LBT-ADOPT
TECHNICAL REPORT

Doc.No : 481f341a
Version : 1.0
Date : 16 July 2009

AOS
The Complete Guide

Luca Fini, Alfio Puglisi, Lorenzo Busoni

CAN: 481f341a

INAF - Osservatorio di Arcetri
http://adopt.arcetri.astro.it/

ABSTRACT

This report is a complete guide to the AOS both from the functional point of view and for many implementation aspects. It is intended to be useful in order to understand AOS functionalities, so that instrument software programmers can better exploit AOS capabilities for their purposes, system programmers can be helped in troubleshooting or maintenance activities and telescope operators can better know how to use the AdOpt subsystem.
This report also includes all the information that was previously covered in document CAN 481f340 [1]

Contents

I AOS Software Description **4**

1 Introduction **4**

1.1 AO system operating modes 5

2 AOS Architecture **5**

2.1 Main 5

2.2 AOSSubsystem 6

2.3 AOSClient 6

3 AOSAoApp **6**

3.1 Handlers 7

3.1.1 `varnotify_hndl` 7

3.1.2 `hexapod_hndl` 8

3.1.3 `housekeep_hndl` 8

3.1.4 `default_hndl` 8

3.2 AOSAoApp::Run() 8

4 Variables **9**

5 Events **10**

6 Commands **12**

7 Mode Offload **13**

8 TV image frames **14**

9 AOS GUI **15**
9.1 Information panel 15
9.2 Command panel 15

10 Configuration items **16**

II AOS Operational Functions **19**

11 Introduction **19**

12 Commands accepted by AOS from other TCS subsystems **19**
12.1 StartObs 20
12.2 PresetFlat 21
12.3 PresetAO 22
12.4 AcquireRefAO 23
12.5 CheckRefAO 24
12.6 RefineAO 24
12.7 ModifyAO 25
12.8 StartAO 25
12.9 OffsetXY 26
12.10OffsetZ 26
12.11CorrectModes 26
12.12Stop 27
12.13Pause 27
12.14Resume 27
12.15Terminate 27
12.16UserPanic 27

13 Commands issued by AO-Supervisor to request services from TCS **28**
13.1 Housekeeping commands 28
13.1.1 LogItem 28
13.1.2 LogOn/LogOff 29
13.1.3 DbgLevel 29
13.1.4 Warning 29
13.1.5 Error 29
13.1.6 Panic 29
13.2 Engineering mode commands 30

13.2.1 RequestService 30
13.2.2 EndOfService 30
13.2.3 Stop 30
13.2.4 Hexapod Commands 31

A Source Files Identification **32**

B Interface with AO-Supervisor **32**

Glossary of terms and acronyms

AdSec. The Adaptive Secondary Mirror. In this context usually refers to the group of *AO-Supervisor* components controlling the hardware devices related to the secondary mirror.

ADAM. Ethernet controlled digital output. Used to enable various devices within the AdSec.

AdSec Computer. The workstation running the *AO-Supervisor* components which control the Adaptive Secondary hardware.

Adsec-Arb. The Adaptive Secondary Arbitrator. A component of the *AO-Supervisor* which executes commands related to the *AdSec* coordinating the operations of the hardware devices in the Adaptive Secondary. Commands to *AdSec-Arb* may come either from a specific GUI or from *AO-Arb*.

AO System. The hardware and software components of the LBT first light Adaptive Optics System. Includes the Wavefront Sensor, the Adaptive Secondary Mirror, the *AO Computer* and some auxiliary devices (such as networking hardware) and includes the *AO-Supervisor* and the real-time software.

AO-Arb. The AO Arbitrator. A component of the *AO-Supervisor* which manages the execution of high-level commands, coordinating the operations of *WFS-Arb* and the *AdSec-Arb*. Commands to *AO-Arb* may come either from a specific interface or from *AOS*.

AO Computer. The computer (or farm of computers) running the *AO-Supervisor*.

AO Console. The operator console of the *AO Computer*.

AO-Supervisor. The software system which manages all the components of the *AO System*

BCU. Basic Control Unit. Electronics board used as basic building block for most of the electronics in the AO System (see [2]).

BCU 47. The BCU used as frame grabber for the CCD 47.

C-BCU. Crate BCU. The six BCUs which controls the AdSec.

CCD 39. The CCD used for the Wavefront sensor (also: *WFS CCD*).

CCD 47. The CCD used for the *TV*.

Copley. Motor driver for the Bayside stages.

Fastlink. The real-time data communication link between the WFS and the AdSec.

FLAO. First Light AO system for the LBT. The whole AO system for the LBT, including both hardware and software components.

Flowerpot. Auxiliary unit to control the calibration source and the related optics (cube beam splitter).

MsgD. Message Dispatcher, the *AO-Supervisor* message dispatching daemon.

RTDB. AO Real Time Database, the *AO-Supervisor* own variable repository. Its functionalities are supported by `MsgD`.

S-BCU. Switch BCU. The BCU operating as input source switch for the AdSec.

TTM. Tip-Tilt Mirror. A small mirror used to modulate the pupil image un top of the WFS pyramid.

TV. Technical Viewer. An auxiliary CCD camera used by the Wavefront Sensor to acquire the reference star.

WFS. The Wavefront Sensor. In this context usually refers to the software subsystem controlling the hardware devices related to the wavefront sensor.

WFS CCD. The CCD used in the *WFS* to measure the light wavefront deformation (also: *CCD 39*).

WFS Computer. The workstation running the *AO-Supervisor* components which control the Wavefront Sensor hardware.

WFS-Arb. The WFS Arbitrator. A component of the *AO-Supervisor* which executes commands related to the *WFS* coordinating the operation of the hardware devices of the WFS. Commands to *WFS-Arb* may come either from a specific GUI or from *AO-Arb*.

Foreword

This document is divided into two parts. Part I describes AOS architecture and code structure, and includes details about the interaction between AOS and TCS, from one side and the AO-Supervisor, from the other. Part II, describes the operational functions implemented by AOS to support the Adaptive Optics System. This second part is directly derived from a previous document (CAN 481f340 [1]) which is now obsolete.

Part I

AOS Software Description

1 Introduction

The Adaptive Optics Subsystem (AOS) is a standard component of the LBT Telescope Control System (TCS) whose purpose is to interface the TCS to the AO-Supervisor[1] It provides all the functionalities needed for the interaction between the LBT Adaptive Optics system and the rest of the telescope, including instruments.

Figure 1 shows the relationships between relevant software components.

The AOS operates essentially as a thin interface layer between the AO-Supervisor and the TCS as a whole. By providing TCS with a small number of AO related commands it allows the instrument software to operate the AO System during an observation.

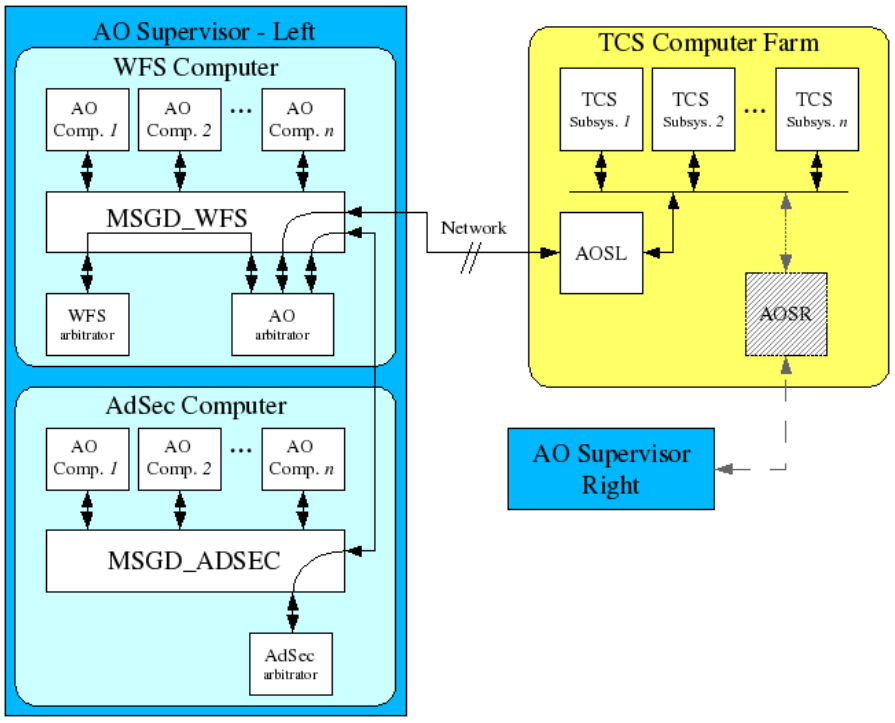

Figure 1: Interaction between the TCS and the AO Software

[1]Please note that in the following pages we will use the acronym **AOS** to indicate the Adaptive Optics Subsystem, i.e.: the software component of the TCS which communicates which the AO-Supervisor, while we will use the term **AO-System** to indicate the Adaptive Optics System as a whole. The term **AO-Supervisor** indicates the software controlling the AO-System.

In general, communication between AOS and the AO-Supervisor will be based on the two mechanisms defined in AO-Supervisor: access to `RTDB` variables and direct messages sent through the `MsgD` to some AO-Supervisor component.

AOS will maintain a subset of the TCS status variables reflected in the `RTDB` so that AO-Supervisor components have them available, and will have read access to the full set of AO-Supervisor status variables.

From the AO System side, the interaction with TCS is managed by a specific process the "AO Arbitrator" (AO-Arb), whose main function is to receive commands from the TCS and coordinate the execution of needed functions by sending subcommands to the processes controlling the WFS and the AdSec: WFS-Arb and AdSec-Arb, respectively.

A description of functionalities implemented by the AO Supervisor and details on the implementation of the AO Arbitrator can be found in [3].

1.1 AO system operating modes

When supporting an observation, the AO System has four different operating modes:

- **FIX-AO**, when operating in "seeing limited" mode. In other words, this is not an "adaptive" mode in that the shape of the adaptive mirror is fixed, but is anyway supported by the AO System.
- **TTM-AO**, when operating in adaptive mode, but correcting only the tip-tilt components of the atmosphere aberration.
- **ACE-AO**, when operating in full adaptive mode with an automatic selection of AO loop parameters.
- **ICE-AO**, when operating in full adaptive mode and providing means to allow the observer to adjust AO loop parameters.

2 AOS Architecture

The AOS is a standard subsystem of TCS. As such it is structured as others TCS subsystems. It is divided essentially into three source modules, as detailed in the following sections.

2.1 Main

The Main module[2] performs the following three steps:

1. Gets configuration data; e.g.: the IP number of the machine where the AO-Supervisor `MsgD` is running, the initial debug level, etc. (for a list of all configuration parameters see section 10).

[2]Main module code is in files `Main.cpp`, `Main.hpp`.

2. Starts the `AOSSubsystem`'s thread.
3. Initializes the commands.

2.2 AOSSubsystem

`AOSSubsystem`[3] is executed in its own thread (as all other TCS subsystems) and it is structured as any other TCS subsystem. It provides an `execute()` method which is where part of the job is done.

`execute()` is essentially a loop which controls the communication rendez-vous with the `MsgD`.

The loop performs three simple steps:

1. Instantiates an `AOSAoApp` object.
2. Start `AOSAoApp` by executing its `Exec()` method.
3. Sleeps 1 second.

The `AOSAoApp` object is derived from the `AOApp` class provided by the AO-Supervisor software, it connects as a client to the `MsgD` and implements the communication exchange with the AO-Supervisor.

The above described loop has the purpose of continuously retrying to connect to the `MsgD` until it is ready to accept `AOSAoApp` as a client. When the communication is activated the `Exec()` method will not return.

Because most of the functionality of the AOS is embedded into the class `AOSAoApp`, it is described in deeper details in section 3.

2.3 AOSClient

`AOSClient`[4] implements the class to be used by clients (from the TCS side) to send commands to the AOS. This follows closely the TCS standard for client communication and its purpose is simply to receive from other TCS subsystems commands to be directly translated into commands to be delivered to the AO-Supervisor (see sect. 6).

3 AOSAoApp

`AOSAoApp`[5] is the class at the heart of the AOS; it is derived from the standard `AOApp` class used to implement clients for the AO-Supervisor and provides all the functionalities needed to the communication between AO-Supervisor components. It implements a number of base functionalities such as message queue management, automatic replies to several housekeeping messages, error logging, and more.

[3] AOSSubsystem code is in files: `AOS.cpp`, `AOS.hpp`.
[4] The source code is in files: `client/AOSClient.cpp` and `client/AOSClient.hpp`.
[5] The related source code is in files: `AOSAoApp.cpp`, `AOSAoApp.hpp`.

An application based on AoApp is intrinsically multi threaded, although this is hidden into the implementation and multi-threading is not directly managed by the application programmer.

After being instantiated AOSAoApp executes an Exec() method which performs a few steps:

1. Installs handlers for asynchronous messages coming from the AO-Supervisor.
2. Connects to the MsgD.
3. Executes the Run() method.

The Run() method is where the functionalities of AOSAoApp are implemented and is essentially a loop which terminates when the communication with MsgD is interrupted for any reason or on some explicit termination command.

While connected as an MsgD client the AOS registers itself either with the name AOSL00 (left side) or as AOSR00 (right side).

As seen in section 2.2, if the initialization process fails (e.g.: because the MsgD is not running or not ready to accept clients) the Run() method is not executed and the control returns to to the caller, which in turn will retry the instantiation after a proper delay.

3.1 Handlers

As part of AOSAoApp initialization the (overridden) method install_handlers()[6] is called to install handlers for specific messages received from the AO-Supervisor.

Each handler runs in its own thread and receives a predefined subset of the messages sent by MsgD to be processed.

Here follows a brief description of each handler.

3.1.1 varnotify_hndl

The variable notification feature of the RTDB is used to mirror relevant variables from the RTDB into the TCS DD. As part of its initialization the AOSAoApp registers with the MsgD to be notified of variation on a specified set of variables.

Whenever an AO-Supervisor client writes to one of the variables this handler receives a notification (which includes the variable value). Upon notification, the handler writes the variable value into the corresponding variable in TCS DD.

A list of notified variables can be found in section 4.

[6] The related source code is in files: AOSHandlers.cpp and AOSHandlers.hpp.

3.1.2 hexapod_hndl

This handler manages hexapod related commands sent from the AO-Supervisor in ENGINEERING mode of operation.

When an hexapod command is received, the handler checks whether the command can be accepted and, if positive, converts it into a command to OSS to be applied to the hexapod.

An acknowledge message is sent back to the sender of the command.

Details on hexapod commands are covered in a following section (see section 6).

3.1.3 housekeep_hndl

This handler manages a set of housekeeping commands, mostly useful for debugging and troubleshooting operations. For a list of defined housekeeping commands see section 6.

3.1.4 default_hndl

The default handler will receive all messages not trapped by other handlers. It is used to manage the `TERMINATE` command[7].

All other commands are unexpected and thus will only generate warnings.

3.2 AOSAoApp::Run()

After a successful instantiation and `AOSAoApp` and its registration as a client of `MsgD`, the `Run()` method is executed.

The `Run()` method is the main loop of the AOS, despite the fact that most of the work is performed elsewhere, i.e.: by handlers for commands originating from the AO-Supervisor, and by the TCS Command Interface code for commands originating from other TCS subsystems.

The task of the `Run()` method is to implement the variable mirroring from the TCS `DD` to the `RTDB`. This is done by periodically polling values from the `DD` and writing them into the `RTDB` in a loop as follows:

1. Read from `RTDB` the variable `L.AOSL00.SERVSTAT`[8].
2. If not OK: exit
3. Iterate on the polled variable table and write values to `RTDB`.
4. Sleep for the polling interval time.

The first step is essentially a watchdog mechanism to ensure that the `MsgD` is communicating properly and is also useful to identify possible status changes of the AO-Supervisor.

[7] All AO-Supervisor clients are required to properly manage the `TERMINATE` command. It is not intended for normal use but only in debugging or troubleshooting operations.

[8] For the right side it would be `R.AOSR00.SERVSTAT`.

4 Variables

AOS, alike all other TCS subsystems, uses a dedicated section in the TCS Data Dictionary to hold variables for various purposes.

Table 1: AOS housekeeping variables

aos.side[s].AOSupervisorIP	string	IP address of machine running the AO-Supervisor
aos.side[s].connected	boolean	Indicates when the AOS is connected to `MsgD`
aos.side[s].lasterror	int	Last error code
aos.side[s].running	boolean	Indicates when the AOS is connected to `MsgD` and communicating
aos.side[s].servstat	string	Current AO-Supervisor service status
aos.side[s].wfs.tv_filename	string	Filename for TV image store

AOS variables can be divided into four groups:

- **Housekeeping variables**, used for various housekeeping and information tasks, listed in table 1.
- **TCS variables**, i.e.: variables defined into TCS `DD` which are mirrored into AO-Sup `RTDB`. They are listed in table 2. For these variables, values are updated by periodically polling the TCS Data Dictionary; the polling mechanism is described in section 3.2.

  Note: Most variables have direct correspondence between the `DD` and the `RTDB`. A few are functions of more than one `DD` variable (e.g.: `ISTRACKING` is a boolean value derived from `mcs.azDrive.trackingMode`, `mcs.elDrive.trackingMode` and `mcs.onSource`.
- **RTDB variables**, variables defined into the AO-Sup `RTDB` which are mirrored into TCS `DD`. These variables are divided into subgroups and are listed in the following tables: table 3, for variable related to the AO system as a whole, table 4, for variables related to the AdSec, table 5, for variables related to the WFS.

  Values for these variables are updated as needed by means of the notification mechanism provided by `MsgD`.
- **GUI variables**, variables used to support the command panel of the AOS GUI (see also section 9.2). They are listed in the following tables: table 6, for the `PresetFlat` command; table 9, for the `PresetAO` command; table 7, for the `RefineAO` command; table 8, for the `ModifyAO` command; table 10; for the `OffsetXY` and `OffsetZ` commands and table 11, for the `StopAO` command.

  Variable meaning and use is not detailed in these tables because each variable correspond to a command argument which is described in the AOS command section (see section 12).

Notes on variables:

1. In the tables variable types are indicated as string, int, real. They are currently implemented in the AOS with corresponding types defined for the `DD`: string, int, double

Table 2: Telescope related variables

C.AOSx00.AMBIENT.HUMIDITY	real	`ecs.weatherStation.smt.humidity` Ambient humidity
C.AOSx00.AMBIENT.PRESSURE	real	`ecs.weatherStation.smt.ambientPressure` Ambient pressure.
C.AOSx00.AMBIENT.TEMPERATURE	real	`ecs.weatherStation.smt.ambientTemperature` Ambient temperature
C.AOSx00.AMBIENT.WINDSPEED	real	`ecs.weatherStation.smt.wsWindSpeed` External wind speed
C.AOSx00.AMBIENT.WINDDIRECTION	real	`ecs.weatherStation.smt.wsWindDirection` External wind direction
x.AOSx00.HEXAPOD.ABS_POS	real[6]	`oss.side[0].secondary_mirror.abs_pos` Hexapod absolute position
x.AOSx00.HEXAPOD.STATUS	int	*Function* Hexapod status.
x.AOSx00.ROTATOR.ANGLE	real	`pcs.mcsDemand.rotatorDemand.angle` Rotator angle
x.AOSx00.ROTATOR.VELOCITY	real	`pcs.mcsDemand.rotatorDemand.rotatorVelocity` Rotator velocity
C.AOSx00.TEL.AZIMUTH	real	`pcs.pointingStatus.achieved.azimuth` Current telescope azimuth.
C.AOSx00.TEL.ELEVATION	real	`pcs.pointingStatus.achieved.elevation` Current telescope elevation.
C.AOSx00.TEL.ISTRACKING	int	*Function* When not zero, indicates that the telescope is tracking on the requested object.

2. In variable names related to TCS `DD` the character "`s`" is used to indicate the telescope side (current corresponding values are 0 for left side and 1 for right side).

3. In variable names related to AO-Sup `RTDB` the character "`x`" also indicates the side. Its value may be either `L` or `R` for left and right side, respectively. "Non sided" variables have the prefix `C`.

4. The source code relevant for the management of variables can be found in files: `VarUtils.cpp` and `VarUtils.hpp`.

5 Events

`AOSAoApp` uses the standard TCS event mechanism to log relevant events. Examples are: command execution, mode offload requests and errors at various levels of severity.

Table 3: AO System related variables

x.AO.CORRECTEDMODES	int	`aos.side[s].ao.correctedmodes` Number of corrected modes
x.AO.MODE	string	`aos.side[s].ao.mode` Current observation mode (`FIX_AO`, `TT_AO`, `ACE_AO`, `ICE_AO`.)
x.AO.MSG	string	`aos.side[s].ao.msg` Generic message string
x.AO.STATUS	string	`aos.side[s].ao.status` Adaptive Optics system status
x.AO.STREHL	real	`aos.side[s].ao.strehl` Estimated Sthrel ratio
x.AO.WFS_SOURCE	real	`aos.side[s].ao.wfs_source` Wavefront sensor currently in use. Default is the WFS for first light AO. Other sources of wavefront data will be used for other optical configurations.

Table 4: Adaptive Secondary specific variables

x.ADSEC.LOOPGAIN_LIMITS	real[2]	`aos.side[s].adsec.loopgain_limits` Loop gain value limits
x.ADSEC.LOOPGAIN	real	`aos.side[s].adsec.loopgain` Current loop gain
x.ADSEC.LOOPSTATUS	string	`aos.side[s].adsec.loopstatus` Current loop status
x.ADSEC.MSG	string	`aos.side[s].adsec.msg` Generic message related to AdSec
x.ADSEC.OFFLOAD	int[22]	`aos.side[s].adsec.offload` AdSec offload hint (see sect. 7)
x.ADSEC.OFLMODES	real[22]	`aos.side[s].adsec.oflmodes` AdSec offload vector (see sect. 7)
x.ADSEC.SHAPE	string	`aos.side[s].adsec.shape` Currently selected mirror shape
x.ADSEC.STATUS	string	`aos.side[s].adsec.status` AdSec status

When the AO System is in either OBSERVATION or in ENGINEERING mode, logging information is also sent to to the `MsgD`.

Table 5: Wavefront Sensor specific variables

x.WFS.CCDBIN	int	`aos.side[s].wfs.ccdbin` WFS CCD binning
x.WFS.CCDFREQ_LIMITS	real[2]	`aos.side[s].wfs.ccdfreq_limits` WFS CCD frequency limits
x.WFS.CCDFREQ	real	`aos.side[s].wfs.ccdfreq` WFS CCD current frequency value
x.WFS.COUNTS	int	`aos.side[s].wfs.counts` WFS CCD average counts per frame
x.WFS.FILTER1	string	`aos.side[s].wfs.filter1` WFS filter 1 position
x.WFS.MOD_AMPL	real	`aos.side[s].wfs.mod_ampl` WFS TT-mirror modulation amplitude
x.WFS.MSG	string	`aos.side[s].wfs.msg` WFS related generic message
x.WFS.NO_SUBAPS	int	`aos.side[s].wfs.no_subaps` Number of subapertures in current WFS configuration
x.WFS.PYRAMID_POS	int[2]	`aos.side[s].wfs.pyramid_pos` Position of WFS pyramid with respect to TV CCD
x.WFS.SOURCE	string	`aos.side[s].wfs.source` WFS CCD current light source
x.WFS.STATUS	string	`aos.side[s].wfs.status` Current WFS status
x.WFS.TV_BINNING	int	`aos.side[s].wfs.tv_binning` TV CCD current binning
x.WFS.TV_EXPTIME	real	`aos.side[s].wfs.tv_exptime` TV CCD current exposure time
x.WFS.TV_FILTER1	string	`aos.side[s].wfs.tv_filter1` TV CCD current fiter 1 selection
x.WFS.TV_FILTER2	string	`aos.side[s].wfs.tv_filter2` TV CCD current fiter 2 selection

6 Commands

Commands are extensively described from the functional point of view in the second part of this report (see sections: 12, 13), here we will cover some implementation aspects which may be useful to understand the AOS code.

We may divide the set of AOS commands in three groups:

- **Commands delivered to the AO-Supervisor**. This group of commands is used mostly in OBSERVATION mode to control the AO System operation. They are received as standard TCS commands as defined in the AOS interface[9] and then they are simply translated into exactly

[9]See code in source file `.../aos/commands/AOSCommands.cpp`.

Table 6: Variables for `PresetFlat` command

aos.side[s].presetflat.flatspec	string

Table 7: Variables for `RefineAO` command

aos.side[s].refine.method	string

Table 8: Variables for `ModifyAO` command

aos.side[s].modify.f1spec	string
aos.side[s].modify.f2spec	string
aos.side[s].modify.itime	real
aos.side[s].modify.nbins	int
aos.side[s].modify.nmodes	int
aos.side[s].modify.ttmod	real

Table 9: Variables for `PresetAO` command

aos.side[s].presetao.aomode	string
aos.side[s].presetao.cindex	real
aos.side[s].presetao.elev	real
aos.side[s].presetao.gravang	real
aos.side[s].presetao.mag	real
aos.side[s].presetao.r0	real
aos.side[s].presetao.rocoord1	real
aos.side[s].presetao.rocoord2	real
aos.side[s].presetao.rotang	real
aos.side[s].presetao.skybrgt	real
aos.side[s].presetao.slnull	string
aos.side[s].presetao.socoord1	real
aos.side[s].presetao.socoord2	real
aos.side[s].presetao.wfs	string

Table 10: Variables for `OffsetXY` and `OffsetZ` commands

aos.side[s].offset.x	real
aos.side[s].offset.y	real
aos.side[s].offset.z	real

Table 11: Variables for `StopAO` command

aos.side[s].stop.msg	string

corresponding calls to the AO-Arb commands as defined in the related interface[10]. The only operations added by AOS is the logging to the TCS event system and possibly some reorganization of arguments. See section 12.

- **Housekeeping commands received from the AO-Supervisor**. A group of commands which are essentially used for debugging and troubleshooting. Generally speaking, they affect logging level, record messages in the event logging system of TCS and so on. See section 13.1 for detailed description.
- **Engineering mode commands received from the AO-Supervisor**. Commands to operate some telescope device to be used in engineering mode. Currently they only include commands to allow the AO Supervisor to directly operate the hexapod during some engineering operations. They are implemented by a specific command handler (see section 3.1.2). For a detailed description of defined commands, see section 13.2.

7 Mode Offload

Mode offload is needed during Adaptive Optics closed loop when the AdSec gets near to some physical limit, in order to offload the error on the first modes to other telescope devices. A typical example is

[10] See AO-Arb command implementation code in file `AOS.cpp`.

when the telescope accumulates tracking errors: if the adaptive loop is on, those errors are compensated by the adaptive secondary. As errors increase the AdSec will finally get close to intrinsic limits in the maximum amount of tip-til it can compensate. The error must thus be lowered by adjusting the telescope pointing.

The mode offload mechanism is based on the use of two variables defined in the AO-Supervisor `RTDB`: `x.ADSEC.OFLMODES`[11] (an array of 22 double values) and `x.ADSEC.OFFLOAD` (an array of 22 integer values), and is implemented as follows:

1. The AO-Supervisor diagnostic subsystem is continually monitoring the modal errors on the AdSec mirror.
2. The modal errors computed by the diagnostic subsystem are written into the `RTDB` variable `x.ADSEC.OFLMODES` as soon as they are computed.
3. Whenever the value of some component of the modal error vector goes beyond the first level threshold, the corresponding component of the integer array `x.ADSEC.OFFLOAD` is set to 1.
4. A second level threshold is also checked. If this is exceeded the value of the corresponding element in `x.ADSEC.OFFLOAD` is set to 2.
5. The two arrays are mirrored by AOS into the corresponding variables in TCS `DD` (see table 4), respectively: `aos.side[s].adsec.oflmodes` and `aos.side[s].adsec.offload`.
6. AOS will also detect any change in `aos.side[s].adsec.offload` and will issue a specific command to PSF whenever the value changes.

Based on the above behavior, PSF can adopt various strategies for mode offloading:

1. It may continuously poll the variable `aos.side[s].adsec.oflmodes` and offload errors as soon as they grow up.
2. It may continuously poll the variable `aos.side[s].adsec.offload` and offload errors as soon as any value in it grows to 1 or to 2.
3. It may wait for commands issued by AOS.

8 TV image frames

Due to technical details of the data management in the TCS middle ware, it is not practical to use either the parameter passing mechanism of TCS commands or the `DD` to transmit even moderately large data sets such as the 256x256 image from the TV.

For this reason an indirect mechanism is used to send TV frames to be displayed on the AOS GUI: frames from the TV are stored by the AO Supervisor into the `RTDB` variable `x.WFS.TV_IMAGE` whenever available. The AOS registers to be notified of variable change but instead of writing the data vector into `DD`, it writes the data in a temporary file[12] from which it can be read back by other subsystems (typically the AOS GUI).

[11] `x` is either `L` or `R` depending on telescope's side.
[12] See also section 10.

9 AOS GUI

Although AOS is essentially a thin interface layer to allow TCS subsystems to operate the AO system, and thus it should be essentially controlled by the instrument software, through TCS, it is anyway provided with a GUI which is intended to be up and running at the operator console during observations.

The main task of the AOS GUI is strictly informative: it provides access to a set of parameters which allow the operator to verify the good health of the AO system and the correctness of the operations.

The AOS-GUI includes also a command panel (usually hidden) from which commands usually received from TCS subsystems can be manually sent to the AO Supervisor.

9.1 Information panel

In figure 2 a snapshot of the main information panel of AOS-GUI is shown.

Starting from top we may identify 4 main sections:

1. **Wavefront sensor section**, displaying data related to the Wavefront sensor. It includes a snapshot of the technical viewer[13], a label showing the current status of the WFS and a CCD subsection showing parameters related to the WFS-CCD.
2. **Adaptive secondary section**, containing parameters related to the Adaptive Secondary: a status indicator, the current selection for seeing limited operations.
3. **AO system section**, to the right of the AdSec section, displaying data and parameters related to the whole Adaptive Optics System. It includes an operating mode indicator, the number of corrected modes, three values showing the amount of "offload" currently required[14]. It also include a temporal graph of an image quality indicator (Strehl ratio).
4. **Log & Ctrl section**, containing items related to the management of the AOS subsystem. It will display status information, such as the current command in execution, messages and errors.

9.2 Command panel

Figure 3 shows a snapshot of the command panel available in the AOS-GUI.

The purpose of the command panel is twofold.

It provides the capability to test the AOS and the interactions with the AO Supervisor independently from the rest of TCS and, which is more interesting during the operative life of the system, allows an operator a manual intervention when some procedure fails for unexpected reasons.

The command panel provides a set of buttons which correspond to most of the operating commands defined for the AOS described in section 12.

[13]Due to operating requirements of the WFS subsystem, the technical viewer will receive enough light to generate a useful image only before the AO loop is closed. Thus, the TV image will not be available during the observation.

[14]The two tip-tilt components and the coma.

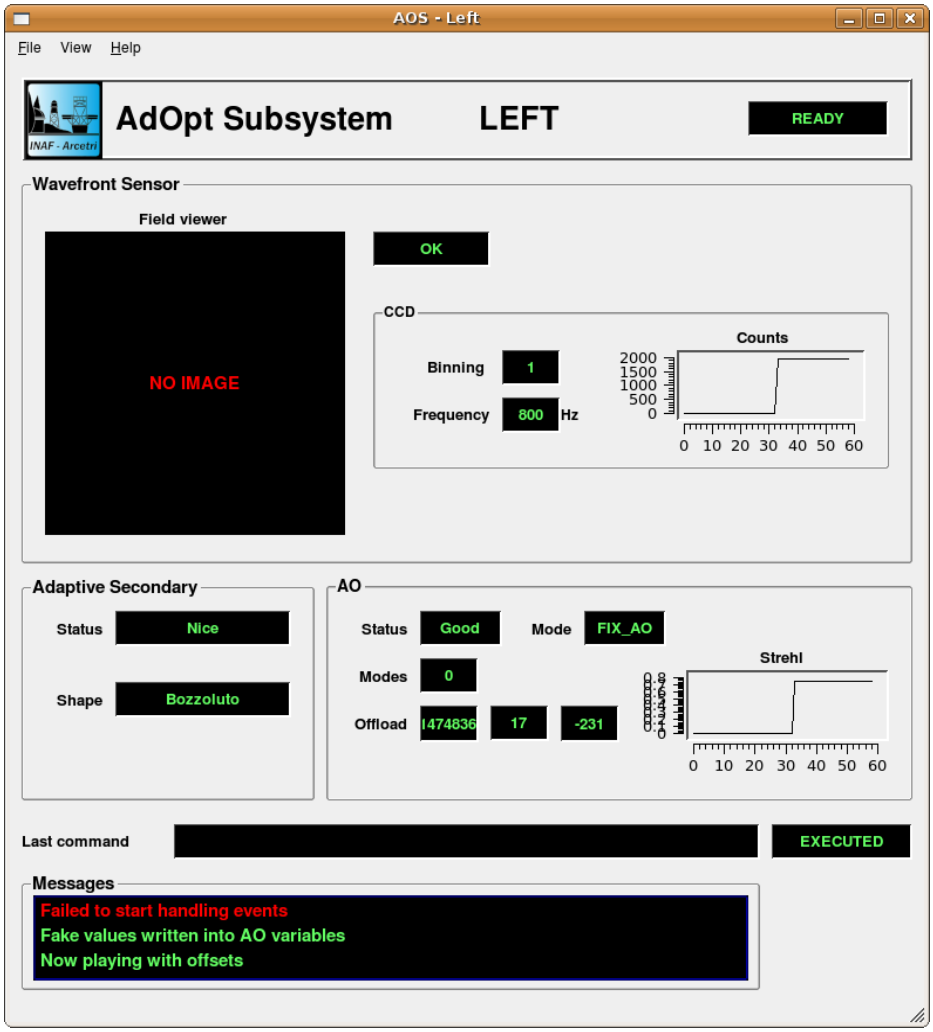

Figure 2: AOS GUI information panel

A few of the commands have associated arguments which may be modified before sending the command. Because the panel is intended either for debugging or for unexpected conditions, there are hardly preliminary checks applied to commands issued from the panel or to values sent as arguments. All the security checks are performed by the AO Supervisor and any illegal or potentially dangerous command is simply rejected with an error message.

10 Configuration items

AOS has a dedicated section in the general configuration file (`lbt.conf`). A list of all parameters is shown in table 12. The naming follows a simple convention: they are all prefixed by either `AOSL` or `AOSR`, respectively, for left and right side.

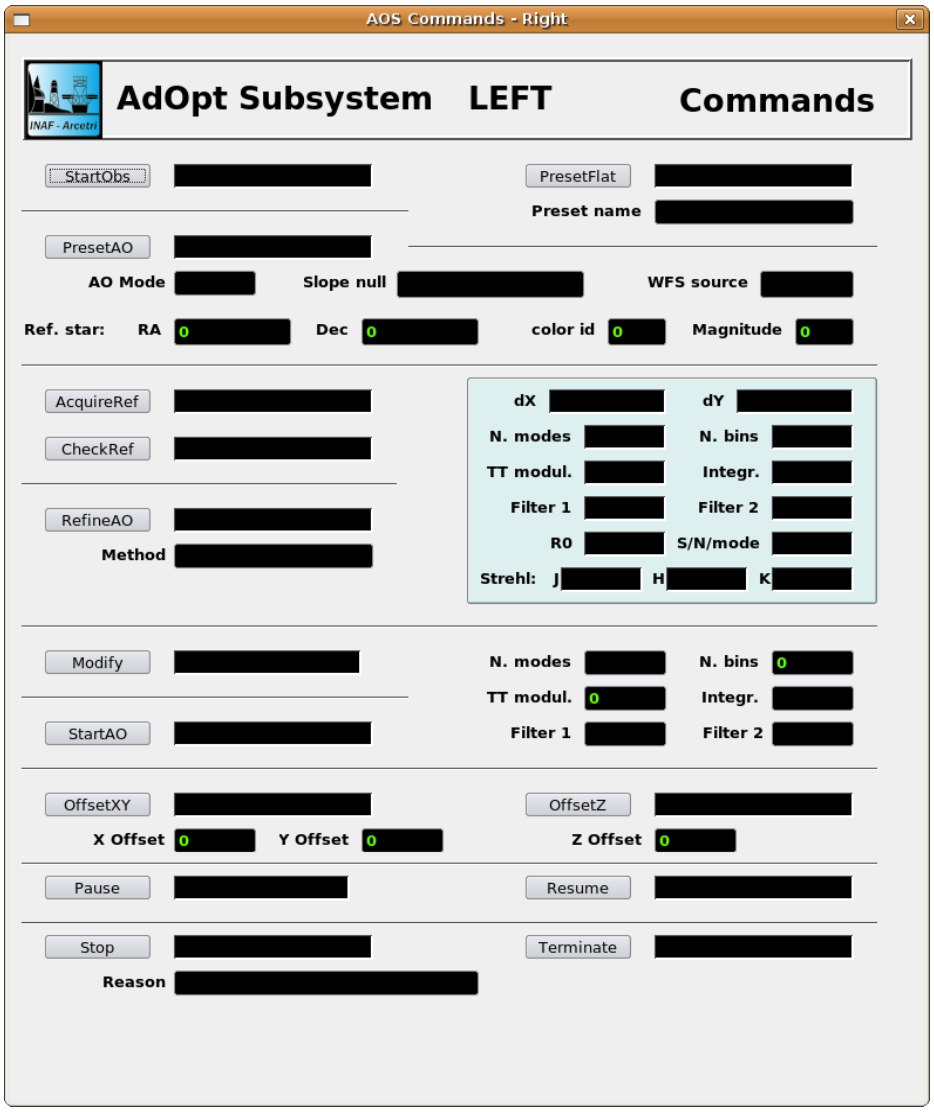

Figure 3: AOS GUI command panel

Item	Type	Description
AOSLMsgDIp AOSRMsgDIp	string	Message daemon IP number. The IP address of the workstation running `MsgD` in the numerical dotted form.
AOSLtv_file AOSRtv_file	string	Basename of temporary file for the TV snapshot image. The procedure will create and update files named `<name>.dat` and `<name>.tmp`
AOSLdbgLevel AOSRdbgLevel	int	AOS debug level. For debugging purposes AOS is provided of an internal log mechanism writing onto syslog. In normal use a debug level of 0 must be used; values 1 and 2 will provide increasingly verbose log.
AOSLsimulation AOSRsimulation	bool	Enable simulation mode when true. Simulation mode is used to perform a limited number of tests on AOS without the need to activate the AO Supervisor.

Table 12: AOS Configuration parameters

Part II

AOS Operational Functions

11 Introduction

AOS, being an interfacing layer between the TCS and AO-Supervisor will both define commands to be used by other subsystems of the TCS to operate the AO System and, on the other side, it will receive messages from AO-Supervisor and translate them into proper commands to be executed by TCS[15].

In the following two sections we describe two set of operational commands in some deeper details: the first set is for commands which are received by the AOS from other TCS subsystem and are converted in messages to be sent to the AO-Supervisor; the second set refers to messages coming from the AO-Supervisor and received by the AOS. The description is essentially from the point of view of messages exchanged between the AOS and the AO-Supervisor. E.g : it may happen that an AO-Supervisor command to the AOS is entirely managed within the AOS itself (e.g.: by simply setting the value of a `DD` variable), or that it might result in a short sequence of commands sent to different subsystems of the TCS; on the other side some function executed by the AOS as the result of a command issued by another TCS subsystem, may result in a short sequence of messages exchanged with the AO-Supervisor.

12 Commands accepted by AOS from other TCS subsystems

This section describes the set of commands which are accepted by AOS from other TCS subsystems in order to support an AO based observation.

Commands to be used by instruments during observation, which are translated by AOS into commands for the AO-Supervisor are listed in table 13 and are described in some details in the following paragraphs. Prior of sending the command message to AO-Supervisor, AOS will check the "status precondition".

Note that whenever necessary for each command the required parameters have been listed. This doesn't necessarily imply that the corresponding values will be sent together with the command: in some cases required values can be available as variables which are continuously mirrored between the AO-Supervisor and AOS (see table 2).

Some commands may have unspecified parameters, i.e.: parameters that may be omitted in a command without affecting the capability of the AO system to operate or values returned by commands which in some cases cannot be evaluated. To declare a parameter not defined throughout the AOS code we use the value `NaN` for real parameters and `MAXINT` for integer ones.

[15]The identification of the set of functions is the result of discussions with various people both from Arcetri and from Tucson: N. Cushing, M. De La Pena, S. Esposito, R. Green, J. Kraus, D. Miller, A. Riccardi, P. Salinari, M. Wagner, with a special mention to J. Hill.

Command	Status precond.	Description
StartObs	READY	Start an observation
PresetFlat	OBSERVATION	Preset AO System for seeing limited operation
PresetAO	OBSERVATION	Preset AO System for adaptive operation
CheckRefAO	OBSERVATION	Returns offset values between reference star actual and nominal position
AcquireRefAO	OBSERVATION	Acquire the reference star and become ready for closing the AO loop
RefineAO	OBSERVATION	Perform optimization of AO loop parameters
ModifyAO	OBSERVATION	Modify some AO loop parameter
StartAO	OBSERVATION	Start the AO mode (i.e.: close the AO loop)
OffsetXY	OBSERVATION	Offset AO pointing
OffsetZ	OBSERVATION	Offset AO focus
CorrectModes	OBSERVATION	Apply mirror shape correction
Stop	OBSERVATION	Stop current operation
Pause	OBSERVATION	Temporarily suspend current operation.
Resume	OBSERVATION	Resume suspended operation.
Terminate	OBSERVATION	Terminate an observation.
UserPanic	OBSERVATION ENGINEERING READY	Emergency shutdown request

Table 13: Commands accepted by AOS from TCS

Most of the commands have an one-to-one relation with AO-Arb commands, i.e.: for each AOS command there is a corresponding command in AO-Arb. In some cases (e.g.: the `PresetAO` command) the two corresponding commands may have a different organization of arguments. AO-Arb command interface is described in [].

AOS commands are obviously non independent from each other: in figure 4 a state diagram of the legal command flow is shown.

12.1 StartObs

This command is used by the TCS to request the AO System to move from READY service status to OBSERVATION service status, this must be done at least at the beginning of an observation.

No parameters are required.

AOS will send the request message to AO-Supervisor and wait for an acknowledge. AO-Supervisor will perform the required checks and operations (e.g.: it will modify the behavior of the engineering interface at the AO Console) and acknowledge the request. At the end AOS will properly update the value in `DD`.

It should be safe to repeat the command many times during an observation night: if the AOS is already up and communicating when the command is issued, it will be interpreted as a request to

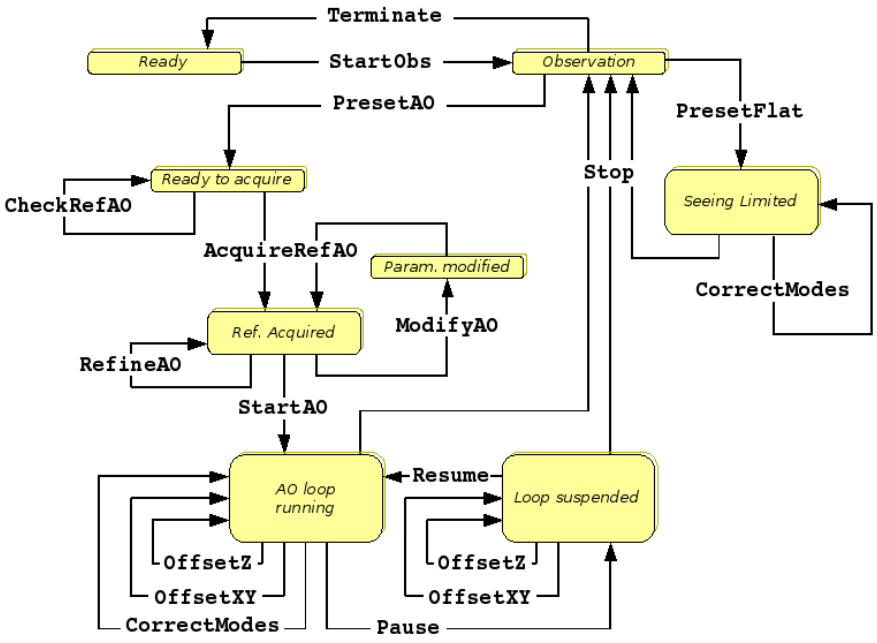

Figure 4: AOS commands state diagram

restore the AO system to its initial status.

12.2 PresetFlat

This command is issued in OBSERVATION service status in order to request the AO System to prepare itself for the FIX-AO[16] mode of observation.

This mode doesn't require a reference star, but it will be possible to select among several different precalibrated "flat" shapes depending on the instrument in use.

`PresetFlat` Command Parameters

Name	Type	Units	Comment
`FlatSpec`	string		Specification of the desired "flat"

After sending the command to AO-Supervisor the AOS will wait for a status change of the corresponding AO variable. It will then reflect the service status into `DD`.

The `PresetFlat` command, when successful, returns a single argument: the name of the selected flat shape:

[16] See sect 1.1.

PresetFlat Return Values

Name	Type	Units	Comment
FlatSpec	string		Specification of the selected "flat"

12.3 PresetAO

This command is issued in OBSERVATION service status in order to prepare the AO System for an observation in adaptive mode, i.e.: one of TTM-AO, ACE-AO, ICE-AO[16].

Following this command the AO System will perform all set up operations needed to prepare for the acquisition of a reference star[17].

When the PresetAO command has been completed and the telescope has reached the pointing position and is tracking and guiding on the target requested by the instrument, a command AcquireRefAO or CheckRefAO may follow.

The AO system must receive all parameters needed to set up the AO System for the requested operating mode, they will be either specified as command arguments or derived from suitable DD variables. A few argument are not mandatory and can be omitted. The following table lists them all.

PresetAO Command Parameters

Name	Type	Units	Source		Comment
AOMode	string		arg	Req.	Either "TTM-AO" or "ACE-AO" or "ICE-AO"
WFS	string		arg	Req.	Specifies the source of WFS data. It is forced to "default" for the AGW hosted WFS.
SlNull	string		arg	Req.	Specifies the selected slope null vector
SOCoords	real[2]	mm	DD	Opt.	Position of the scientific object in focal plane coordinates
ROCoords	real[2]	mm	arg	Req.	Position of the reference object in focal plane coordinates
Elevation	real	radians	DD	Req.	Telescope elevation[a]
RotAngle	real	radians	DD	Req.	Angular position of rotator[a]
GravAngle	real	radians	DD	Req.	Angular position of rotator with respect to gravity[a]
Mag	real	TBD	arg	Opt.	Magnitude of reference star
Color	real	TBD	arg	Opt.	Color Index of reference star
R0	real	TBD	DD	Opt.	Estimated value of R0 (optional)
SkyBrghtn	real	TBD	DD	Opt.	Sky brightness (optional)
WindSp	real	TBD	DD	Opt.	Wind speed (optional)
WindDir	real	TBD	DD	Opt.	Wind direction (optional)

[a] This is the estimated value when the pointing position is reached. This value may be used to select the proper look-up table and to preset the ADC.

[17] The setup operation has been split in two steps (PresetAO followed by a mixture of AcquireRefAO RefineAO and CheckRefAO) in order to allow the AO System to perform potentially time consuming adjustments (such as moving the mechanical assets of the WFS) while the telescope is slewing to point the source.

After sending the command to AO-Supervisor, AOS will wait for a change of the corresponding AO variable. It will then reflect the new value into `DD`.

The AO system is expected to perform all set up operation needed except acquiring the reference object, which will be performed when the subsequent `AcquireRefAO` command will be issued.

At the end of preset the AO System will send an acknowledge to the AOS.

12.4 AcquireRefAO

The `AcquireRefAO` command is issued after a `PresetAO` in order to request the AO System to proceed to reference object acquisition by moving it's internal stages.

Before issuing this command the AOS must check both that the previous `PresetAO` command has been successfully completed and that the telescope has reached the pointing position and the guiding system is operating.

If the above preconditions are fulfilled, the AO System must find the reference star within the field of view of the Technical Viewer and thus is able to adjust the mechanical position of the WFS to put the reference star in the right spot. Then it will compute parameters needed for optimization of the AO loop and set up all the needed optical devices.

During the execution of the command some indication of partial completion of the operation will be sent back to the AOS[18].

The command requires no arguments.

After successful completion, the function will send back the computed AO loop parameters as detailed in the following table.

`AcquireRefAO` Return Values

Name	Type	Units	Comment
ΔX	real	mm	Pyramid X displacement with respect to nominal position
ΔY	real	mm	Pyramid Y displacement with respect to nominal position
`slNull`	string		Selected slope null
`NModes`	int		Number of corrected modes
`Itime`	real	s	CCD integration time
`Nbins`	int[2]		CCD binning (row wise,column wise)
`TTMod`	real	TBD	Tip-Tilt internal mirror modulation
`F1spec`	string		Selected position of filter wheel # 1
`F2spec`	string		Selected position of filter wheel # 2
`Strehl`	real[3]	TBD	Measured Strehl ratio in J,H,K bands
`R0`	real	TBD	Measured R0
`MSNratio`	real[672]	TBD	Measured S/N per mode

[18]Note that the AO setup procedure may require several seconds to be completed.

After receiving the parameter block, the AOS will usually issue either a `StartAO` or a `RefineAO` or a `ModifyAO` command.

Note the ΔX and ΔY return values which are the amount of displacement of the WFS stages needed to put the source on top of the pyramid: in this mode they can be used to evaluate the errors in the pointing model and, possibly, to correct it.

12.5 CheckRefAO

This command can be used to obtain the centering of the reference star by adjusting the pointing of the telescope (instead of the WFS stages). Because it requires more complex interaction with the telescope, it will need a few steps.

After the `PresetAO` command a `CheckRefAO` command is sent. The AO-Supervisor will take an image with the technical viewer, compute the ΔXY value and return it back. Based on that value the TCS can offset the telescope by the right amount and then can send an `AcquireRefAO` command to the AO-Supervisor. After that the WFS will actually perform the acquisition of the reference star.

After successful completion, the command returns the reference star position:

`CheckRefAO` Return Values

Name	Type	Units	Comment
ΔX	real	mm	Pyramid X displacement with respect to nominal position
ΔY	real	mm	Pyramid Y displacement with respect to nominal position

12.6 RefineAO

The `RefineAO` command is used to request the AO system to perform better estimation of the AO loop parameters. It may be issued after the `AcquireRefAO` and will start the AO-Supervisor procedure[19] to optimize the selection of parameters.

The command has a single argument which selects the specific optimization method.

`RefineAO` command parameters

Name	Type	Units	Comment
`Method`	string		Optimization method

This command can be issued only in reply to the successful completion of a previous `AcquireRefAO` command. It is thus assumed that the reference object selected in the previous `PresetAO` command is still correctly positioned.

[19]The AO loop parameters optimization procedure evaluates the performances of the AO for a small interval of parameters values in order to determine the optimal set. It may require several minutes to complete, so it is offered as an optional function.

AO-Supervisor will perform the optimization procedure and reply with the same parameter block described for the `AcquireRefAO` command (see sect. 12.4).

12.7 ModifyAO

The `ModifyAO` command is used to support the ICE-AO operating mode. It may be used to request the AO System to modify the value of some AO loop parameter before closing the AO Loop. The command must specify the set of AO loop parameters selected by the observer as detailed in the following table[20].

`ModifyAO` command parameters

Name	Type	Units	Comment
`NModes`	int		Number of corrected modes
`Itime`	real	s	CCD integration time
`Nbins`	int		CCD binning
`TTMod`	real	TBD	Tip-Tilt internal mirror modulation
`F1spec`	string		Selected position of filter wheel # 1
`F2spec`	string		Selected position of filter wheel # 2

This command can be issued only in reply to the successful completion of a previous `AcquireRefAO` and/or `RefineAO` command. It is thus assumed that the reference object selected in the previous `PresetAO` command is still correctly positioned.

AO-Supervisor will perform a check of the validity of parameters[21], recompute the optimization values, and will reply with the same parameter block described for the `AcquireRefAO` command (see sect. 12.4).

The AOS can in principle issue an arbitrary number of `ModifyAO` commands before requesting the closing of the AO loop with a `StartAO` command.

12.8 StartAO

The `StartAO` command is issued by AOS to request the closing of the AO loop. It doesn't require any parameter in that it is only needed to synchronize the AO operation with the scientific instrument operation and when it is issued the AO System must be fully ready to close the AO loop.

After sending the corresponding command to AO-Supervisor, AOS will wait for a change of the corresponding AO variable.

It will then reflect the value into `DD`. The instrument software will thus be able to know from this status variable when the AO system is ready to start scientific data acquisition.

[20] The capability to specify some 'free" parameters (i.e.: parameters which can be optimized by the AO system, and not forced to specified values) will be provided.

[21] Parameter checking will be particularly strict for this command in order to ensure safe operation of the AO system.

12.9 OffsetXY

This command is issued in OBSERVATION service status in order to offset pointing of the AO System (i.e.: by moving the WFS stages). The command has different effects depending on the current mode. When the AO loop is closed any offset applied to the WFS stages will be corrected by the AO system and this will result in an offset of the field on the scientific camera[22]. When the AO loop is not closed (e.g.: after a `Pause` command) it will simply move the WFS stages.

The command requires two delta position values:

`OffsetXY` command parameters

Name	Type	Units	Comment
`DeltaX`	real	mm	Requested X position offset
`DeltaY`	real	mm	Requested Y position offset

12.10 OffsetZ

This command is issued in OBSERVATION service status in order to offset the focus position of the AO System (i.e.: by moving the WFS stages). If the command is issued when the AO loop is closed, it will be compensated by the AO system, thus it will result in a focus offset in the focal plane of the scientific camera; if the command is issued when the AO loop is open, it will simply move the Z stage of WFS.

The command requires one delta value:

`OffsetZ` command parameters

Name	Type	Units	Comment
`DeltaZ`	real	mm	Requested focus offset

12.11 CorrectModes

This command is used in OBSERVATION service status to apply a modal correction to mirror shape (e.g.: to make active optics corrections in seeing limited mode). A vector of Δ values must be specified.

`CorrectModes` command parameters

Name	Type	Units	Comment
`DeltaM`	real[672]	TBD	Modes correction vector

[22] Offsets in closed loop mode are compensated by AdSec with tip-tilt movements, which will need to be offloaded to telescope pointing correction by the mode offload mechanism (see section 7). Due to round-trip delays this means that only offsets of the order of about one half arcsec can be done in a single step. Bigger offsets must be either done in small steps with proper delay between them, or must be implemented by pausing the AO loop (see section 12.13) sending an OffsetXY and, concurrently, offsetting the telescope pointing and the resuming the AO (see section 12.14.

AOS will send the related request message to AO-Supervisor. No reply is expected to this command. Possible error conditions are notified by AO-Supervisor with the proper message.

12.12 Stop

This command will be issued to stop the current operation. After this command any setting defined by a previous `PresetAO` command will be canceled. The command requires a string parameter containing a description of the reason for stopping (see also sect. 12.16).

`Stop` command parameters

Name	Type	Units	Comment
`Msg`	string		Reason for stopping

AOS will send the request message to AO-Supervisor and wait for acknowledge. Then it will properly update the related variable in `DD`.

12.13 Pause

Temporarily suspend the AO loop; the AO System must remain ready to resume, i.e.: to close again the AO loop. AOS will send the related request message and wait for acknowledge, then it will reflect the status in `DD`. AO-Supervisor will take the proper action (i.e.: open the AO loop) and then acknowledge the request. The command may be followed by either the `Resume` or the `Stop` command.

12.14 Resume

Resume suspended AO loop after a `Pause`. AOS will send the related request message and wait for acknowledge, then it will reflect the status in `DD`. AO-Supervisor will resume the operation and acknowledge the request.

12.15 Terminate

This command is issued at the end of an observation night to terminate operations. After the command the AO-Supervisor will properly put all devices into safe conditions and go back to READY service status. The actual shutdown of AO hardware, and possibly of the AO-Supervisor and AO Computer can then be performed by the operator at the AO-Console.

12.16 UserPanic

This command is issued whenever some TCS subsystem (including an instrument) detects any dangerous conditions and decides to to perform a fast shutdown; the command may be also fired by the observer or operator at some user interface. AOS will send the corresponding request message to AO-Supervisor and Immediately close the connection. AO-Supervisor will will do its best to shutdown

as soon as possible. After the acknowledge no other interaction will be possible between AOS and AO-Supervisor; i.e.: the command cannot be canceled. The command requires a string parameter containing a description of the event which caused it (see also sect. 12.12).

`Panic` command parameters

Name	Type	Units	Comment
`Msg`	string		Reason for panic

13 Commands issued by AO-Supervisor to request services from TCS

Commands sent by AO-Supervisor to AOS in order to request services from the Telescope are subdivided into two subsets: housekeeping commands and hexapod commands.

13.1 Housekeeping commands

The first subset includes housekeeping commands listed in table 14.

Table 14: Commands Issued by AO-Supervisor. I - Housekeeping commands and messages

Command	State	Description
LogItem	ENGINEERING OBSERVATION	Requests to log some piece of information into the TCS Logging System.
LogOn	ENGINEERING OBSERVATION	Activate mirroring to `MsgD` of AOS generated log Messages.
LogOff	ENGINEERING OBSERVATION	Deactivate mirroring to `MsgD` of AOS generated log Messages.
DbgLevel	ENGINEERING OBSERVATION	Set debug level of AOS
Warning	ENGINEERING OBSERVATION	Notify warning condition
Error	ENGINEERING OBSERVATION	Notify error condition
Panic	ENGINEERING OBSERVATION	Notify AO System panic condition

13.1.1 LogItem

Requests to log some piece of information into the TCS Event System.

AO-Supervisor will have it's own logging system to be used for troubleshooting and engineering tasks. A selected subset of the log data will be stored in the TCS log system by means of `LogItem` commands.

13.1.2 LogOn/LogOff

Activate/deactivate the mirroring of AOS generated log messages to the `MsgD`.

After activation all the log messages[23] generated by the AOS other than being logged via the standard TCS logging mechanism, will also be echoed to the `MsgD` in order to be merged with AO-Sup specific logs. This will help in the maintenance of the system by allowing to easily correlate the relevant log messages.

13.1.3 DbgLevel

Activate/deactivate the AOS log messages. This command may be used ad debugging tool to increase/decrease the verbosity of messages generated by AOS. Under normal condition logging should be off (`DbgLevel=0`).

Note that if a LogOn command has also been sent, all logs will be mirrored in the `MsgD` log file.

13.1.4 Warning

This command is used by AO-Supervisor to notify that it has detected some condition which may potentially cause an error (e.g.: some mechanical device is close to a limit position). In ENGINEERING mode AOS will simply update the proper variables, while in OBSERVATION mode it will also notify the operator on the telescope console. Actions to be performed by AOS following a `Warning` command vary, depending on the reason why the warning was generated.

The AOS, when receiving the `Warning` command, will make a notification to the operator GUI.

13.1.5 Error

This command is used by AO-Supervisor to notify that it has detected some error condition (possibly because a previous warning notification has not been properly managed) and has stopped the current operation. In ENGINEERING mode AOS will simply update the corresponding variables, while in OBSERVATION mode it will also notify the operator on the telescope console.

13.1.6 Panic

This command notifies that AO-Supervisor has detected a dangerous error condition and has initiated a shutdown procedure. AOS will stop any current operation and provide that a notification is displayed on the operator's console. After this the AO System will be set in a safe status and it may be necessary that an operator go to the AO engineering console to further investigate the problem and take proper actions.

[23] Except for messages explicitly logged to TCS Syslog by the `LogItem` command.

13.2 Engineering mode commands

The second set of commands, listed in table 15, are used when the AO System is performing engineering operations which do not require pointing and tracking in the sky. In this mode of operation the AO System may need to directly control the hexapod.

Table 15: Commands Issued by AO-Supervisor. II - Low level commands

Command	State	Description
RequestService	ENGINEERING	Request service from TCS
EndOfService	ENGINEERING	Notifies to the rest of TCS that AO System has finished with AO engineering operations, and releases telescope resources.
Stop	ENGINEERING	Terminate current operation
HXPINIT	ENGINEERING	Initialize hexapod
HXPMOVETO	ENGINEERING	Move hexapod to absolute position
HXPMOVEBY	ENGINEERING	Move hexapod relative tom current position
HXPMOVSPH	ENGINEERING	Move hexapod on a sphere
HXPNEWREF	ENGINEERING	Set new reference point
HXPGETPOS	ENGINEERING	Get hexapod current position
HXPGETABS	ENGINEERING	Get hexapod absolute position
HXPOPENBRAKE	ENGINEERING	Open brake
HXPCLOSEBRAKE	ENGINEERING	Close brake
HXPISINITIALIZED	ENGINEERING	Test if hexapod has been initialized
HXPISMOVING	ENGINEERING	Test if hexapod is moving

13.2.1 RequestService

Notifies that the AO System wants to start engineering operation and needs service from some telescope subsystem. AOS will first check that the telescope is in suitable status (e.g.: no instrument has currently authorization to operate the telescope). If the request is accepted AOS will set a proper flag into the telescope status so that other subsystem will not receive authorization to control TCS functions and will request the TCS to activate the related focal station; then it will acknowledge the command to AO-Supervisor. The authorization status variable will be checked by AO-Supervisor prior of issuing any other request to AOS.

13.2.2 EndOfService

Notifies to the rest of TCS that AO System has finished with AO engineering operations, and releases telescope resources.

13.2.3 Stop

Request AOS to stop current operation.

13.2.4 Hexapod Commands

This commands are used in ENGINEERING service status to set hexapod position. AOS will issue the proper commands to the OSS. The current hexapod position and command completion status will be reflected in the corresponding variables in `RTDB`.

A Source Files Identification

Table 16: AOS Source Files

`AOS.cpp` `AOS.hpp`	Implementation of the AOS subsystem framework. It essentially provides for the management of the thread. The actual functionalities are implemented in `AOSAoApp`.
`AOSAoApp.cpp` `AOSAoApp.hpp`	The core of the AOS. Together with code in `AOSHandlers` implements all the functionalities of AOS.
AOSHandlers.cpp	Contains the code for handling messages sent by the AO-Supervisor.
`Main.cpp`	Just launches the subsystem task and cleans up at the end.
`VarUtils.cpp` `VarUtils.hpp`	Contain code for the implementation of variable mirroring back and forth.
`Version.hpp`	Major.Minor version number.

In table 16 the AOS source files are listed together with a brief explanation of their contents and purpose.

B Interface with AO-Supervisor

The code needed to interface AOS with the AO-Supervisor is contained in a directory subtree rooted in `../aos/aosupervisorlib`.

In other words the TCS source code tree contains all source files needed to properly compile and build the AOS executable.

Although modification of the interface is done with the greatest care, in order to avoid incompatibilities in the communication, it may be sometimes necessary to upgrade the interface code to align it with new features added to the AO Supervisor code.

This is essentially a manual operation, because it might require changes in the organization of source files, anyway whenever modifications are limited to file content and no change in file organization has been made, a specific shell command (`update`) is included in the source code tree for updating the files. This anyway requires that the AO Supervisor source tree is available and properly installed.

References

[1] Luca Fini, Lorenzo Busoni, and Alfio Puglisi. Aos functional description. Technical Report 481f340, INAF-Arcetri, May 2009.

[2] Roberto Biasi, Mario Andrighettoni, and Daniele Veronese. LBT672 Design Report: Electronics. Technical Report 641a006, Microgate, May 2008.

[3] Lorenzo Busoni, Simone Esposito, Luca Fini, Alfio Puglisi, Armando Riccardi, and Marco Xompero. Ao supervisor - functional description. Technical Report 486f009, INAF-Arcetri, Mar 2009.

Doc_info_start
Title:
Document Type: Specification
Source: Osservatorio di Arcetri
Issued by: Luca Fini
Date_of_Issue: 16 July 2009
Revised by:
Date_of_Revision:
Checked by:
Date_of_Check:
Accepted by:
Date_of_Acceptance:
Released by:
Date_of_Release:
File Type: PDF
Local Name:
Category: 400
Sub-Category: 480
Assembly: 481 Telescope Control Software
Sub-Assembly:
Part Name:
CAN Designation: 481f341
Revision: a
Doc_info_end