANS system (would ideally like a default selection based on the FrontEnd ID).	Integer	1 Byte	User/Astronomer	

5	ALC(on/off) set the ALC (Automatic Level Control) on or off	Boolean	Logic 1 /0	User/Astronomer	
6	SAVE(a) Save current SIGTRANS settings into “SET A”. Could have N (=4?) possible sets	String	String	User/Astronomer	
7	RESTORE(a) restore SIGTRANS settings from “SET A”.	String	String	User/Astronomer	

iv) <u>Monitoring Parameters of SIGTRANS Subsystem (USER GUI):</u>

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARA METER	SIZE OF PARAME TER	Alarm/Statu s	PRIVILE GE	REM ARK
1	STATUS- status of the SIGTRANS system (power on/off)	Boolea n	Logic 1 / 0	Flash if Off(Alarm)	All	
2	CONTROL status of the control card (enabled/disabled)	Boolea n	Logic 1 / 0	Flash if Off and command is given(Alarm)	All	
3	iii. POWERSUPPLY currents and voltages from the power supply (trend plot option and out of range alarm needed)	Float		Alarm needed for voltage and current	All	
4	BAND- band selected	String		Status	All	
5	ALC Automatic Level Control status (on/off)	Boolea n	Logic 1 /0	Status	All	
6	BIAS- bias current for the Laser Diode (trend plot option, and out of range alarm needed)	Float		Alarm needed if Bias is low	All	
7	OPTICPOWER optical power level at the photodiode (trend plot option and out of range alarm needed).	Float		Alarm needed if no power	All	
8	Temperature monitor	Float		Alarm needed if exceeds	All	

6.7.6 ABC (Antenna Based Control) Subsystem:

i) **Controlling Parameters of ABC Subsystem (ENGINEER GUI):**

SR NO	SETTABLE PARAMETERS OF THE SUBSYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	ALARMS/STATUS	REMARK
1	REBOOT- reboot the ABC (power on if shut off)	String	6 Bytes	Expert and Engineer	Alarm	
2	SHUTDOWN- shutdown the ABC computer	String	8 Bytes	Expert and Engineer	Alarm	
3	OPEN- open the link to the ABC	String	4 Bytes	Expert and Engineer	Alarm	
4	CLOSE- close the link to the ABC	String	5 Bytes	Expert and Engineer	Alarm	
5	SETTIME set the ABC time	String		Expert and Engineer	Status	
6	SHELL open a remote shell on the ABC (essentially this command will execute an “ssh” command on the ABC, the user will be prompted for a password by the ABC).	String	5 Bytes	Expert and Engineer	Status	
7	SAVE(a) save the current settings of all subsystems that the ABC is configured for into “SET A” (could have N (=4?) possible sets)	String	7 Bytes	Expert and Engineer	Status	
8	RESTORE(a) settings from “SET A”	String	10 Bytes	Expert and Engineer	Status	
9	MODE Local/remote. In the local mode a CMS User Interface is opened on the local console. In the remote mode, the control is via Ethernet TCP/IP to the CMS	char	1 Byte	Expert and Engineer	Alarm	
10	Enable/Disable specified subsystem.	Boolean	1 bit	Expert and Engineer	Alarm	

11	INIT - Initialize the module.	String	4 Bytes	Expert and Engineer	Status	
12	RELEASE -ALL PROCESSES RUNNUNG ,Killing the processes in order manner	String	7 Bytes	Expert and Engineer	Status	

ii) <u>Monitoring Parameters of ABC Subsystem (ENGINEER GUI):</u>

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SEE THE PARAMETER	ALARMS/STATUS	REMARK
1	STATUS- Status of the ABC ON/OFF	Boolean	Logic 1/0	All		
2	LINK status of the link to the ABC (open/close)	Boolean	Logic 1/0	All		
3	TIME - Time on the ABC clock	String		All		
4	VERSION ABC version	Float		All		
5	4SUBSYS list of subsystems that the ABC is currently configured to control	String		All		
6	CURRENTSET current parameter set (e.g. “SET A”).	char	1 Byte	All		
7	MODE current mode (local/remote)	char	1 Byte	All		

iii) <u>Controlling Parameters of ABC Subsystem (USER GUI):</u>

SR. NO.	SETTABLE PARAMETERS OF THE SUBYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	ALARMS/STATUS	REMARK
1	REBOOT- To reboot the ABC (power on if shut off)	String	6 Bytes	Expert and Engineer	Alarm	
2	SHUTDOWN- Shutdown the ABC computer	String	8 Bytes	Expert and Engineer	Alarm	
3	OPEN- Open the link to the ABC	String	4 Bytes	Expert and Engineer	Alarm	

4	CLOSE- Close the link to the ABC	String	5 Bytes	Expert and Engineer	Alarm	
5	SETTIME set the ABC time	String		Expert and Engineer	Status	
6	SHELL open a remote shell on the ABC (essentially this command will execute an "ssh" command on the ABC, the user will be prompted for a password by the ABC).	String	5 Bytes	Expert and Engineer	Status	
7	SAVE(a) save the current settings of all subsystems that the ABC is configured for into "SET A" (could have N (=4?) possible sets)	String	7 Bytes	Expert and Engineer	Status	
8	RESTORE(a) settings from "SET A"	String	10 Bytes	Expert and Engineer	Status	
9	MODE Local/remote. In the local mode a CMS User Interface is opened on the local console. In the remote mode, the control is via Ethernet TCP/IP to the CMS	char	1 Byte	Expert and Engineer	Alarm	
10	Enable/Disable specified subsystem.	Boolean	1 bit	Expert and Engineer	Alarm	
11	INIT - Initialize the module.	String	4 Bytes	Expert and Engineer	Status	
12	RELEASE -ALL PROCESSES RUNNUNG ,Killing the processes in order manner	String	7 Bytes	Expert and Engineer	Status	

iv) <u>Monitoring Parameters of ABC Subsystem (USER GUI):</u>

SR. NO.	PARAMETERS TO BE MONITORED	TYPE OF PARAM	SIZE OF PARA	PRIVILEGE TO SEE THE	ALARMS/STATUS	REMARK

		ETER	**METER**	**PARAMETER**		
1	STATUS - Status of the ABC ON/OFF	Boolean	Logic 1/0	All		
2	LINK status of the link to the ABC (open/close)	Boolean	Logic 1/0	All		
3	TIME- Time on the ABC clock	String		All		
4	VERSION ABC version	Float		All		
5	4SUBSYS list of subsystems that the ABC is currently configured to control	String		All		
6	CURRENTSET- Current parameter set (e.g. “SET A”).	char	1 Byte	All		
7	MODE current mode (local/remote)	char	1 Byte	All		

6.7.7 SENTINEL Subsystem:

i) <u>Controlling/Monitoring Parameters of SENTINEL Subsystem (ENGINEER GUI):</u>

SR NO	MONITORING/CONTROLLING PARAMETRES OF THE SUBYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE	ALARMS /STATUS	REMARK
1	Three phase voltage	Integer	2 bytes	All	Audio-Visual-Status	Alarm if limit is exceeded.
2	Three phase Current	Integer	2 bytes	All	Audio-Visual-Status	Alarm if limit is exceeded.
3	Three phase frequency	Integer	2 bytes	All	Audio-Visual-Status	Alarm if limit is exceeded.
4	Temperature of Antenna control room If Temp exceeds 32 degree Celsius switch off the entire antenna sub systems	Boolean	0 or 1	All	Control	Alarm if limit is exceeded.
5	Temperature of Antenna control room	Float		All	Status	Monitoring actual temp value

6	Fire and Smoke If detected switch on the video system and if it is a worst scenario and no one is around switch on the CO_2 flashing	Boolean	0 or 1	Engineer (Control)	Alarm	
7	Intruder alarm for door(IR based)	Boolean	0 or 1	All	Audio-Visual-Alarm	
8	Vibration sensor (peizo electric)	Boolean	0 or 1	All	Audio-Visual-Alarm	
9	Humidity of wind If it is not in specs switch off the entire system.	Float		All	Audio-Visual-Status	
Note:	**Provide acknowledgement for each of the control action initiated**					

ii) Controlling/Monitoring Parameters of SENTINEL Subsystem (USER GUI):

SR NO	MONITORING/CONTROLLING PARAMETRES OF THE SUBYSTEM	TYPE OF PARAMETER	SIZE OF PARAMETER	PRIVILEGE TO SET THE PARAMETER	ALARMS/STATUS	REMARK
1	Three phase voltage	Integer	2 bytes	All	Audio-Visual-Status	Alarm if limit is exceeded.
2	Three phase Current	Integer	2 bytes	All	Audio-Visual-Status	Alarm if limit is exceeded.
3	Three phase frequency	Integer	2 bytes	All	Audio-Visual-Status	Alarm if limit is exceeded.
4	Temperature of Antenna control room If Temp exceeds 32 degree Celsius switch off the entire antenna sub systems	Boolean	High/Low	All	Control	Alarm if limit is exceeded.

5	Fire and Smoke If detected switch on the video system and if it is a worst scenario and no one is around switch on the CO2 flushing	Boolean	High/Low	All	Status	Monitoring actual temp value
6	Intruder alarm for door(IR based)	Boolean	High/Low	NA	Alarm	
7	Vibration sensor (piezoelectric)	Boolean	High/Low	All	Audio-Visual-Alarm	
8	Humidity of wind If it is not in specs switch off the entire system.	Float	1	All	Audio-Visual-Alarm	
Note:	**Provide acknowledgement for each of the control action initiated**					

7 System performance requirements

7.1 Overview

The overall telescope has several real time response requirements. For example, for accurate control of the position as per the tracking requirement, the servo controller has to respond in real time to changes in load condition. Similarly the sentinel functionality demands a hard real time response to events such as fire etc.

Considering the overall tiered architecture, the hard real time response requirements, that are independent of a user input should be met at the lowest level of control, viz at the hardware controller / or perhaps the hardware driver / interface layer. However, it is important for the CMS to be reasonably responsive in its ability to update the system status and transmit user commands to the layer below. This is because the overall master control and also fine tuning of the telescope operations are under human user control and the telescope is not a completely autonomous system, and significant delays in the presentation of alarms, status and processing of user commands can affect the smooth functioning of the telescope.

From a performance point of view, the telescope controls are thus analogous to a distributed control system with hard real time requirements being met at the level of the distributed controllers (hardware layer), and the supervisory control layer (in this case the CMS) having reasonable bounded response requirements.

7.2 Performance requirements:

The response from the CMS can be characterized as either:

i) Round time taken for a command to be triggered at the presentation tier, i.e. GUI screen or message, its receipt by the lowest control layer, i.e. hardware controller and the response or acknowledgement being reflected in the presentation tier.*

ii) Time taken to reflect a system event in the presentation tier, i.e. GUI screen or message on the command console.

Generally the requirement of i) above is the more stringent one since it reflects a two way traversal of command / control

*Note: It should be noted that this is valid when the hardware controller returns a response / acknowledgement within a few tens of milliseconds or so. Further it is not expected that the commanded action should be taken by the hardware controller within the specified time – that is dependent on the design of the hardware program and is beyond the control of the CMS

Based on discussions with the IUCAA / NCRA stakeholders, the bounds proposed for the CMS response

i) A maximum of 3 seconds for the updating of non-alarm events or actions;

ii) A maximum of 1 second for the updating of alarm events from hardware into the GUI

Further from a usability perspective,

iii) A maximum of 1 second should be taken to present a new page on the GUI when the context demands, and a maximum of 2 seconds to update all the information on the new page (except trend plots which would be continuously updating)

Although these limits are somewhat flexible, it is believed that a specification of this order of magnitude is both necessary and sufficient for an efficient user control of the telescope.

The optimization of the architecture of critical information flows and handling components as well as needed calculations of processing performance would have to be undertaken in the design phase. The specification is however believed achievable for the architecture specified.

It should be noted that these performance specifications are those seen by intranet clients; for remote clients accessing the CMS through the internet; the performance would depend on the internet speed.