LBT-ADOPT
TECHNICAL REPORT

Doc.No : 687f400
Version : B
Date : 25 Nov 2016

WFS Arbitrator interface

Prepared by A. Puglisi

Approved by

Released by

INAF – Osservatorio Astrofisico di Arcetri
Largo E. Fermi, 5 - 50125 Firenze - ITALY
http://adopt.arcetri.astro.it

ABSTRACT

This document describe the software interface for the WFS arbitrator of the First-light AO units. The interface is identical for right and left units.

Modification Record

Version	Date	Author	Section/Paragraph affected	Reason/Remarks
A	27 01 2009	A. Puglisi		First release of the document
B	25 Nov 2016	A. Puglisi		Updated to reflect UAO changes

Abbreviations, acronyms and symbols

Symbol	Description
LBT	Large Binocular Telesecope
WFS	WaveFront Sensing unit of the LBT AO system

Contents

1. Introduction

The WfsArbitrator is the main control program for the WFS system. It is based on a finite state machine model of the WFS operation, with some additional features for specific AO and engineering operations.

WfsArbitrator is an AOApp which integrates in the AO Supervisor system. It is a server which waits for commands from the higher-level AO Arbitrator, or from other parties (e.g. GUIs or calibration interfaces). Commands are exported using an ICE interface. All commands return correctly or throw an exception in case of errors.

The old arbitrator library interface is still supported but is deprecated. This document refers the ICE interface, and notes corresponding arbitrator library fuction name where applicable.

2. ICE interface definition

The ICE interface exported by the Wfs Arbitrator can be found in these three files (paths relative to the AOSupervisor source tree):

- lib/iceIF/AODefinitions.ice - definitions common to all AO ICE interfaces
- lib/iceIF/PyramidWFS.ice - minimal set of commands to allow control of the WFS as truth sensing
- lib/iceIF/PyramidFLAO.ice - extended set of commands for full WFS control

The PyramidFLAO interface extends the PyramidWFS one, making all the definitions in the latter available to the former.

An ICE client will normally use the PyramidFLAO interface. Systems that only need truth-sensing functionality, like ARGOS, will use the PyramidWFS interface.

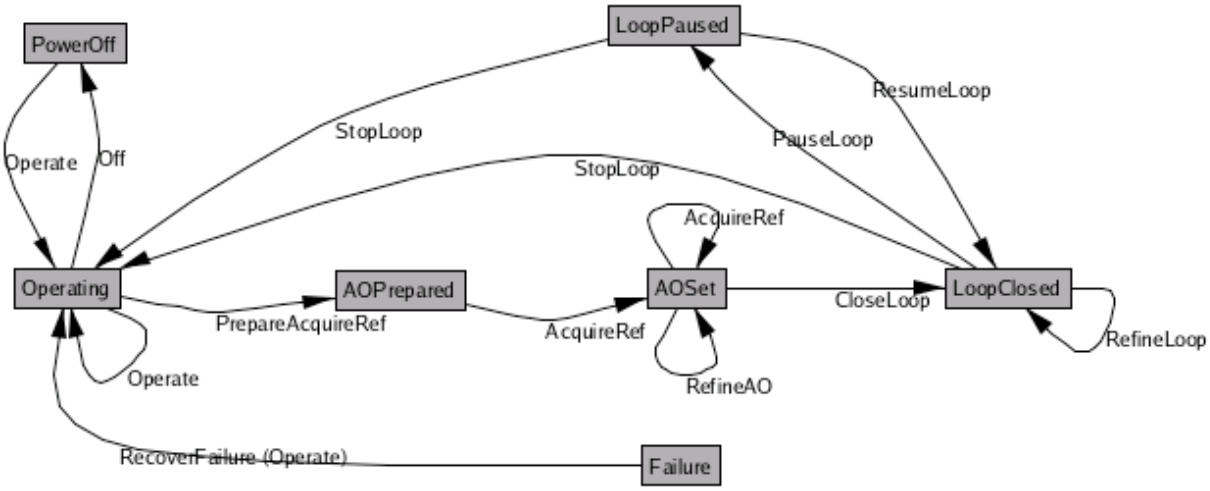
3. Integration in the AO system

The WfsArbitrator manages the WFS part of the first light AO system. As such, its operation is managed by the higher-level AO Arbitrator in order to coordinate with the ASM. For debug purposes, the WfsControl GUI can send the same commands, and incorporates some additional functionality in order to allow an expert operator to control the system without the AO Arbitrator.

The Engineering Interface completely bypasses the WfsArbitrator and talks instead with the hardware. Therefore, when the WfsArbitrator is operating no commands must be sent from the Engineering Interface (which can remain as a read-only status information). The WfsArbitrator provides some special commands to allow limited manual control of devices, but these commands are intended to be used only in engineering mode and not during observation.

4. FSM design

The following is the FSM model of the WfsArbitrator. Most commands are implemented as transitions between states.



In general, there are three completion codes for a state transition:

- SUCCESS: the command was executed successfully
- ERROR: the command cannot be executed because of an internal WFS failure (e.g. hardware failure)
- RETRY: the command cannot be executed now because of an external error (e.g. no light on sensor) and can be tried again if needed.

Commands ending in SUCCESS leave the WFS in the target state. In case of RETRY the WFS is rollbacked to the previous state (therefore no state transition occurs), while ERROR leave the WFS in Failure state and a RecoverError command must be issued.

Each command may add more detailed status information to the completion codes described above.

When using the ICE interface, both RETRY and ERROR will result in an exception being thrown. In case of a RETRY completion, the exception string will contain the “WARNING” tag. In case of ERROR completion, the exception string will contain the “FATAL” tag.

5. Main commands

This chapter contains a detailed description of the most important Wfs commands. This is given as a reference only, the official description is always the ICE interface files.

5.1. PyramidWFS::powerOn

This command put the WFS hardware into an operating state and checks all needed functionality.

Arbitrator library name: wfsarb::Operate()

Valid states: Off, PowerOff, AOPrepared, AOSet, Operating, Stop, RecoverableError

End state: Operating

Return values: Status code.

Time needed: about 2 minutes for full system.

Parameters:

Name	Type	Description
powerOnConfig	String	Requested configuration (from a list provided by the WFS arbitrator). A null value default to full WFS system.
boardSetupFilename	String	Board setup file to load. Empty string means no board setup.
sourceConfig	String	Deprecated, set to empty string

5.2. PyramidWFS::powerOff

This command turns safely off all the WFS hardware. Intended as normal power off procedure.

Arbitrator library name: wfsarb::Off()

Valid states: Operating

End state: Off

Return values: Completion code.

Time needed: 20 seconds.

Parameters: None

5.3. PyramidFLAO::presetAO

This command prepares the WFS to reference star acquisition. All filters are selected, stages are moved to the expected star position and CCDs are started at the expected frame rate. This command can be sent before the telescope has pointed or tracked the scientific object, to minimize the star acquisition overhead.

The result of this code is a description of the WFS parameters that will be used for the star acquisition and closed loop, as they were selected by the software. It is possible to override some of this parameters with the RefineAO command before closing the loop.

Arbitrator library name: wfsarb::PrepareAcquireRef()

Valid states: Operating, AOPrepared, AOSet

End state: AOPrepared

Return values: Completion code plus parameters (see below)

Time needed: up to 2 minutes.

Mandatory Parameters:

Name	Type	Description

AOMode	enum	See AODefinitions::AOMode for list
Instrument	String	Instrument name
roCoordXinMM, roCoordYinMM	float	Expected reference star position in focal plane mm from the telescope FOV center
expectedStarMagnitude	Float	Expected reference star magnitude
badSeeing	Boolean	If true, select bad seeing table

Output parameters:

Name	**Type**	**Description**
pyramidWFSCCD	CameraSettings	CCD configuration (binning, frequency, etc). See AODefinitions for the CameraSettings structure
pupilsTag	string	Selected pupils name
modalBasisTag	string	Selected modal basis name
reconstructorTag	string	Selected reconstructor name
aoLoopTimeFilter	string	Selected time filtering name
tipTiltModulationAmplitude	Float	Internal tip-tilt modulation radius that will be used for the AO loop, in radians.
filterWheel1Pos	float	Selected filter for filter wheel #1
filterWhell2Pos	float	Selected filter for filter wheel #2
telemetryDataDecimation	int	Decimation of telemetry data, 0=no decimation
nModes	int	Number of corrected modes.

5.4. PyramidFLAO::centerStar
PyramidFLAO::checkFlux
PyramidFLAO::centerPupils

These three commands perform the guide star acquisition (previously prepared by a presetAO). It is assumed that the telescope is pointing and tracking the scientific object.

Valid states: AOPrepared, AOSet

End state: AOSet

Return values: Completion code plus parameters (see below)

Time needed: about 1 minute

Parameters: None

centerStar output parameters:

Name	Type	Description
deltaMovementXinMM, deltaMovementYinMM	float	Reference star position in focal plane mm from the telescope FOV center.

5.5. PyramidWFS::modifyAO

.This command can modify the automatic parameters chosen for the loop. Can be used in AOSet mode before closing the loop.

Valid states: AOSet

End state: AOSet

Return values: Completion code

Time needed: usually 10 seconds. Up to 1 minute if the binning is changed.

Parameters:

Name	Type	Description
pyramidWFSCCDFrameRateInHz	Float	Requested frame rate
pyramidWFSCCDBinning	Int	Requested binning.
tipTiltModulationAmplitude	Float	Requested tip-tilt modulation amplitude, in lambda/D.
filterWheel1Pos	float	Requested filter wheel #1 position
filterWheel2Pos	float	Requested filter wheel #2 position

5.6. PyramidWFS::closeLoop

Closes the loop using the current parameters. AdSec is assumed to be in state RunningAO. Time needed is 1 second. This command just enables the fastlink fiber.

Arbitrator library name: wfsarb::CloseLoop()

Valid states: AOSet

End state: LoopClosed

Return values: Completion code

Time needed: 5 seconds.

Parameters: None

5.7. PyramidWFS::offsetXY

Requests an XY correction by moving the WFS stages.

Arbitrator library name: wfsarb::OffsetXY()

Valid states: LoopClosed, Loop Paused

End state: LoopClosed, Loop Paused

Return values: Completion code

Time needed: variable depending on offset length.

Parameters:

Name	Type	Description
offsetXinMM offsetYinMM	float	Requested offset in FoV coordinates (mm).

5.8. PyramidWFS::offsetZ

Requests a Z (focus) correction moving the WFS stages

Arbitrator library name: wfsarb::OffsetZ()

Valid states: LoopClosed, LoopPaused

End state: LoopClosed, LoopPaused

Return values: Completion code

Time needed: variable depending on offset length.

Parameters:

Name	Type	Description
offsetZinMM	float	Requested focus offset in mm.

5.9. PyramidWFS::pauseLoop

Pauses the closed loop, stopping the slope data towards the secondary mirror. Time needed is 1 second. The current illumination level on the pyramid CCD is saved and will be used during the following resumeLoop command.

Arbitrator library name: wfsarb::PauseLoop()

Valid states: LoopClosed

End state: LoopPaused

Return values: Completion code

Time needed: 1 second

Parameters: None

5.10. PyramidWFS::resumeLoop

Resumes a previously paused closed loop, without re-acquiring the reference star. The resume operation can fail if in the meantime tracking has drifted too far (>1 arcsec) and there is no light anymore on the WFS. Time needed is 1 second.

Arbitrator library name: wfsarb::ResumeLoop()

Valid states: LoopPaused

End state: LoopClosed

Return values: Completion code

Time needed: 1 seconds

Parameters: None

5.11. PyramidWFS::stopLoop

Stops a closed loop. The loop cannot be resumed and an additional presetAO command must be used.

Arbitrator library name: wfsarb::StopLoop()

Valid states: LoopClosed, LoopPaused

End state: Operating

Return values: Completion code

Time needed: 1 second

Parameters: None

6. Calibration

6.1. WFS-only calibrations

Some calibrations are specific to the WFS system only. The WfsArbitrator provides commands to perform such calibration. Such commands can be sent by the higher level AO arbitrator or by an operator. All calibration commands are accepted when in Operating state. Results are saved in disk files on the AO workstation as lookuptables and reference frames, that will be automatically referenced during operation.

6.1.1. PyramidWFS::calibrateHODarkFrame

Calibrates dark frames for the Pyramid CCD. The system must be in the dark with no light coming from any source.

Valid states: Operating

End state: Operating

Return values: Completion code

Time needed: variable depending on the number of frames requested.

Parameters: number of frames to average

1.1.1. PyramidWFS::calibrateTechnicalViewerDarkFrame

Calibrates dark frames for the Technical viewer CCD. The system must be in the dark with no light coming from any source.

Valid states: Operating

End state: Operating

Return values: Completion code

Time needed: variable depending on the number of frames requested.

Parameters: number of frames to average

1.1.2. PyramidWFS::calibrateSlopeNull

Calibrates the reference slope vector. Assumes a flat Adaptive Secondary.

Valid states: Operating

End state: Operating

Return values: Completion code

Time needed: variable depending on the number of frames requested.

Parameters: number of frames to average

2. Variables

The WfsArbitrator assumes that a few variables from the DD are made available by the AOS and kept regularly updated. The following is a list of such variables:

Name	Type	Description
AOS.TEL.EL	Float	Current telescope elevation
AOS.<side>.ROTATOR.ANGLE	Float	Current instrument rotator angle

3. References

[1] Luca Fini, Lorenzo Busoni, Alfio Puglisi, “AOS functional description”, CAN number 481f340b

[2] S. Esposito, “FLAO Operating Modes", Presentation at LBT-AO Review Meeting. Firenze 10-11 November 2005.

[3] L. Fini, A. Puglisi, and A. Riccardi, “LBT-adopt control software". In Advanced software, control, and communication systems for astronomy. Edited by L. Hilton and G. Ra_, vol. 5496 of Proc. SPIE, pp. 528-537, 2004.