 

**SPICA-VIS**

**Observing Software**

Authors: Frédéric Morand and Denis Mourard

Doc. No. : SPICA-VIS-0003

Issue : 2.0

Date : 10/07/2020

**CHANGE RECORD**

|  |  |  |  |
| --- | --- | --- | --- |
| **ISSUE** | **DATE** | **SECTION** | **COMMENTS** |
| 1 | 9/03/2020 | All | Creation DM, FM |
| 1.1 | 26/06/2020 | All but 1,2, 3 | SPICA-VIS/OS general requirements and scheduler description FM |
| 1.2 | 07/07/2020 | 5, 6, 8 | Update |
| 1.3 | 10/07/2020 | 7 | Ready for Design Review |
| 2.0 | 10/07/2020 | All | Final corrections for Design Review |

Table of Contents

[1. Scope 4](#_Toc45704871)

[2. General principles 4](#_Toc45704872)

[3. Overall architecture 4](#_Toc45704873)

[4. SPICA-VIS/OS General requirements 5](#_Toc45704874)

[4.1 Functional requirements 5](#_Toc45704875)

[4.2 Scheduler general requirements 5](#_Toc45704876)

[4.3 GUI general requirements 6](#_Toc45704877)

[4.4 Real Time Displays 6](#_Toc45704878)

[4.5 Detectors software requirements 6](#_Toc45704879)

[4.5.1 Science detector (SDT) 7](#_Toc45704880)

[4.5.2 Image/Pupil detector (BCD) 7](#_Toc45704881)

[4.6 Logs – Ancillary data 7](#_Toc45704882)

[5. SPICA-VIS/OS scheduler 7](#_Toc45704883)

[5.1 General organization 7](#_Toc45704884)

[5.2 Sequences description (MF1 and SF functions) 8](#_Toc45704885)

[5.2.1 MF1.1: Initialization and setup 8](#_Toc45704886)

[5.2.2 MF1.2: Record of science data 9](#_Toc45704887)

[5.2.3 MF1.3: Record of calibration data 11](#_Toc45704888)

[5.2.4 SF1: Record of technical data 11](#_Toc45704889)

[5.2.5 SF2: End of night operations 12](#_Toc45704890)

[5.2.6 SF3: Technical issues troubleshooting 12](#_Toc45704891)

[6. SPICA-VIS/OS Graphical User Interface 12](#_Toc45704892)

[6.1 GUI general architecture 13](#_Toc45704893)

[6.2 Real Time Displays (RTD) 13](#_Toc45704894)

[7. SPICA-VIS/OS – CD messages 13](#_Toc45704895)

[8. SPICA-VIS/OS – SPICA servers 14](#_Toc45704896)

[8.1 General requirements 14](#_Toc45704897)

[8.2 Specific requirements 14](#_Toc45704898)

[8.3 Detectors servers 14](#_Toc45704899)

[9. Power and network mapping 14](#_Toc45704900)

[9.1 Network mapping 14](#_Toc45704901)

[9.2 Power control 14](#_Toc45704902)

[10. References 15](#_Toc45704903)

# Scope

The scope of this document is to describe the SPICA-VIS Observing Software (SPICA-VIS/OS) within the CHARA environment.

SPICA-VIS/OS allows the execution of the observations in connection with CHARA Cosmic Debris and SPICA-FT. The observations are prepared in terms of a succession of observing blocks (OB) that could either be: Target, Reference, Check, or Calibration. It is foreseen (depending on resources and evolution of CD) to develop a server of OBs (OB Supervisor) with functionalities like optimization of night strategy, follow-up of execution and completion...

# General principles

SPICA-VIS/OS will use the standard CHARA library and the general encoding principles. It is based on gtk/C and messages architecture.

SPICA-VIS/OS is supposed to run either on the CHARA control network directly connected to the hardware or on the Remote Observing Facility in Atlanta through ssh tunneling. The Remote Observing Facility is accessible via VNC connections and offers tools to open all the required tunnels to the control network.

# Overall architecture

The CHARA principle is that CHARA-Cosmic Debris (CD) is the highest client, surrounded by servers (see Fig. 1). The communication is made on the basis of messages, on which every piece of software register. CD asks the OB Supervisor for the next observation (TBC) or the information is manually set inside CD, then it proceeds to all the required functions, including SPICA-FT. SPICA-VIS/OS obtains the necessary information and status from messages of CD and operates all the SPICA/VIS servers connected to the different hardware pieces. In this document, we will describe SPICA-VIS/OS itself, its links with CD, and its links with the various SPICA/VIS servers, the three modules in gray in Figure 1.

CHARA Cosmic Debris

OB Supervisor

OB database and management

**SPICA/VIS-OS**

Various ICS Servers

Various CHARA servers (also SPICA-FT)

Figure 1: Overall architecture with the OB supervisor, CHARA-CD and its servers, SPICA-VIS/OS and its servers.

# SPICA-VIS/OS General requirements

## 4.1 Functional requirements

SPICA-VIS/OS main functions are:

* MF1 : Scheduling operations for recording data
	+ MF1.1 : Initializing and setting up SPICA/VIS
	+ MF1.2 : scheduling operations for recording science data
	+ MF1.3 : scheduling operations for recording calibration data
* MF2 : Controlling all SPICA/VIS devices thru low-level servers
* MF3 : Communication with CHARA modules (mainly CD, SPICA-FT and STS; the case of LDCs has to be discussed)
* MF4 : Providing a Graphical User Interface (GUI) for observers

SPICA-VIS/OS secondary functions are:

* SF1 : Scheduling operations for recording technical data
* SF2 : End of night operations
* SF3 : (auto)diagnosis in case of a technical issue

To achieve these functions, SPICA-VIS/OS software includes the following main modules: a scheduler, a GUI (including real time displays (RTD)) and communication modules (figure 2). Requirements and description of these modules are given in the next sections.

GUI

(MF4)

Scheduler

(MF1 / SF1)

Figure 2: SPICA-VIS/OS general organization

## 4.2 Scheduler general requirements

SPICA-VIS/OS should provide automated operations as much as possible. For this purpose, the main module of the software is a **scheduler**, dedicated to the control and execution of all operations needed in MF1 and SF functions.

Concerning the scheduler, each main or secondary function is seen as a sequence of several steps it must execute.

Each step is made of elementary commands that can be sent to any SPICA/VIS server or used by the scheduler itself to manage the operations flow.

The execution of a sequence and its corresponding steps must be automated. Nevertheless, observer should be allowed to pause or stop the scheduler at any time. In the same way, a user validation could be requested at the end of some steps to proceed to the next step.

It should be possible to change any sequence or step command without modifying or recompiling SPICA-VIS/OS code.

Detailed descriptions of the scheduler and the sequences it must execute are given in section 5

## 4.3 GUI general requirements

First, the GUI must be compliant with CHARA general encoding principles and then use GTK toolkit.

It must be adaptive: depending of what SPICA-VIS/OS is doing, it must display only relevant text or graphical information.

GUI must provide (to be completed) :

* General informations about the observation
* SPICA-VIS status
* SPICA-FT status
* Scheduler status (about what it is currently doing)
* Record progression (if relevant)
* Last relevant logs entries of the various servers
* Errors and warnings display (including popup window in case of an emergency)
* Some real time displays that are described in the next subsection

## 4.4 Real Time Displays

The GUI must be able to display in real time:

* Image/Pupil detector (BCD) images: for IMG/PUP alignment and for servoing TTT mirrors
* Science Camera (SDT) images: full images or ROI (interferometric or photometric channels)
* Fourier transforms of interferometric channel (15 separated baselines)
* Photometry (extracted from photometric channel)
* More detailed considerations on the final needs will be given during the integration phase

## 4.5 Detectors software requirements

Science detector (SDT) and Image/Pupil detector (BCD) software are based on the same principle.

Detector server grabs each recorded frame and copy it into a shared memory, which is connected to several processes. As seen before, both have a real time display process. This process could be a part (thread) of the detector server or be included in SPICA-VIS/OS (TBD). The other processes should be parts of detector server.

The next subsections describe the processes needed for SDT and BCD.

### 4.5.1 Science detector (SDT)

Two processes must be connected to the shared memory:

* Record frames into FITS files, including all ancillary data needed for filling in FITS header
* Real time display. This process is involved in several tasks:
	+ Full images display or ROI display (e.g. interferometric channel or photometric channel)
	+ Photometry monitoring. It could be done by using bargraph display that can help to detect unexpected variations.
	+ Fourier transforms (FT) of interferometric channel. It must display FT for each of the 15 baselines (for instance in a 6x6 array) and provide OPD, V², SNR estimation.

Communication with OPLE must be possible in order to provide coherencing capabilities.

### 4.5.2 Image/Pupil detector (BCD)

Three processes must be connected to the shared memory:

* Real time display. It must display full images. Different ROI can also be used depending on which information is looking at (pupils or images).

During the alignment procedure, photocenters are computed, either on pupils or star images, to give correction signals applied to IMG/PUP mirrors thru IMG/PUP command matrix.

* Record frames into FITS files (as ancillary data or for diagnosis purposes). This recording is optional during scientific observations but will be mainly used during debugging phase.
* Servo the 6 beams onto the corresponding optical fiber cores.

For this purpose, 6 ROI are defined to extract the 6 star images. Position of the bright speckle is computed (procedure to be defined) and it is servoed to a reference position using a PID controller and TTT command matrix.

BCD software can also put or remove pupil optics in order to switch between pupils and images modes

## 4.6 Logs – Ancillary data

All servers and SPICA-VIS/OS itself must write a formatted log (ASCII file), including a timestamp of each event (received command or action). Errors must be highlighted (e.g. use of red color) to be clearly identified.

The Linux multitail command could be used to display last records of each server log.

All ancillary data needed for science and technical purposes or for checking health of SPICA/VIS must be recorded in specific files if they cannot be found in logs (e.g. TTT real time corrections).

The complete list of ancillary data has not been defined yet.

# SPICA-VIS/OS scheduler

## 5.1 General organization

As presented before, SPICA-VIS/OS scheduler is dedicated to the execution and control of the sequences flow, each sequence corresponding to one specific function (data record, calibration record, technical record, …).

In accordance to section 4.2 requirements, each sequence is divided in several steps that are presented in the next subsection (5.2). Each step can be seen as a macro-command that performs a high-level operation (e.g. alignment, record, …)

To be able to easily change execution flow without code modification, all sequences will be described in terms of steps in a script file (ASCII file). This principle was used for the control software of VEGA.

In the same way, each step (or macro-command) will be described in terms of elementary commands (i.e. one of the SPICA/VIS servers command) in a script file (ASCII file).

Each script line is then an elementary command that can be sent by the scheduler to the corresponding SPICA-VIS server and has the following syntax:

SERVER COMMAND [PARAMETERS]

where SERVER is the name of the relevant server, COMMAND is the command to be executed and PARAMETERS the command parameters if needed.

There are also several commands dedicated to scheduler flow control:

* Skip one or more line in the script
* Unconditional jump to any line in the script
* Conditional jump to any line in the script
* Sleep

Script files for relevant sequences and corresponding steps will be read by the scheduler before any MF or SF functions execution. A command parser will decode all elementary operations and execute them.

## 5.2 Sequences description (MF1 and SF functions)

Here we describe the sequences of operations for MF1 and SF functions. These sequences are based on the templates described in SPICA/VIS Template Manual [SPICA-VIS-0001].

### 5.2.1 MF1.1: Initialization and setup

There are two main sequences of initialization: for starting SPICA/VIS instrument at the beginning of a run and for setting it up at the beginning of each night. Both sequences are described hereafter.

#### 5.2.1.1 At the beginning of a run

* Switch power on for all devices (but spectral and back-illuminating sources)
* Start all SPICA/VIS servers
* Init and/or home each movable devices
* Perform some health tests, especially on detectors (TBD)
* Record technical data (see 5.2.4) :
	+ Birefringence calibration
	+ Reference pixels

#### 5.2.1.2 At the beginning of a night

* check that all power switches are alive and switch on devices if needed
* start all SPICA/VIS servers if they are not running (or restart them if they are stuck)
* init and/or home each movable device if needed (i.e. if power shutdown or server restart)
* check links with CHARA modules (CD, SPICA-FT, STS, OPLE, …)
* check disk space
* record all calibration and technical data needed (see 5.2.3 and 5.2.4) :
	+ Detector calibration
	+ Kappa matrix calibration
	+ IMG/PUP command matrix
	+ TTT command matrix
	+ SPICA-VIS / SPICA-FT co-phasing

### 5.2.2 MF1.2: Record of science data

This function is the standard running function during observations.

For each star (target or calibrator), an observing block (OB) manager will send all information to CD. SPICA-VIS/OS will get all OB data from CD (e.g. star name, pops, PI name, wavelength, …)

It will then setup SPICA/VIS devices, send relevant OB data to SPICA-FT and wait for CHARA to be locked on star.

When light is sent to SPICA/VIS, alignment and flux optimization is performed and FTT control loop is closed.

SPICA/VIS then looks for fringes (interacting with SPICA-FT) and record them.

Figure 3 shows the sequence of operations step by step, as well as interactions with CD (brown) and SPICA-FT (yellow).

A more detailed description of each step can be found in SPICA/VIS Template Manual [SPICA-VIS-0001].

START

Get record infos from CD

Check power and servers

Initialize

Wait for CHARA to be locked on star

Optimize flux injection and close FTT control loop

Align pupils and images

FRINGES record

DARK record

NOFRINGES record

END

Find fringes Start tracking

Figure 3: Typical scheduler sequence for science data recording (MF1)

### 5.2.3 MF1.3: Record of calibration data

#### 5.2.3.1 Spectral calibration

Spectral calibration must be recorded during the night as soon as spectral resolution or central wavelength is about to change.

The main steps for this sequence are:

* Turn spectral calibration sources ON
* Record SPECTRA frames
* Turn spectral calibration sources OFF

#### 5.2.3.2 Kappa matrix

Main steps for kappa matrix sequence are:

* Initialize STS and SPICA/VIS
* Alignment (IMG/PUP) and injection optimization
* Record DARK frames
* For all beam, record NOFRINGES frames (with TTT servo loop ON)

A more detailed description of each calibration record sequence is given in SPICA/VIS Template Manual [SPICA-VIS-0001].

### 5.2.4 SF1: Record of technical data

#### 5.2.4.1 Birefringence calibration

This sequence determines the best position of the PDC (Polarization Delay Compensator)

Main steps for this sequence are:

* Initialize STS and SPICA/VIS
* Alignment (IMG/PUP) and injection optimization
* Record DARK frames
* With beam 1 shutter open, for each beam (2 to 6), record frames for each position of PDC
* Process data and set PDCs to their best positions

#### 5.2.4.2 SPICA/VIS – SPICA-FT co-phasing

This sequence ensures co-phasing between SPICA/VIS and SPICA-FT

Main steps for this sequence are:

* Initialize STS (visible and IR mirrors)
* Initialize SPICA-FT and optimize flux
* Start fringe tracking
* Initialize SPICA/VIS
* Alignment (IMG/PUP) and injection optimization
* Search for fringes on SPICA/VIS with internal delay lines

#### 5.2.4.3 Reference pixels

This sequence defines the reference pixels corresponding to nominal position of the 6 CHARA beams on image/pupil detector.

Main steps for this sequence are:

* Turn the back-illuminating source
* Determine the reference pixels on the BCD and save them

#### 5.2.4.4 IMG/PUP command matrix

This sequence determines the 6 IMG/PUP command matrix (4 x 4 matrix) for images and pupils alignment

Main steps for this sequence are:

* Initialize STS and SPICA/VIS
* For all beams:
	+ Move IMG along the x axis and compute the image and pupil photocenter
	+ Move IMG along the y axis and compute the image and pupil photocenter
	+ Move PUP along the x axis and compute the image and pupil photocenter
	+ Move PUP along the y axis and compute the image and pupil photocenter
* Compute the IMG/PUP command matrix and save them

#### 5.2.4.5 Fast Tip-Tilt (TTT) command matrix

This sequence determines the 6 TTT command matrix (2x2 matrix) for TTT servoing.

Main steps for this sequence are:

* Initialize STS and SPICA/VIS
* For all beams:
	+ Move TTT along the x axis and compute the image and pupil photocenter
	+ Move TTT along the y axis and compute the image and pupil photocenter
* Compute the TTT command matrix and save them

A more detailed description of each technical data record sequence is given in SPICA/VIS Template Manual [SPICA-VIS-0001].

### 5.2.5 SF2: End of night operations

At the end of the observing night, some operations must be done to:

* Securely stop all devices and close servers if necessary
* Save raw data and ancillary data in archive disks
* Start automated reduction pipeline (see SPICA/VIS Data Flow and DRS document [SPICA-VIS-0006]).

### 5.2.6 SF3: Technical issues troubleshooting

In order to simplify issues troubleshooting, we plan to have several tools that can help to diagnose and troubleshoot issues. Most of these tools will be based on parts of science or technical sequences described above.

We are also considering writing autodiagnosis procedures.

These procedures are not defined yet.

# SPICA-VIS/OS Graphical User Interface

GUI requirements are presented in section 4.3 and section 4.4 (specific to RTD)

## 6.1 GUI general architecture

The GUI design has not been started yet.

Our guidelines will be the following ones:

* General and important information always visible. As an example, scheduler current sequence is always visible and currently executed step is clearly identified. In the same way, status of SPICA/VIS (servers, devices) must be displayed and errors must be clearly visible.
* Use of several tabs corresponding to main functions or specific procedures (alignment, record of science data, calibration, …). Only relevant information or RTD are displayed.
* Observer should be able to stop any sequence at any time and take manual control of SPICA/VIS

This section is to be completed, as well as the next one.

## 6.2 Real Time Displays (RTD)

RTD for BCD and SDT should be rather simple displays, able to show images and have elementary control on them (gain, threshold, ROI selection, …).

Other SPICA/VIS RTD could be very similar to those of MIRC-X.

# SPICA-VIS/OS – CD messages

SPICA-VIS/OS will use CHARA messages to share informations with CD during observing sequences.

Existing messages will be used as much as possible, but we will need to create specific messages related to SPICA/VIS (e.g. record duration, DIT, FPS, …).

The comprehensive list of all messages that will be used has not been etablished yet. As an example, a table of a few already identified messages is given below

|  |  |
| --- | --- |
| **CHARA message** | **Direction** |
| ASTROMOD\_SET\_STAR | From CD |
| ASTROMOD\_SET\_ACTIVE\_SCOPES | From CD |
| ASTROMOD\_SET\_REFERENCE\_CART | From CD |
| ASTROMOD\_SET\_OBS\_WAVELENGTH | From CD |
| OBSERVE\_PI\_NAME | From CD |
| OBSERVE\_PROGRAM\_NAME | From CD |
| OBSERVE\_OBJECT\_TYPE | From CD |
| OBSERVE\_STAR\_ACQUIRED | From CD |
| OBSERVE\_FRINGES\_FOUND | From CD |
| OBSERVE\_FRINGES\_DATA\_DONE | To CD |

To be completed

# SPICA-VIS/OS – SPICA servers

## 8.1 General requirements

Each server should be able to register to sockman and CHARA messages

Each server should provide the following commands:

* PING command, to know if the server is alive or not
* STATUS command, to know if the server is idle or busy and what it is doing
* UPDATE command, to get up-to-date information
* INIT (and/or HOME) command, to initialize the server when power has been switched off. For servers that control moving devices, a HOME command should be added or replace the INIT command to put the device in home position.
* CLOSE command, to be able to close the server from high level software

Each server must also write a log (see 4.6)

## 8.2 Specific requirements

Specific commands and functions for each SPICA-VIS server are described in SPICA-VIS Instrument Control Software document [SPICA-VIS-0002]

## 8.3 Detectors servers

Detectors servers need to run several processes, dedicated to the different real time tasks they must achieve. They must save acquired frames in a shared memory that is used by these processes.

Specific requirements for these servers have been described in section 4.5

# Power and network mapping

Here we describe the network of SPICA/VIS, including all the devices (IP and Port) as well as all the remote controlled plugs for power control.

## 9.1 Network mapping

To be defined

## 9.2 Power control

SPICA/VIS power supply is managed thru remote controlled switches.

Main power blocks are:

* Injection table (all devices but detector)
* Spectrograph table (all devices but detector)
* Image/pupil detector
* Science detector
* Retrofeed source
* Spectral calibration source

The number of controlled plugs devices and what they exactly control is to be defined.

# References

SPICA-VIS-0004 : SPICA-VIS General Requirements

SPICA-VIS-0001 : SPICA-VIS Template Manual

SPICA-VIS-0002 : SPICA-VIS Instrument Control Software

SPICA-VIS-0006 : SPICA-VIS Data Flow and Data Reduction Software