European Organisation for Astronomical Research in the Southern Hemisphere

Programme: GEN

Project/WP: Control Software & Engineering

ICS Software Specification

Document Number: ESO-351071

Document Version: 1

Document Type: Standard (STD)

Released On: 2021-08-23

Document Classification: ESO Internal Use [Confidential for Non-ESO Staff]

Prepared by: Kiekebusch, Mario

Validated by: Haucke, Jochen

Approved by: Egner, Sebastian Elias

Name

Authors

Name	Affiliation
CINS Group	ESO

Change Record from previous Version

Affected Section(s)	Changes / Reason / Remarks
All	First Version

Contents

1. INTRODUCTION....7
1.1 Scope....7
1.2 Definitions and Conventions....8
1.2.1 Abbreviations and Acronyms....8
1.2.2 Naming Conventions....10
1.3 Applicable Documents....12
1.3.1 ESO Documents....12
1.4 Reference Documents....13
2. OVERVIEW....15
2.1 Hardware architecture....15
2.1.1 Instrument Network....16
2.1.1.1 VLT Control Network....17
2.1.1.2 ELT Network Infrastructure....17
2.2 Basic Software Stack....18
2.3 Software architecture....19
2.4 INS Subsystems....20
2.5 Configuration Control and Structure of INS Source Tree....21
2.6 INS Users....24
2.7 Environment Variables....25
2.8 INS Standards....25
2.9 Simulation....26
2.10 Instrument Modes....26
2.11 Operational Modes....26
2.12 Test Software....27
3. FUNCTION CONTROL SYSTEM (FCS)....27
3.1 Structure....28
3.2 States....28
3.3 Commands....29
3.4 Parameters....29
3.5 FITS header keywords....30
3.6 Stand-alone mode....30
3.7 Logging....30
3.8 Sensor Plots....31
3.9 Performance requirements....31
3.10 Multiple FCSs....31

3.11 Graphical User Interface....31
3.12 Standards....32
3.13 Common Software....32
3.14 Naming conventions....32
4. DETECTOR CONTROL SOFTWARE (DCS)....33
4.1 States....35
4.2 Commands....35
4.3 Parameters....35
4.4 FITS header keywords....36
4.5 Stand-alone mode....36
4.6 Logging....36
4.7 Safety....37
4.8 Simulation....37
4.9 Performance requirements....37
4.10 Failure Mode Operation....38
4.11 Data transmission over instrument LAN....38
4.12 Data format....38
4.13 Data Display Tool....39
4.14 Disk space availability....39
4.15 Other requirements....39
4.15.1 Shutter control....39
4.16 Graphical User Interface....40
4.17 Standards....40
4.18 Common Software....40
4.19 Modules naming conventions....40
5. OBSERVATION COORDINATION SYSTEM (OCS)....42
5.1 States....44
5.2 Commands....44
5.3 Parameters....44
5.4 Safety....45
5.5 Observation Templates....45
5.6 Calibration Templates....45
5.7 Execution of exposures....46
5.8 Control of exposures....46
5.9 Changes during an exposure....47
5.10 Exposure Types....47
5.11 FITS header....47

5.12 Setup files 47
5.13 Templates 47
5.14 Graphical User Interface 48
5.15 Standards 49
5.16 Common Software 49
5.17 Modules naming conventions 49
6. ADAPTIVE OPTICS CONTROL SYSTEM (AOCS) 50
7. MAINTENANCE SOFTWARE (MS) 51
7.1 Instrument Configuration 51
7.1.1 Privileges 51
7.1.2 Change Instrument Configuration Parameters 51
7.2 Maintenance and Verification procedures 51
8. ALARMS 52
9. INTERFACES 53
9.1 Graphical User Interface 53
9.1.1 General Guidelines 53
9.2 Performance Requirements 54
9.2.1 User Station 54
9.3 Programmatic Interface 54
9.3.1 Interface to Observation Handling Tool 54
9.3.2 Interface to on-line Archive 55
9.3.3 Interface to TCS (only for VLT instruments) 55
9.3.4 Interface to CCS (only for ELT instruments) 55
10. INSTALLATION 55
10.1 Deployment 55
10.2 Start-up / Shut-down 55
11. SYSTEM ATTRIBUTES 56
11.1 Safety 56
11.2 Security 56
11.3 Availability 56
11.4 Maintainability 57
11.5 Adaptability and enhancement potential 57
11.6 Documentation 58
12. DEVELOPMENT FACTORS 58
12.1 Development Process 58
12.2 Design considerations 59
12.3 Test requirements 59

12.4 Hardware Obsolecense 59
12.5 Software Evolution 59
13. COMMISSIONING 60
13.1 Software Version 60
13.2 Device Configuration 60

1. INTRODUCTION

[R-ICSS-4] / / /
The purpose of this document is to refine the requirements for instrument control software defined in [AD3] and to provide an specification that can be used by instrument developers. The control software is an integral part of the Instrument Control System (ICS) whose standard architecture is defined in [RD14]. These requirements are applicable to all new instruments designed for ESO telescopes at Paranal (VLT and VLTI), Cerro Armazones (ELT) and La Silla. The document specifies:

- A standard and modular structure which is applicable to any instrumentation software package.
- The common functionality of the standard software modules.
- The basic principles for user interaction and status display.
- The functional interfaces to the other parts of the ELT or VLT Software.
- Software which is common to all instrumentation software packages.

[R-ICSS-5] / / /
Additional software requirements, either derived from the Technical Specification or explicitly specified, will exist for each different instrument. Based on the common requirements which are given in this document and on the instrument specific requirements, a Software requirements shall be written for every instrument.

1.1 Scope

[R-ICSS-7] / / /
The instrumentation software concerns to all new instruments designed for ESO telescopes at Paranal (VLT and VLTI), Cerro Armazones (ELT) and La Silla.

1.2 Definitions and Conventions

1.2.1 Abbreviations and Acronyms

[R-ICSS-10] / / /

The following abbreviations and acronyms are used in this document:

AD	Applicable Document
ADC	Atmospheric Dispersion Corrector
AO	Adaptive Optics
API	Application Programming Interface
CCF	Camera Control Framework
CCS	Central Control System (E-ELT)
CII	Core Integration Infrastructure
COTS	Commercial Off-The-Shelf (components)
CS	Control System
DB	Database
DC	Distributed Clocks
DCS	Detector Control System
DDT	Data Display Tool
DFS	Data Flow System
DPM	Data Product Manager
DROT	De-Rotator
DRS	Data Reduction Software
DWS	Detector WS
ELT	European Extremely Large Telescope
FB	Function Block (PLC programming)
FCF	Function Control Framework
FCS	Function Control System
FITS	Flexible Image Transport System
FW	Framework
FSM	Finite SM
GUI	Graphical User Interface
HDU	Header Data Unit (FITS)
HMI	Human Machine Interface

HRTC	Hard RTC
HW	Hardware
I/O	Input/Output
ICS	Instrument Control System
INS	Instrumentation Software
ILS	Interlock System
LCS	Local Control System
LCU	Local Control Unit
NI	Network Infrastructure
OCF	Observation Coordination Framework
OCM	Observation Coordination Manager
OCS	Observation Coordination System
OLAS	On-line Archive Subsystem
OP	Operational
PLC	Programmable Logic Controller
PTP	Precision Time Protocol
TRS	Time Reference System
RTC	Real-Time Computer
RTMS	Real-Time MUDPI Stream
SCP	Service Connection Point
SDK	Software Development Kit
SM	State Machine
SRTC	Soft RTC
ST	Structured Text
SW	Software
TAI	International Atomic Time
TBD	To be defined
UU	User Units
WFS	Wavefront sensor
WS	Workstation

1.2.2 Naming Conventions

[R-ICSS-12] / / /

Component	A Component is a major entity representing a logical or physical function or service of the ICS Framework.
Control GUI	GUI used for control purposes.
Data Acquisition	The process of acquiring data captured by the science detectors, TCCD's or other systems, acquiring/producing data.
Device	A device encapsulates the high and low level software to control a Hardware Function. It includes the supervisory part as well as the Device Controller running in a control unit such as a PLC.
Device Controller	A Device Controller is the SW implementing the control of a HW Function, e.g. a Function Block for the control of shutters. Several instances of Device Controllers may run inside a control unit. Its interface is generic so that multiple, similar HW functions can use the same interface. The Device Controller functionality is restricted to the direct access to the HW.
Device Hardware Simulator	This is the SW part of the Local Device that emulates the hardware interfaces and behaviour of the Hardware Function, allowing the testing of the Device Controller functionality.
Device Manager	Is the application that provides high-level coordination of a set of devices.
Device Simulator	Is the software that emulates the Device Controller allowing the testing of the Device without the Local Control System.
Exposure	It encompasses the setup of the instrument, one or more integrations, followed by the readout of at least one detector and the storage of the obtained data frame(s) in a FITS file and/or in memory for display.
Function	See Hardware Function.
Hardware Function	Any actuator or sensor that is part of the instrument and controlled by the ICS. Examples of a HW is a shutter, a filter wheel or a cabinet control unit.
Instrument Configuration	A particular composition of an instrument – made up of individual instrument components and represented in SW by a component description.
Instrument Mode	Refers to a specific setup of the instrument, to use it for a certain purpose.
Integration	It is the time interval for which a detector is collecting data. The integration is a subunit of an exposure and does not imply a readout operation.
Local Control System	A Local Control System is the part of the ICS dealing with the HW control. It is composed of at least one local control unit, e.g. a PLC, and a number of Device Controllers.

Local Control Unit	A Control Unit is a general purpose computing unit aimed for controlling and monitoring remotely or locally standalone HW Functions. A control unit may coordinate a set of HW Adapters interconnected through a local or distributed bus system. A control unit can be programmed using standard programming languages and is designed to be used in industrial environments. A typical control unit for instruments is a PLC.
Local Device	It refers to the group of software components running within the LCS and dealing with the control of one hardware function. It includes the Device Controller, the Device Hardware Simulator and the Device HMI.
Observation Coordination System	The system responsible for observation coordination. It consists of the Observation Coordination Manager, Data Product Manager, System Supervisor, guiding facilities and the scripts implementing the scientific observation sequences.
Sequencer	The Sequencer is the SW application, which handles the execution of a series of operations defined in one Observation Block.
Setup	A Setup defines a set of parameters that is applied to the instrument in order to prepare it for a certain operation. A Setup may be submitted as a set of single parameters in commands, or as a collection of parameters contained in a document (file).
Software Component	See Component.
Status GUI	A GUI used to display the status of the complete Instrument under control, or just parts of it. No control actions are possible from a Status GUI.
Template	An executable script that executes a specific type of observation.
Widget	A Widget is a reusable GUI Component, which can be deployed in panels, e.g. to represent a physical or logical function of a system.

1.3 Applicable Documents

[R-ICSS-14] / / / The following documents form part of this document to the extent specified herein. In the event of conflict between the documents referenced herein and the content of this document, the content of this document shall be considered as superseding.

1.3.1 ESO Documents

AD1 Control Development Standards (excluding sections 5.3 and 5.4, section 6.2 and section 10.2);
ESO-193358

AD2 Software Assurance Requirements for E-ELT;
ESO-224035

AD3 ELT Instrument Control System Common Requirements;
ESO-264642

AD4 E-ELT Programming Language Coding Standards
ESO-254539

AD5 Coding Conventions for IEC61131-3 PLC Development;
ESO-266508

AD6 ELT Control GUI Toolkit Architecture
ESO-377114

AD7 Data Interface Control Document (excluding sections 5,6 and 7);
ESO-044156

AD8 ICD Between ICS and OHS
ESO-375289

AD9 ICS Between ICS and OLAS
ESO-384590

AD10 Common ICD between OCS and DCS
ESO-305615

AD11 ICD between the Instruments and the CCS;
ESO-311982

AD12 E-ELT Instrument Control System Development Process;
ESO-267497

1.4 Reference Documents

[R-ICSS-29] / / / The following documents shown herein, are listed as background references only. They are not to be construed as a binding complement to the present document.

RD1 Guide to Developing Software for the EELT
ESO-288431

RD2 E-ELT Linux Installation Guide
ESO-287339

RD3 GigE Vision Standard
https://www.visiononline.org/vision-standards-details.cfm?type=5

RD4 OPC-UA Standard
https://opcfoundation.org/about/opc-technologies/opc-ua/

RD5 Wavefront Sensor Cameras MUDPI Data Packet Description
ESO-319192

RD6 Telescope Control System – User Manual
ESO-223189

RD7 Control GUI Developers Guidelines
ESO-288608

RD8 VLT-ELT Gateway Software Design Description
ESO-351097

RD9 ICS Framework – Function Control Framework – User Manual
ESO-320177

RD10 ELT Instrument Adaptive Optics Real-Time Computer - Real -Time MUDPI Stream Protocol
ESO-310635

RD11 Nomad
https://www.nomadproject.io/

RD12 Common Interface Control Document between VLT Scientific Instrument and the VLT Observatory
ESO-379345

RD13 ELT ICD between the Network Equipment and its Clients
ESO-320983

RD14 Instrument Control System Standard Architecture
ESO-380034

RD15 AO RTC Standard Architecture
ESO-384404

RD16 ICS Framework – Transfer Document
ESO-363356

RD17 RTC Toolkit Design
ESO-349737

RD18 E-ELT Core Integration Infrastructure Software
ESO-287081

RD19 E-ELT Core Instrument Control System Common Definitions
ESO-288424

RD20 NGCII Requirements
ESO-372233

2. OVERVIEW

[R-ICSS-49] D/ / / Each instrument shall be characterized by an **identifier** (or ID) and a **prefix**.

[R-ICSS-50] / / / The Instrument ID is normally set to the name of the instrument (uppercase). Example: *MICADO*. Instrument names shall not exceed eight characters.

[R-ICSS-51] D/ / / The Instrument prefix shall be a two or three characters string depending whether is Paranal or Cerro Armazones control network. Paranal uses two characters but Cerro Armazones will use three. The prefix is used e.g. to identify easily the nodes belonging to the Instrument LAN, files and processes. Example: *mic* (for MICADO).

[R-ICSS-52] D/ / / The very first step in the design of a new Instrument is to **define its ID and prefix in agreement with ESO.**

2.1 Hardware architecture

[R-ICSS-54] / / / Paranal and Cerro Armazones have distributed control systems consisting of a large number of Local Control Units (LCUs) and a set of Workstations (WSs) connected to each other via the network infrastructure. LCUs are logically grouped in a Local Control System that is doing the interface with the hardware.

[R-ICSS-55] / / /

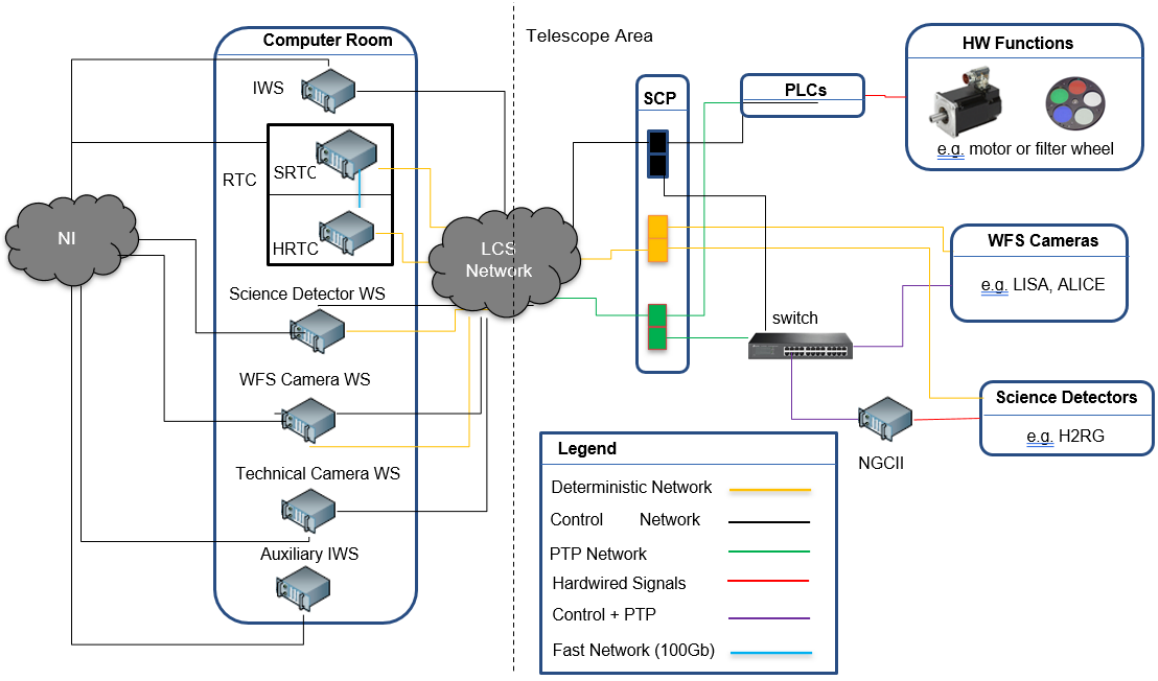

Figure 1: Sample of an Instrument Hardware architecture (ELT).

[R-ICSS-56] D/ / / The instrument hardware (devices and detector camera head and front-end electronics) shall be located in the Telescope Area. Here, network nodes are connected to a SCP which is the interface to the site infrastructure. The Instrument LCS control all devices, except the detectors. The WS connects to the LCS via the OPC-UA communication protocol [RD4].

[R-ICSS-57] D/ / / Science detectors shall be controlled by a dedicated controller (NGCII). A detector WS (DWS) is connected to the controller for data acquisition. The detector data is transferred over the deterministic network using the RTMS packet specification. Control and data information is transferred over the Instrument LAN between the Instrument Workstation and the Detectors WSs.

[R-ICSS-58] D/ / / The Instrument Workstation (IWS) shall be located in the Computer Room. Auxiliary WS may exist depending of the load of the IWS and the needs of the each instrument.

[R-ICSS-59] D/ / / Each IWS shall be statically assigned to an instrument and directly connected to the instrument network infrastructure. The Instrumentation Software on the IWS and LCS usually remain always active, i.e. during day and night time, also when it is currently not used for observations. During this time it monitors the hardware status of the instrument, performs test procedures when requested by operations staff, etc.

[R-ICSS-60] D/ / / Time critical synchronization between WS and LCUs shall be achieved via the Time Reference System. The TRS uses the PTP protocol to distribute the time over a dedicated Ethernet network to all the nodes.

[R-ICSS-61] D/ / / The RTC shall receive the pixel data from the WFS cameras through the deterministic network (ELT) or from a private network (VLT). The RTC has a fast network connection between the SRTC and HRTC.

[R-ICSS-62] D/ / / Technical cameras shall be connected to the control network and transfer the pixel data using GigE Vision protocol.

[R-ICSS-63] D/ / / A number of screens of the User Station in the control room shall be dedicated to the instrument control.

- VLT: Normally the two screens of the User Station console are used: one for control and status display and the other one for real-time image display.

- ELT: User Station configuration for the ELT is TBD.

2.1.1 Instrument Network

[R-ICSS-70] D/ / /
- The IWS node name shall begin with ***w<prefix>.*** If the first two (VLT) or three (ELT) letters of the Instrument ID correspond to the prefix, the IWS node name it is set to w<ID> (lowercase).

Example: *wmicado* (MICADO IWS).

[R-ICSS-71] D/ / /
- Additional WSs might be defined for specific subsystems. The node name for these WS shall begin with **w<prefix><subsystem>**

Examples: *wmicfcs (MICADO FCS)*.

[R-ICSS-72] D/ / /
- **The Instrument LCSs node name shall be *p<prefix>fcs<index>*.** The index always starts from 1. The character 'p' indicates that the node is a PLC.

Examples: *pmicfcs1, pmicfcs2* (MICADO PLCs 1 and 2).

[R-ICSS-73] D/ / /
- **For scientific detectors, WFS and technical cameras, the WS node name shall begin with *w<prefix>*.**

Examples: *wmicdcs1, wmicdcs2* (MICADO detector WSs).

[R-ICSS-74] D/ / /
- For all the other LCUs or network devices, the node name shall begin with *o<prefix>*

[R-ICSS-69] D/ / / The naming conventions for the Instrument LAN nodes are defines by the rules of each control system (VLT [RD12] and ELT [RD13]). In particular, all node names shall be maximum 8 (VLT) or 10 (ELT) characters long.

2.1.1.1 VLT Control Network

[R-ICSS-65] D/ / / Every instrument shall have its own dedicated LAN so that the full bandwidth of the LAN is available for the instrument. The LAN traffic from other instruments and systems might be filtered by a switch.

[R-ICSS-67] D/ / / The Instrument LCS nodes and the Technical Cameras, if any, shall have a normal Ethernet connection to the Instrument LAN. In case of the need of higher time resolution, LCS nodes might be connected to the PTP network.

[R-ICSS-66] / / / For test and maintenance work close to the instrument, it is possible to connect X-terminals or a laptop directly to the instrument LAN. This could be used for LCS troubleshooting or maintenance tasks.

2.1.1.2 ELT Network Infrastructure

[R-ICSS-68] / / / CCS and instruments are interconnected through the ELT Network Infrastructure (NI). The NI defines three main networks according to the type of traffic: ELT Control Non-Deterministic (CND), ELT Control Deterministic (CD) and ELT Time Reference System (TRS).

[R-ICSS-425] D/ / / Instruments shall have its own network infrastructure. This project specific network infrastructure is denominated as Local Communication Infrastructure (LCI). The instrument LCI shal implement a project specific CND and CD networks to satisfy the instrument needs according to the guildelines provided by ESO.

[R-ICSS-428] D/ / / Instruments shall have its own network infrastructure. This project specific network infrastructure is denominated as Local Communication Infrastructure (LCI). The instrument LCI shal implement a project specific CND and CD networks to satisfy the instrument needs according to the guildelines provided by ESO.

[R-ICSS-427] D/ / / All traffic between the instrument shall be in the scope of the instrument LCI.

2.2 Basic Software Stack

[R-ICSS-76] D/ / /
The INS shall be based on a number of ESO products and frameworks. These products represent the basic software stack for the INS. At the bottom of the stack is the Development Environment (DevEnv) that includes CII (MAL and services). For new VLT instruments, the ELT/VLT Software Gateway shall be used by the INS to interface VLT existing applications like the TCS. The Software Gateway is translating the communication between the two environments. For more information about the Software Gateway see [RD8]

[R-ICSS-77] D/ / /
For scientific detectors and standard WFS cameras, NGCII provides the software infrastructure to control the detector and acquire the pixel data. INS shall be based on the IFW which provides the basic architecture and defines a set of standard components for the control software. At the top of the stack is the RTC toolkit which is a framework for the construction of Software Real-Time Computing subsystems of an Adaptive Optics (AO) Real-Time Computer (RTC).

[R-ICSS-78] / / /

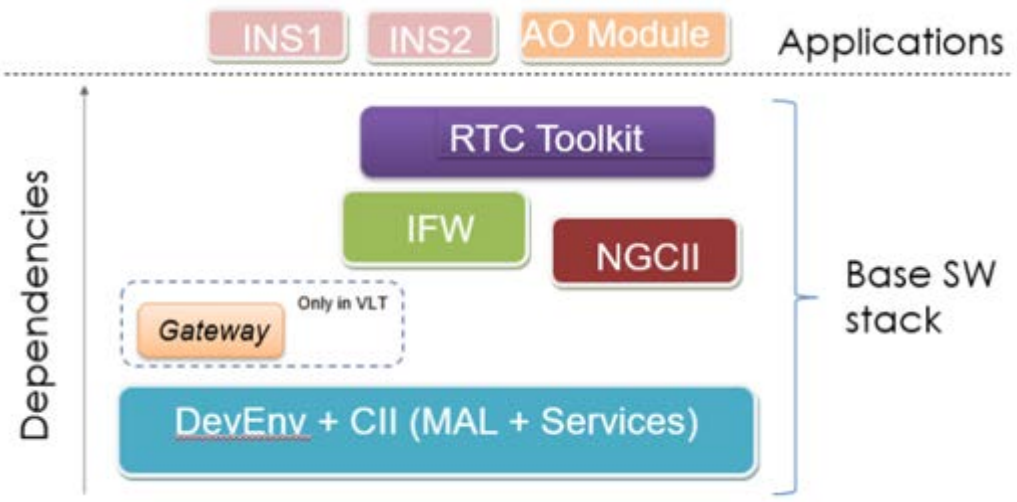

Figure 2: Basic Software Stack.

2.3 Software architecture

[R-ICSS-80] / / / Figure 3 shows the standard architecture of an Instrumentation Software and the data flow between its components.

[R-ICSS-81] / / /

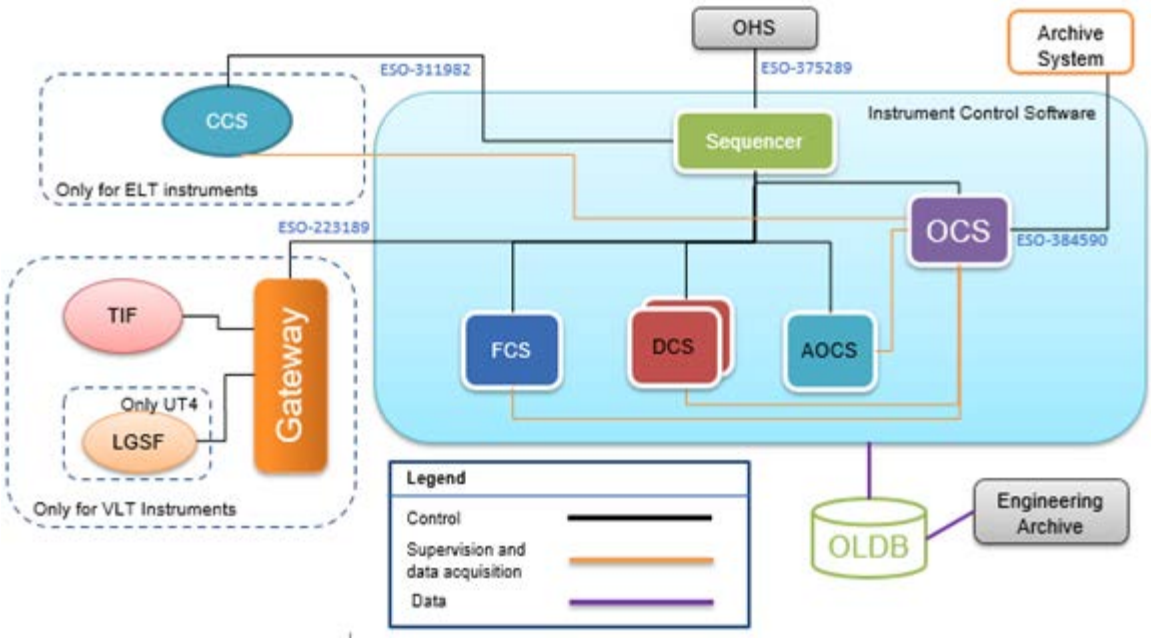

Figure 3: Control Software architecture.

[R-ICSS-82] D/ / / Observation Blocks (OBs) shall be prepared by the observing team well before the observing night using the Phase 2 preparation software.

[R-ICSS-83] / / / During the observing run, the next OB to be executed is loaded again in the Unified Observation Tool on the Observation Handling WS. It is then ready to be fetched by the Sequencer. The Sequencer reads the contents of the OB and executes one by one the templates specified in there. Each template consists in general of a sequence of commands to be sent to the various INS subsystems like the OCS, FCS, DCSs, AOCS and CCS. In case of the VLT, the Sequencer will send the commands to the TCS through the ELT/VLT Gateway. The corresponding subsystems will take care of instructing the respective LCS of all actions they should carry out on the connected hardware.

[R-ICSS-84] / / / The typical activities involved in an observations are:

- Configure the systems for the upcoming observation. The Sequencer sends one or more commands to the subsystems. The subsystems will configure the hardware according to the settings that have been selected by the user, e.g. a particular filter wheel or detector readout.
- Prepare for the data acquisition. The OCS specifies upfront how the final data products will be created.
- Wait for the exposure to be finished.

[R-ICSS-85] D/ / / As a result of an exposure, DCS shall generate detector data and save it in a FITS file. The Observation Coordination System (OCS) that is responsible for coordinating the data acquisition, shall takes care of merging into that file the information, coming from the other sub-systems, related to the same exposure. The OCS then shall transfer the file to the On-Line Archive Subsystem (OLAS).

2.4 INS Subsystems

[R-ICSS-87] D/ / / An instrumentation software package shall be subdivided into the following **standard INS subsystems**:

- **Observation Coordination System** (OCS)
- **Function Control System** (FCS)
- **Detector Control System** (DCS)
- **Adaptive Optics Control System** (AOCS)
- **Maintenance Software** (MS)

[R-ICSS-88] D/ / / The **Observation Coordination System (OCS)** shall be used to coordinate the execution of an exposure for a given observing mode.

[R-ICSS-89] / / / It supervises the different instrument subsystem and takes care of the data acquisition. It interfaces to other applications like the Online Archive system. It also completes the final FITS header for the data product.

[R-ICSS-90] D/ / / The OCS shall not access hardware functions of the instrument. It has the "knowledge" of how to coordinate the control systems to perform exposures for given observing modes.

[R-ICSS-91] D/ / / The OCS shall deal with single exposures. Sequences of exposures are instead executed by templates.

[R-ICSS-92] / / / Acquisition, calibration and science templates (signature files and scripts), needed to build the Instrument Package, required by Unified Observation Tool, are also considered part of OCS.

[R-ICSS-93] D/ / / The **Function Control System** shall control and monitor instrument hardware functions and sensing systems. It has a LCS that interfaces with the hardware and a supervisory part coordinating the LCS and doing the interface with high-level software.

[R-ICSS-94] D/ / / The **Detector Control System (DCS)** shall carry out all tasks to control the detector sub-system, to perform real-time image processing, if and when needed, and to transfer detector data to the workstation.

[R-ICSS-95] / / / ESO provides software for the standard scientific infrared and optical detector systems. This software is part of the NGCII product (RD20), NGCII software also provides the core software for the control of the ESO standard WFS cameras.

[R-ICSS-96] / / / For the control of technical cameras, ESO provides the Camera Control Framework (CCF).

[R-ICSS-97] D/ / / The **Adaptive Optics Control System (AOCS)** shall control a Real Time Computer (RTC) and auxiliary systems to provide adaptive optics corrections.

[R-ICSS-98] D/ / / The **Maintenance Software (MS)** shall be used for instrument configuration, check-out and troubleshooting. It shall also provide technical templates, e.g to verify the instrument calibration, which involves the instrument and detector WSs.

[R-ICSS-99] D/ / / The INS package shall also contain **facilities to build, install and startup/shutdown the Instrumentation Software**.

[R-ICSS-100] D/ / / The data reduction is performed by the Pipeline Software on a dedicated WS as part of the DHS. The description of this Software is outside the scope of this document. It is important to stress that the results of the pipeline data processing are not available for the INS. Whenever the Instrumentation Software (INS) on the IWS needs to know the results of data processing, e.g. to decide what to do next, and cannot accept that this decision is taken by the operator after checking the pipeline results on a dedicated separate screen, then that kind of data processing shall be performed by the INS. If there are no real-time requirements, the **data processing required by INS shall be implemented within the templates**.

[R-ICSS-101] D/ / / **If and which data processing shall be implemented within the INS** has to be discussed at the Instruments Design Reviews and **must be subject to prior ESO approval**.

[R-ICSS-102] D/ / / **The usage of a tool different from the current standard is also subject to prior ESO approval**.

[R-ICSS-103] D/ / / Users shall interact with the INS via Graphical User Interfaces (GUI). All GUIs shall have a common "look and feel". Specific panels shall be developed for every instrument**, following the guidelines for developing GUIs [RD7]**

[R-ICSS-104] D/ / / Last but not least, **the INS packages shall include also test Software** (unit and integration tests) for each of the INS modules. The minimum set of requirements for testing are defined in [AD3].

2.5 Configuration Control and Structure of INS Source Tree

[R-ICSS-106] D/ / / It shall be possible to rebuild the Instrumentation Software from scratch. In order to achieve this purpose, all files belonging to the INS package shall be managed using Git as the standard tool for configuration management.

[R-ICSS-107] D/ / / **The usage of Git and Gitlab is mandatory** and requires that files are located in one instrument group with many Git projects (repositories). The usage of different repositories withing the instrument group is needed to provide certain flexibility for different parts of the control software. It also enables custom configurations for specifics repositories

storing binary files. In general every INS consists of several software modules, online documentation, configuration files and tests.

[R-ICSS-108] D/ / /

***The basic* structure shall be according to the following scheme**:

```
<instrument>
├── <prefix>-ics
│   ├── dcs
│   │   ├── sci[N]
│   │   ├── tec[N]
│   │   └── wfs[N]
│   ├── ddt
│   │   └── <custom viewer>
│   ├── fcs
│   │   ├── hcs
│   │   └── lcs
│   ├── ocs
│   │   ├── <auxiliary_process>[N]
│   │   └── gui
│   ├── odp
│   │   └── <custom image processing>
│   ├── ms
│   │   └── <maintenance tools>
│   ├── seq
│   │   ├── sci
│   │   └── tec
│   ├── doc
│   │   └── <subsys>[N]
│   ├── test
│   │   ├── dcs
│   │   ├── fcs
│   │   ├── ocs
│   │   ├── seq
│   │   └── odp
│   └── wscript
│
├── <prefix>-aocs
│   ├── ddt
│   │   └── <custom viewer>
│   ├── fcs
│   │   ├── hcs
│   │   ├── lcs
│   ├── hrtc
│   │   └── <custom viewer>
│   ├── ocs
│   │   └── <auxiliary process>
│   ├── srtc
│   │   └── <custom viewer>
│   ├── doc
│   │   └── <subsys>[N]
│   ├── test
│   │   ├── fcs
│   │   ├── ocs
│   │   ├── ..
│   │   └── srtc
│   └── wscript
│
```

```
└── <prefix>-resource
    ├── armazones
    │   ├── config
    │   ├── lcs
    │   ├── ..
    │   └── dictionary
    └── <other location>
```

Table 1: Basic INS Tree structure.

[R-ICSS-109] / / /

The standard INS structure forsees at least three main Git projects. This structure shall be flexible to accomodate to the different instrument scenarious. Instrument may add additional repositories to store the source code of other components.

The INS structure includes a **resource** repository where to store the configuration data of an instrument, a repository storing the source code of the ICS and a repository storing the AOCS source code. Depending of the situation, the ICS repository might have an internal AOCS directory as well.

[R-ICSS-110] / / /

Item	Description
<instrument>	The "<instrument>" is the Gitlab group, which will host the different repositories of the instrument software: resources, ICS and AOICS and possible others.
<instrument>/<prefix>-resource/	The instrument resource repository shall follow the definition in [RD1]. The resource tree shall contain the complete configuration of the instrument and can be checked out and used directly, so that no installation of configuration data is needed. Configuration data including binaries and FITS files shall be stored in this repository.
<instrument>/<prefix>-ics	All source code belonging to the ICS software, generated and edited manually, shall be kept in the <prefix>-ics repository. The repository shall be a valid waf project allowing to build and install the ICS software. Under this directory, the source code is grouped by subsystems and includes integration tests and documentation.
<instrument>/<prefix>-ics /fcs	Hosts the various FCS related subsystems. It is split in the PLC code and WS code.
<instrument>/<prefix>-ics /dcs/	Hosts the source code for the Science, WFS and Technical Detector Systems.
<instrument>/<prefix>-ics /ocs/	The source code for the OCS.
<instrument>/<prefix>-ics /odp/	The source code for the online data processing.
<instrument>/<prefix>-ics /seq/	The source code for the Sequencer Templates.

	This is split into a Science and a Technical Templates part.
<instrument>/<prefix>-ics /ddt/	The source code for customised DDT GUIs, based on the DDT Toolkit.
<instrument>/<prefix>-ics /ms/	The source code for INS maintenance software. Technical templates are located under seq directory.
<instrument>/<prefix>-ics /test	Contains the integration tests for the ICS software. Normally there will be one module per subsystem, located in the "<instrument>/<prefix>-ics" branch. I.e., "<instrument>/<prefix>-ics/fcs" → "<instrument>/<prefix>-ics/test/fcs".
<instrument>/<prefix>-ics/doc/	The user manual documentation for the instrument software.
<instrument>/<prefix>-aoics	All source code belonging to the AOCS software, generated and edited manually, shall be kept in the <prefix>-aocs repository. The repository shall be a valid waf project allowing to build and install the AOCS software. Under this directory, the source code is grouped by subsystems and includes integration tests and documentation. The internal structure is similar to the one of the ICS.

Table 2: Main directories in the INS Tree.

[R-ICSS-111] / /I/ **All rules defined in [AD4] apply also to the INS packages**.

[R-ICSS-413] / /I/ Configurations for standalone controllers like PI motion controllers, specific network switches or any other hardware device shall be kept under configuration control following the same rules defined in [AD3].

2.6 INS Users

[R-ICSS-113] / / / **Every instrument defines two user names**, which shall be known on all nodes in Instrument:

[R-ICSS-114] / / / 1. The **INS manager**, responsible for building and installing the Software. The name shall be <*prefix*>*mgr* (all lowercase).

Example: *micmgr*

[R-ICSS-115] / / / 2. The **INS runtime user**, responsible for starting/stopping and running the INS environments and the INS Software. The name shall be <*ID*> (all lowercase).

Example: *micado*

[R-ICSS-116] / / / It is recommended to get used to the repartition of responsibilities between the two users already from the beginning of the development.

2.7 Environment Variables

[R-ICSS-407] / /I/ The environment variable *CFGPATH* shall be defined for the runtime user. This contains a colon separated list of paths of Resource Tree directories, in which various configuration information to execute the instrument software is stored.

In particular, the relevant *<instrument>/<prefix>-resource/<location>* directories in the INS Tree shall be contained in the *CFGPATH* environment variable.

Configuraiton information will be searched for in the Resource Tree paths in the order they are defined in the *CFGPATH* variable.

Note, when referring to files in the Resource Tree, the type of resource file shall be contained in the name (path), e.g. "config/metis/fcs1/motor1.yml'. I.e., the file in question here is a configuration file.

[R-ICSS-408] / /I/ The environment variable *DATAROOT* shall point to a directory where the data produced during runtime, in particular output image data, will be stored.

No specific structure is defined for this directory, but it is possible to allocate subdirectories under the *DATAROOT* directory, to organise the data.

The *DATAROOT* directory shall be writable for the runtime user.

Normally the *DATAROOT* directory shall be hosted on a 'safe drive', e.g. deploying a RAID configuration and shall have enough storage capacity to store the expected volume produced within the given, maximum time to keep the data produced in the storage area.

DATAROOT shall not point into any software source tree area, nor any directory where the sources of the instrument software and generated binaries, are installed.

Moreover, in a production environment, *DATAROOT* shall not point to a directory located under any user account.

2.8 INS Standards

[R-ICSS-118] D/ / / The INS package shall be based on the standard packages as shown in the base software stack shown in Figure 2, in particular:

- Technical Camera control shall be based on the **CCF component**.
- Infrared and Optical scientific DCS shall be based on the **NGCII Software**.
- WFS DCS shall be based on the **software provided by ESO for the standard WFS cameras**.
- The display of detector data shall be based on the ***DDT* component**.

- FCS shall be based on the ***FCF* component**.
- OCS shall be based on the ***OCF* component.**
- The online image processing shall be based on the ***ODP* component**.
- The handling of FITS keywords shall be based on the ***DIT* library**.
- The implementation of auxiliary WS processes shall be based on RAD.
- The implementation of integration tests shall be based on ETR.

2.9 Simulation

[R-ICSS-120] D/ / /
Instrument software shall meet the requirements for simulation specified in [AD3].

[R-ICSS-121] / / /T
It shall be possible to run the Instrument software in simulation, i.e. with no HW attached. The simulation shall permit at least to run/test existing and new templates.

2.10 Instrument Modes

[R-ICSS-123] D/ / /
1. Instrument modes shall be described in the instrument documentation.
2. Instrument modes shall have distinct configurations (setups).

2.11 Operational Modes

[R-ICSS-125] D/ / /
The instrument shall define some basic operational modes to facilitate daily operations: DAY, NIGHT and CALIB. The expected configuration for each operational mode is as follows:

1. DAY:
    - Stable setup in which the instrument can be left for a long time, i.e. day time.
    - CCS/TCS access disabled.
    - Should facilitate access for Instrumentation personnel.
    - As a minimum the global state of the Instrument should be NotOperational/Ready.
2. NIGHT:
    - Instrument ready for night operations.
    - CCS/TCS access should be disabled.

- Global state Operational.

2. CALIB

- Instrument ready for day time calibrations.
- Instrument global state Operational.

2.12 Test Software

[R-ICSS-127] / /I/ Software components shall implement the Software needed to test its proper behaviour. This includes unit tests for each of the component modules and a set of dedicated integration tests according to the testing requirements defined in [AD3].

3. FUNCTION CONTROL SYSTEM (FCS)

[R-ICSS-129] D/ / / **The Function Control System (FCS) shall control all devices belonging to an instrument, except the detectors**. Examples for instrument devices are: slits, gratings, mirrors, filter wheels, temperature sensors, pressure sensors, calibration lamps, etc.

[R-ICSS-130] / / / FCS in general consists of one part, which runs on LCS and one part, which runs on the a WS that could be the IWS or a dedicated WS.

[R-ICSS-131] D/ / / The LCS part shall be responsible for the interface to the hardware functions and in general the low-level control, including real-time functionality, if any. It uses a set LCUs such as Beckhoff PLCs.

[R-ICSS-132] D/ / / The WS part shall be responsible for the coordination between LCUs, if more than one, their WS simulation, if not available (e.g. in a development or test environment) and for the API to the Sequencer and OCS.

[R-ICSS-133]
/ / /

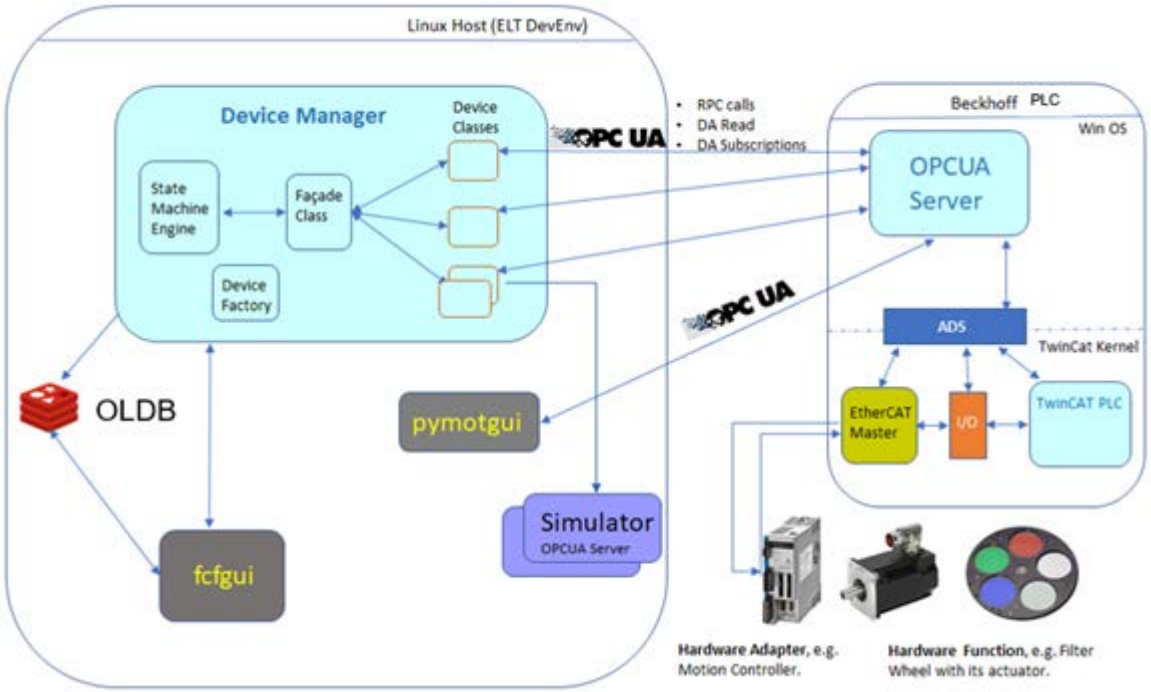

Figure 4: FCS Architecture.

3.1 Structure

[R-ICSS-135]
/ /I/

The internal directory structure of an FCS shall follow the following structure:

```
<FCS Name>
    ├── lcs:   : PLC TwinCAT projects for special controllers.
    ├── hcs:
    │   ├── gui           :   FCS standalone GUI
    │   ├── wdglib          :   Library of special device widgets
    │   ├── devices         :   library of special devices (WS)
    │   ├── devsim          :   WS simulators for special devices
    │   ├── server          :   Custom FCS devmgr
    │   ├── fcsclib         :   Custom python client
    │   └── wscript
```

3.2 States

[R-ICSS-137]
/ / /

All the **standard FCS states**, and the commands to change state, are specified in [RD9].

3.3 Commands

[R-ICSS-139] / / /
All the **standard FCS commands**, the command syntax and conventions are specified in [RD9].

3.4 Parameters

[R-ICSS-141] / / /
The FCS software shall maintain all its parameters in the OLDB.

[R-ICSS-142] / / /
FCS parameters can be grouped into the following categories:

- Configuration parameters (see also section 7.1)
- Setup parameters (set values, see point 3 below)
- Status parameters (actual values, see point 3 below)

[R-ICSS-143] D/ /I/
The following rules are mandatory:

1. **All parameters values** (e.g. wavelength range, operating modes, wanted device position, current device position etc.) shall be stored into the database. They **shall never be *hard-coded* in the Software**. The OLDB shall be populated from the instrument configuration.

2. **The status of the LCS shall be stored into the LCU local memory** in order to enable HMI to monitor the status of the LCS. This implies that the local LCU memory is updated when the hardware status is read or the software status changes.

3. **Parameters set by a command**, like position for a motor, **shall not be modified by the LCS software until a new command to change the value of this parameter is received**. This is called the **set value** of the parameter. Parallel to the set value there shall be a value for the real status which is called the **actual value** of the parameter.

4. **Set and actual value of a parameter shall be stored in separate attributes in the local memory.**

5. For actions like moving a motor, setting a temperature etc**., there shall be a status parameter indicating that the action is going on or was completed or failed**.

6. **Set values shall be checked for validity at the WS level** (e.g. known filter name or slit width in the allowed range).

7. Many parameters from LCS local memory (status, configuration, etc.) must be replicated to the **on-line database** on the workstation, so that they are directly accessible by other programs on the workstation. For example, the User Interface reads from the on-line database status information, which is to be displayed.

3.5 FITS header keywords

[R-ICSS-145] / /I/T **1. The FCS FITS header part shall contain** :

a. The actual status of the FCS, which can be obtained with the standard command Status.

b. It shall be ensured that the FITS header data contain the updated status.

[R-ICSS-146] / /I/T 2. The ICS shall be able to **produce FITS header data in simulation** (test) **mode**, i.e. when the LCS is not available or when the hardware of some devices connected to the LCU is simulated or ignored. Hardware that is ignored or simulated must be clearly stated in the FITS header.

[R-ICSS-147] D/ /I/ 3. Keywords appearing in the FITS header shall comply with the rules described in AD7 and shall be checked against the associated dictionary for syntax validity.

3.6 Stand-alone mode

[R-ICSS-149] / / / The stand-alone mode is used:

1. To enable Operations staff to test and access the individual devices of the instrument without starting the other modules of the INS package. This requires a **dedicated GUI** panel with a detailed status display.

2. To test (debug) the LCS Software without necessarily involving other modules of the INS package.

3.7 Logging

[R-ICSS-151] / / / The mechanism for logging in WS shall be according to the CII Logging Service. Logging in LCS shall be routed to the CII Logging service. FCS shall meet the requirements for logging specified in [AD3].

[R-ICSS-152] / / / Note however that, especially for logs produced by PLCs, too many logs risk to overload the logging system. As a rule of thumb, it shall be avoided to generate periodical logs (e.g. change of temperature) at frequencies higher than 0.1 Hz. If samples collected at higher frequency need to be logged, they should be better grouped into one single log (or, if acceptable, only their average value is logged).

3.8 Sensor Plots

[R-ICSS-154] / / / Temperature and vacuum sensors plots shall be a deliverable for PAE.

3.9 Performance requirements

[R-ICSS-156] / / / General performance requirements are outlined in [AD3], in particular:

1. The execution times for commands, acknowledgements and replies.
2. The need for setting-up instrument devices automatically and in parallel whenever possible.
3. The maximum accepted delay for updating the status display.
4. Time-out values.
5. Times for start-up and shut-down.

3.10 Multiple FCSs

[R-ICSS-158] / / / Depending of the complexity, some instruments may have more than one FCS. The control of multiples FCS is done from the Sequencer. No synchronization is expected across different FCSs.

3.11 Graphical User Interface

[R-ICSS-160] D/ / / FCS shall provide a Graphical User Interface (GUI) for engineering purposes to support the stand-alone mode.

[R-ICSS-161] / / / A tool to help building such a GUI is delivered with the IFW and is contained in FCF.

[R-ICSS-162] D/ /I/ Any other auxiliary GUI, which may be needed, shall be implemented with the ESO UI Toolkit [AD6] and following the guidelines for developing control GUIs (see [RD7]). FCS GUIs shall meet the requirements for graphical interfaces specified in [AD3].

3.12 Standards

[R-ICSS-164] D/ / /
ESO defines a set of standard devices, using ESO standard hardware. Whenever compatible with the instrument specific requirements, **standard hardware and software solutions shall be used.**

[R-ICSS-165] D/ / /
The implementation to any FCS special device for an instrument is subject to a prior ESO approval (typically at PDR or latest at FDR).

3.13 Common Software

[R-ICSS-167] / / /
ESO provides common software for the control of standard devices and to help building the FCS stand-alone GUI.

3.14 Naming conventions

[R-ICSS-169] / / /
The *Git* modules belonging to FCS shall use the following naming conventions:

[R-ICSS-170] / /I/
1. The binaries shall follow the lowerCamel naming convention:
   a. Server shall be <prefix><fcs name>Devmgr, e.g. *micFcs1Devmgr*.

[R-ICSS-171] / / /
2. Libraries shall follow lower case and long names separated with dash:
   a. lib<prefix>-<FCS name>-<module>, e.g. *libmic-fcs1-wdglib.so*

[R-ICSS-172] / /I/
3. Special devices shall be in one library under the module devices.

[R-ICSS-420] D/ / /
The device names shall have a maximum length of 10 characters to simplify the layout of the engineering graphical interfaces.

4. DETECTOR CONTROL SOFTWARE (DCS)

[R-ICSS-174] D/ / / The Detector Control Software (DCS) shall carry out all the control and acquisition functions belonging to the detector(s) and camera(s) of an instrument.

[R-ICSS-175] D/ / / **Unless is a mosaic system, one DCS is normally responsible for one detector or camera only**. If several cameras are present, there shall be **one DCS for each of them.**

[R-ICSS-176] D/ / / A DCS in general consists of one part, which runs on Detector WS and one part, which runs on the IWS. The DWS part shall be responsible for the interface to the detector hardware, also called Detector Front-end Electronics (DFE) in the case of scientific detectors or directly to the cameras in the case of WFS and technical cameras.

[R-ICSS-177] D/ / / The WS part shall be responsible for the detector simulation, if not available (e.g. in a development or test environment) and for the API to OCS.

[R-ICSS-178] / / / The detectors or cameras to control are:

- Optical and Infrared detectors (in particular CCDs)
- WFS Cameras
- Technical Cameras

[R-ICSS-179] / / /

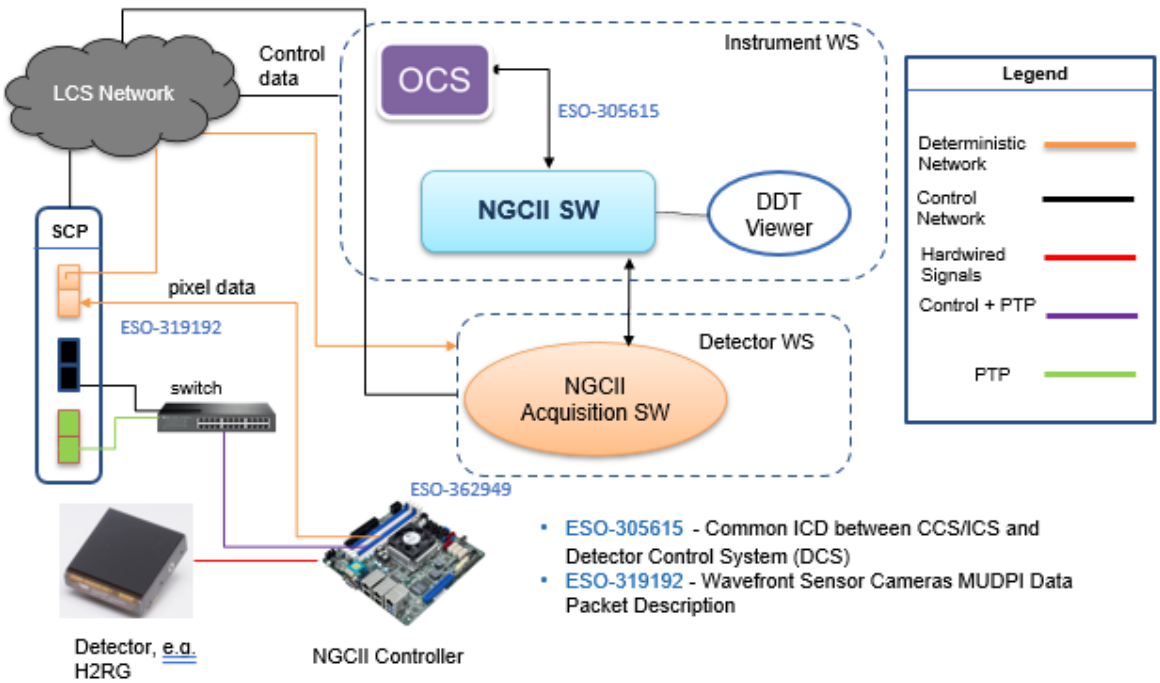

Figure 5: Scientific DCS Architecture.

[R-ICSS-180]
/ / /

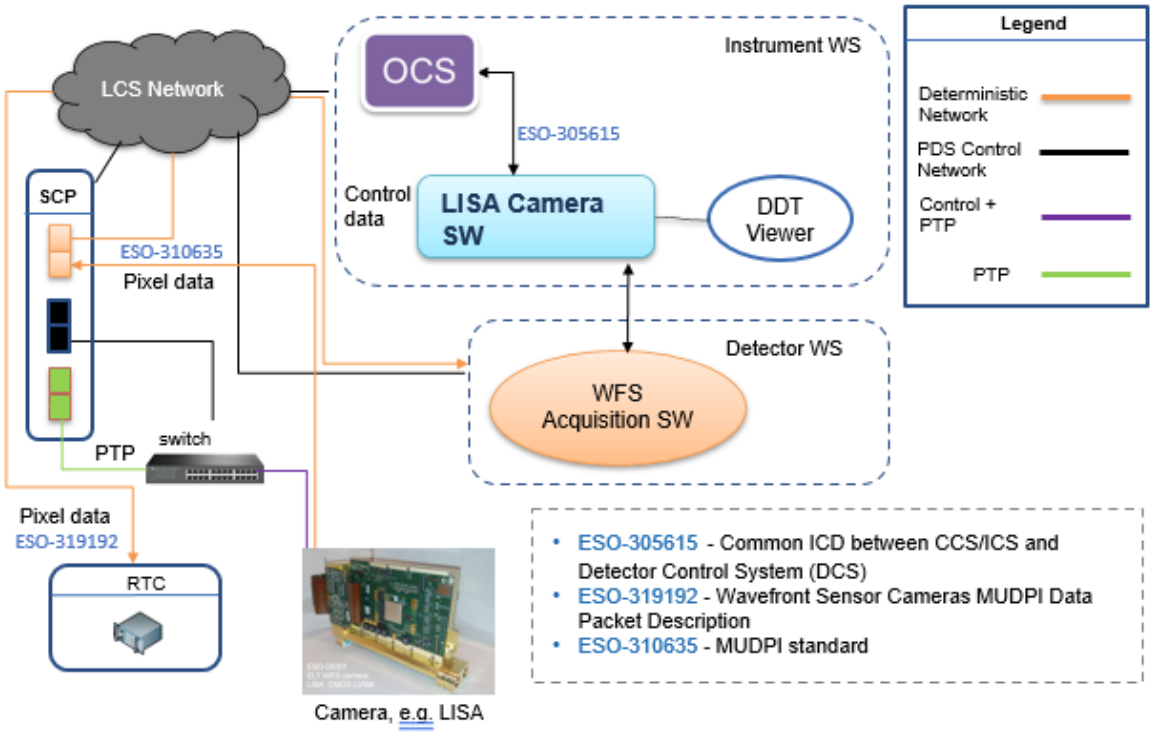

Figure 6: WFS DCS Architecture.

[R-ICSS-181]
/ / /

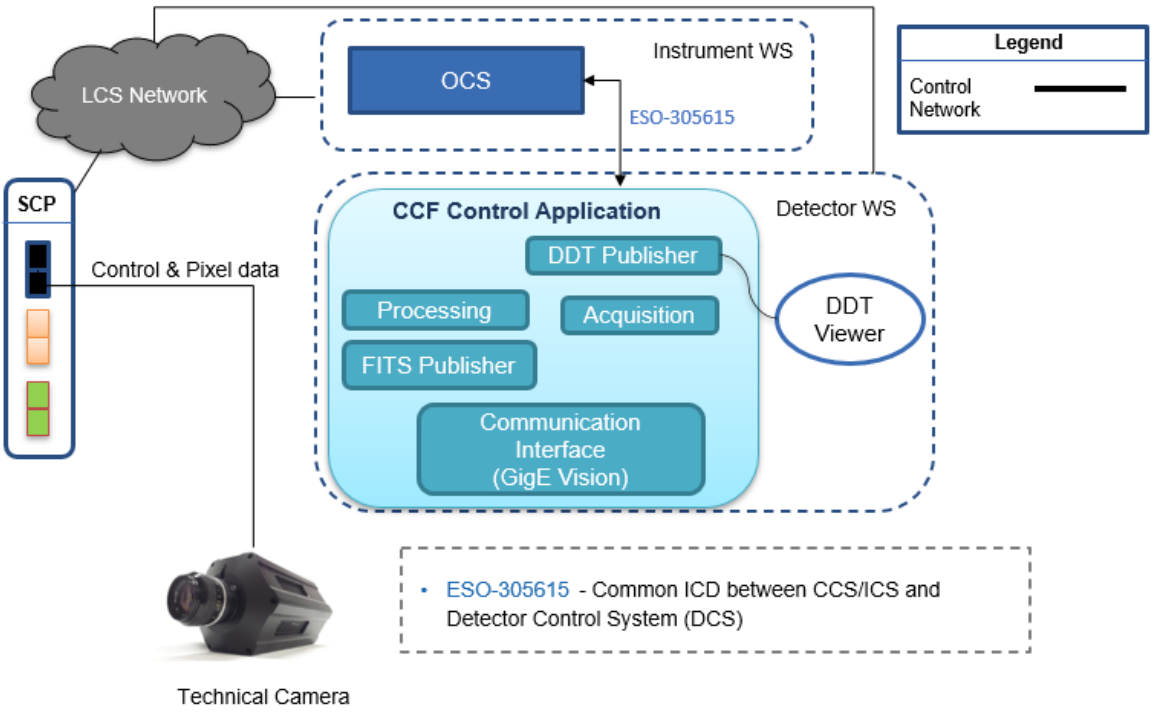

Figure 7: Technical Camera DCS Architecture.

4.1 States

[R-ICSS-183] / / / All the **standard DCS states,** and the commands to change state, are specified in [AD10]

4.2 Commands

[R-ICSS-185] / / / All the **standard DCS commands**, the command syntax and conventions are specified in [AD10].

4.3 Parameters

[R-ICSS-187] D/ / / The DCS software shall maintain all its parameters in the OLDB.

[R-ICSS-188] / / / DCS parameters can be grouped into the following categories:

- Configuration parameters (see also section 7.1)
- Setup parameters (set values, see point 3 below)
- Status parameters (actual values, see point 3 below)

[R-ICSS-189] D/ / / The following rules are mandatory:

1. All **parameters values** (e.g. number of detectors, size of each detector, binning factor etc.) shall be stored into the database. They **shall never be *hard-coded* in the Software**. The OLDB shall be populated from the instrument configuration.

2. The complete status of the detector-WS shall be stored into the database in order to enable GUIs to monitor the status of the detector WS processes. This implies that the database is updated when the hardware status is read or the software status changes**.**

3. Parameters set by a command, like integration time, **shall not be modified by the detector software until a new command to change the value of this parameter is received**. This is called the **set value** of the parameter. Parallel to the set value there shall be a value for the real status which is called the **actual value** of the parameter**.**

4. For actions like running an exposure, opening the shutter, performing a read-out, **there shall be a status parameter indicating that the action is going on or was completed or failed.**

5. Set values shall be checked for validity (e.g. known filter name or slit width in the allowed range)**.**

4.4 FITS header keywords

[R-ICSS-191] / /I/ **1. The DCS FITS header part shall contain:**

a. The complete setup of the related exposure

b. Auxiliary information and possibly statistics (e.g median, r.m.s.) on parameter values which can change during an exposure (e.g telemetry values).

[R-ICSS-192] / /I/ **2. The DCS shall be able to produce FITS header data in simulation** (test) **mode**, i.e. when the detector or cameras are not available**.** Simulation must be clearly stated in the FITS header.

[R-ICSS-193] D/ / / 3. In order to avoid time consuming copying of large amounts of detector data on disk, **DCS shall provide an on-line database attribute, where the total size of the FITS header can be specified.** When creating the FITS file, **DCS must** read this attribute, **reserve the corresponding disk space for the header and append to it the detector data. The header space is then filled in by the PDM at a later stage.**

[R-ICSS-194] D/ /I/ 4. Keywords appearing in the FITS header shall comply with the rules described in (AD7) and shall be checked against the associated dictionary for syntax validity (see AD7 for more information on data interfaces)**.**

4.5 Stand-alone mode

[R-ICSS-196] D/ / / The stand alone mode is used for detector tests at the telescope site and in laboratories. **An engineering GUI shall be provided** to support this mode**.**

4.6 Logging

[R-ICSS-198] D/ / / DCS shall use the CII logging service for logging.

[R-ICSS-199] / /I/ T**he following events shall be logged:**

1. Commands like start, pause, abort, continue, end, etc.

2. Any commands and **errors** during an exposure.

3. Detector **temperature values**, e.g when they change by a certain amount when the detector is in operation.

4.7 Safety

[R-ICSS-201] / / / Usually the detector is protected by hardware against any damage due to software faults.

[R-ICSS-202] D/ / / Critical parameters shall be identified and accordingly monitored:

1. Detector voltages
2. Detector temperatures
3. Detector vacuum

4.8 Simulation

[R-ICSS-204] D/ / / 1. Three levels of simulation shall be implemented:

a. **Simulation for the entire detector WS at the IWS level**. This level is used e.g. to test other Software, like the OCS, when no detector WS is available or connected to the instrument LAN**.**

b. **Simulation for the whole DFE at Detector WS level**. This level is used e.g. when the hardware associated to the camera is not available. This level of simulation is achieved when the application Software skips all actions towards the DFE hardware interface and assumes that the hardware behaves as expected**.**

c. **Simulation of the hardware at DFE level**. This level is used e.g. to test and debug the DFE embedded software**. It might be valid only for NGCII controller.**

[R-ICSS-205] D/ / / 2. The detector WS shall be able to **simulate readout frames** (windowed and binned) so that, e.g. data transmission can be tested without a DFE**.**

[R-ICSS-206] D/ / / **3.** **Simulation**, at whatever level, shall be repeatedly indicated to the user. There **shall be no "hidden"** simulation which can corrupt a real observation**.**

[R-ICSS-207] / /I/ 4. The **simulation level** used shall be clearly stated **in the FITS header**.

4.9 Performance requirements

[R-ICSS-209] / / / General performance requirements are outlined in [AD3]**.**, in particular:

1. Control flow (commands, acknowledgements, replies)
2. The maximum accepted delay for updating the status display.
3. Time-out values.
4. Times for start-up and shut-down.

[R-ICSS-210] D/ / / Furthermore, the design goal shall be to achieve the **highest possible duty cycle**. For example, the observing duty cycle for a number of repeated bias frames (dark frames with no integration) shall not be dominated by data transfers and image processing, but only by the read-out**.**

4.10 Failure Mode Operation

[R-ICSS-212] D/ / / 1. In case that data transmission to the WS fails, the detector-WS shall retain the data and wait for instructions from the WS to re-transmit the detector data**.**

[R-ICSS-213] D/ / / 2. After a crash of the detector-WS and re-initialization, the DCS shall check (via the DFE) if an integration is still active. If so, it should be able to either resume the control of the integration or readout the detector, depending on a user confirmation or urgency for readout.

[R-ICSS-214] D/ / / **3. In case of a non-fatal error, the DCS shall save the detector data whenever possible**, for example, when the detector temperature exceeds an absolute maximum value after a long exposure time and the alarm was not acknowledged by the user within a certain time**.**

4.11 Data transmission over instrument LAN

[R-ICSS-216] D/ / / ESO defines standards for data transmission within a DCS: RTMS [RD10] for scientific and WFS cameras, and GigE Vision for technical cameras. These standards shall be used for data transfer over the instrument LAN.

4.12 Data format

[R-ICSS-218] D/ /I/ 1. Data from scientific detectors shall be stored in **FITS format (uncompressed)** on the workstation disk. This is the official format for data products. The naming conventions for the data FITS file(s) are specified in AD7**.**

[R-ICSS-219] / / / 2. Whenever possible, **binary data format is preferred** without the need for computations on pixels with the optional FITS keyword 'BSCALE'. 'BZERO' should be used to adjust the offset for unsigned 16-bit integer data**.**

4.13 Data Display Tool

[R-ICSS-221] D/ / /
Quicklook tools shall be provided for instantaneous and continuous detector data display (possibly pre-processed). This functionality is needed to display the data from scientific detectors, as well as from WFS or technical cameras (guiders, finders, slit viewers, etc.).

[R-ICSS-222] / / /
DDT is the standard software provided by ESO for implementing quicklook tools.

[R-ICSS-223] / /I/
The quicklook tools shall meet the requirements for data display specified in [AD3].

4.14 Disk space availability

[R-ICSS-225] D/ / /
1. Observations shall not be limited by applications competing for disk space on the IWS. Detector data shall be stored on dedicated disk areas which are not used (filled up) by other software (e.g. image processing packages). It might be desirable, also due to speed considerations, to use separate disks on the IWS for detector data**.**

[R-ICSS-226] D/ / /
2. Disk space availability shall be checked before starting a new exposure**.**

[R-ICSS-227] D/ / /
The DCS shall monitor the amount of free and total disk space and trigger an alarm in case there is no enough space for operations.

4.15 Other requirements

[R-ICSS-229] D/ / /
1. The DCS shall support **windowed** (possibly also **binned**) **readout** for 2-D detector arrays**.**

[R-ICSS-230] D/ / /
2. It may be required that **detector images** can be **displayed** directly **with** almost the **same orientation** as on other display units (acquisition cameras) **or** that they are to be stored in a **different orientation** (e.g reversed, up side down, reversed and up side down), however, without re-binning in order to preserve the original (raw) pixel values**.**

4.15.1 Shutter control

[R-ICSS-232] D/ / /
1. The DCS shall be responsible for accurate exposure times and normally controls the shutter itself. When the ICS must control the shutter, e.g. because of hardware design constraints, then the exact **times for opening and closing are given by the DCS**. The time information is **based on the Time Reference System** (TRS) which is available for **controllers and WFS cameras.**

[R-ICSS-233] D/ / /
2. The exposure times during opening and closing the shutter shall be taken into account, in particular for slower shutters, to maintain the linearity (flux versus

integration time) on the detector chip. Accurate exposure times are especially necessary for short times when, for example, bright standard stars are observed which must not saturate a CCD**.**

4.16 Graphical User Interface

[R-ICSS-235] D/ / / DCS shall provide a Graphical User Interface (GUI) for engineering purposes to support the stand-alone mode.

[R-ICSS-236] D/ / / Furthermore, a GUI for the image display shall also exist. Standard DDT GUIs are delivered with the *DDT Package.*

[R-ICSS-237] D/ / / Any other auxiliary GUI, which may be needed, must be implemented with the ESO UI Toolkit [AD6] and following the guidelines for developing control GUIs (see [RD7]).

[R-ICSS-238] / /I/ DCS graphical interfaces shall meet the requirements for graphical interfaces specified in [AD3].

4.17 Standards

[R-ICSS-240] / / / ESO defines a standard controller for optical and infrared scientific detectors (NGCII controller).

[R-ICSS-241] / / / ESO defines a set of standard WFS cameras: LISA and ALICE.

[R-ICSS-242] / / / ESO defines a standard communication protocol for COTS technical cameras (GigE Vision).

[R-ICSS-243] D/ / / ESO defines a standard interface for communicating with a DCS. This interface shall be used by the high-level software controlling or supervising the DCS [AD10].

4.18 Common Software

[R-ICSS-245] / / / ESO provides common software for each of the standard DCS types described in section 4.18. The functionality provided by these packages fully covers the requirements for the majority of the DCSs.

4.19 Modules naming conventions

[R-ICSS-247] D/ / / **The *software* belonging to DCS shall use the following naming conventions:**

[R-ICSS-248] D/ / / *1.* *<dcs name>* is the main module that contains all the source code about a particular DCS.

[R-ICSS-249] D/ / / 2. The binaries shall follow the lowerCamel naming convention:

[R-ICSS-250] D/ / / a. Server shall be <prefix><dcs name>Control, e.g. micDcs1Control.

[R-ICSS-251] D/ / / 3. Libraries shall follow lower case and long names separated with dash:

[R-ICSS-252] D/ / / a. lib<prefix>-<dcs name>-<module>, e.g. *libmic-dcs1-wdglib.so*

5. OBSERVATION COORDINATION SYSTEM (OCS)

[R-ICSS-254] / / / The Observation Coordination System (OCS) is the highest layer of the control Software (see Figure 8). It consists of:

- System Supervisor (SysSup) process, responsible of the overall monitoring and supervision of the subsystems.
- Observation Coordination Manager (OCM) process, responsible for coordinating and executing the data acquisition of single exposures. It is not responsible of the instrument setup.
- Data Product Manager (DPM), responsible collecting and archiving the results of exposures in the final FITS file and for sending it to the Online Archive System (OLAS).
- Templates, defining and running sequence of exposures.
- Auxiliary processes for controlling secondary loops.

[R-ICSS-255] / / /

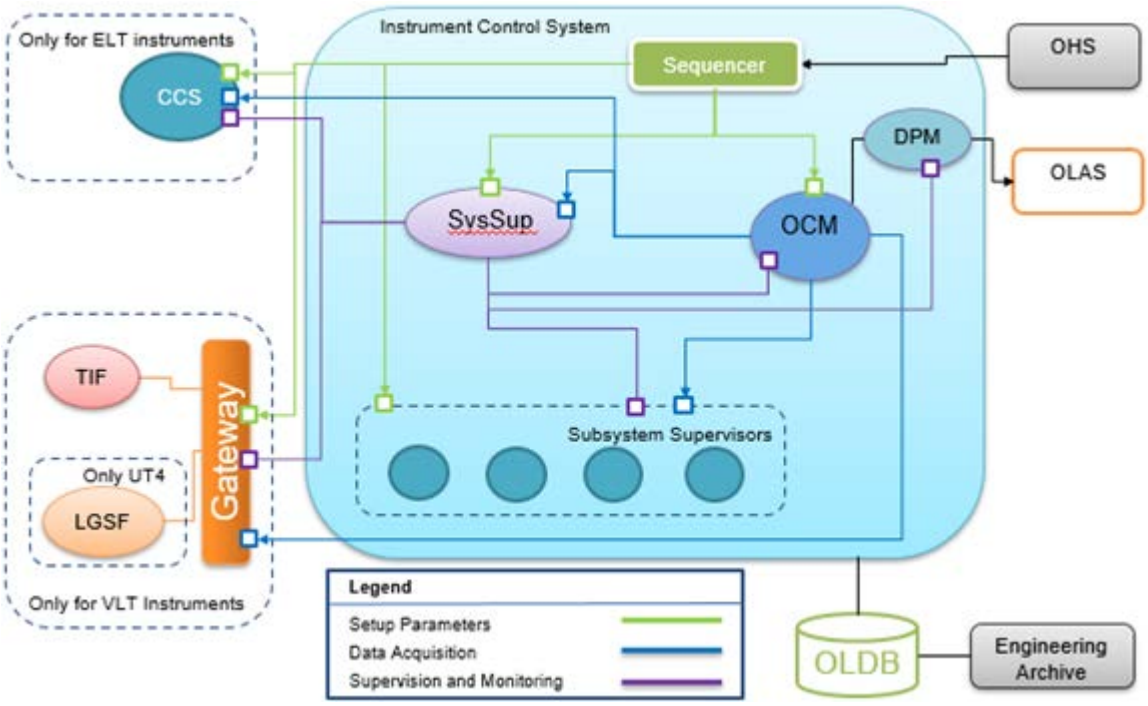

Figure 8: OCS Software Architecture.

[R-ICSS-256] / / / In the simplest case, an exposure involves to setup instrument, detectors and telescope, to collect light on a detector, to read-out the detector and to store the detector data and FITS header on disk. A more complex type of exposure is used e.g. for infrared observations where the Detector Control Software has to readout and average a number

of short integrations, to repeat a measurement several times, to average the measurements, etc.

[R-ICSS-257] D/ / / The result of an exposure shall contain the complete set of data from the read-out operations as well as a full description of them. The description of an exposure consists of a FITS header (or FITS headers in case that an exposure produces more than one output file) and logging information.

[R-ICSS-258] D/ / / An exposure may also require two or more different instrument setups. For example, the alignment of the slits in the Multi-Object spectroscopy mode needs two integrations on a CCD. After the first integration (targets) the CCD charge is shifted and the instrument setup is changed before the next integration (slits) is started. After the two integrations and readout of the CCD the image is stored on disk. The FITS header (there is only one) shall contain the status of both integrations.

[R-ICSS-259] D/ / / **OCM process shall be able to coordinate "overlapping" exposures:** in order to achieve the highest throughput of consecutive exposures, it must be able to perform the next instrument setup in parallel to the readout of a detector, as illustrated in Figure 9.

[R-ICSS-260] / / /

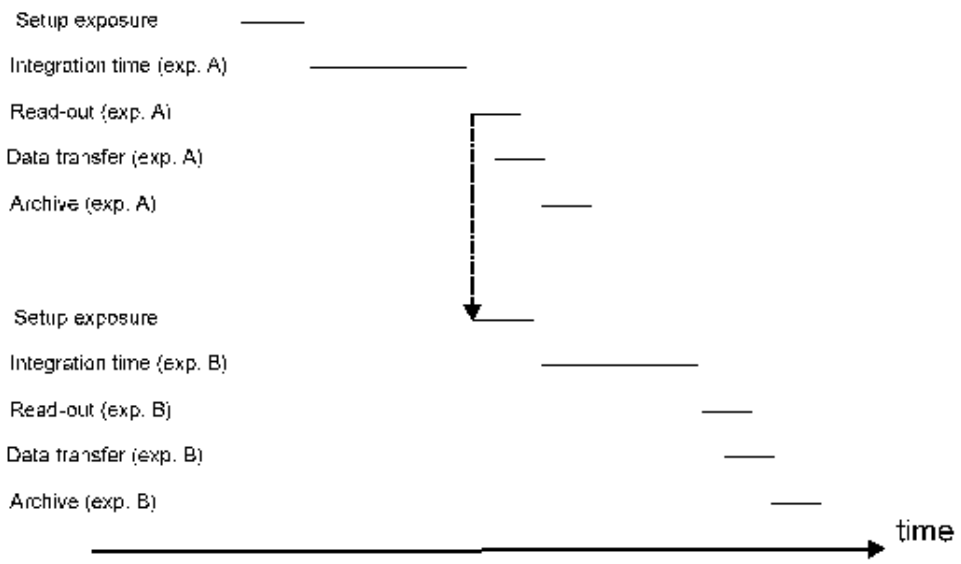

Figure 9 Overlapping exposures

[R-ICSS-261] / / / **OCM may have to manage independent exposures which run in parallel** e.g when an instrument has two detectors. The exposures for each detector could be "overlapping".

[R-ICSS-262] / / / Each of the parallel exposures may have its own setup. For example, one exposure could be a dark current measurement and the other a scientific one. This is only possible if data acquisitions/exposures are independent of each other, (or that subsystems providing data to both exposures support concurrent data acquisions).

[R-ICSS-263] D/ / / **The FITS file containing the results of one exposure**, even if taken for test purposes, **shall be always archived.**

[R-ICSS-264] D/ / / **The archiving operation shall not affect the observing cycle**, i.e. the next exposure shall be started while the results of the previous one are being archived. For this reason, a **separate process**, called **Data Product Manager** in Figure 8, **shall be dedicated to the archiving operations.** The DPM must take care of collecting the metadata files from the various sub-systems and merging them into the final FITS file. Once the final FITS file is ready, DPM must inform the OLAS that a new file is ready to be archived. If an error is encountered by the DPM, OCM shall be able to report this error back to the client, which started the exposure (normally the Sequencer, running a template)

[R-ICSS-265] D/ / / **OCS shall also include** the set of **templates** (scripts and signature files) building the Instrument Package (IP)**.** Templates are responsible for the definition of sequence of exposures, executed through the Sequencer.

[R-ICSS-266] D/ / / Instruments may have one or more **technical CCDs**, used e.g**. for secondary guiding**, that may require to take exposures continuously. When this is the case, **OCM shall be able to start/stop taking continuous exposures with this detector**, independent (and in parallel) of the normal exposures. **It shall be possible to save a frame of this technical detector** (i.e. stop loop, take a single frame, resume loop). **The saved frame shall be archived by the DPM with the proper header**

[R-ICSS-267] / / / If OCM is requested to acquire data from a technical CCD, the technical CCD will either:

- already be integrating/acquiring frames continuously, and OCM command to technical CCD will simply record a subset of this data as requested, or
- it is idle/not acquiring anything and will then automatically start when OCM commands it to begin recording.

5.1 States

[R-ICSS-269] / / / All the **standard OCM states**, and the commands to change state, are to be specified in the OCF documentation.

5.2 Commands

[R-ICSS-271] / / / All the **standard OCM commands**, the command syntax and conventions are to be specified in the OCF documentation.

5.3 Parameters

[R-ICSS-273] D/ / / OCM parameters shall be stored in the on-line database on the workstation. Each parameter has, likewise ICS and DCS, separate items for set value, actual value, state flags, etc. and has to be checked against name/range validity**.**

5.4 Safety

[R-ICSS-275] D/ / /
Before moving devices which could over illuminate sensitive optical components, the outcome of the setup shall be evaluated and rejected in the case it is considered dangerous.

5.5 Observation Templates

[R-ICSS-277] D/ /I/
The templates shall setup the complete light path so that it executes exactly the same way regardless of the previous state of the system.

1. The only exception may come from the acquisition template setting up a part of the light path and the observing template setting up the rest of the light path, in any case the result shall be repeatable.

2. That means, for example, that if the observing template aborts then it must be possible to repeat it, that is if the observing template executed telescope offsets then the telescope should be returned to it's original RA and DEC coordinates as part of the recovery.

3. In case of calibrations the full instrument light path should be set by each template. This includes turning off not needed lamps at the start of the template.

5.6 Calibration Templates

[R-ICSS-279] D/ /I/
1. There is no need to turn off lamps at the end of a particular calibration template, because the next calibration may use the same lamp device.

2. To make sure that the lamp is not left on for a long period of time, the lamp shall be configured to turn off automatically after a predefined period of time (that is as long as the longest expected calibration exposure).

3. The above implies that the lamp setup has to be send before each exposure is started, regardless if the lamp is already on or not, to reset the off timer.

4. At the end of the daytime calibrations OB shall be placed in day mode, which in turn should turn off all lights and close all shutters.

5.7 Execution of exposures

[R-ICSS-281] / / / An observing run at the observatory consists in loading and running Observation Blocks (OBs) through the Sequencer. OBs are basically a sequence of templates. The sequence of actions/commands to be executed within a template is described in the Template Script File, while the parameters to be used are specified in the Template Signature File. The user can modify the value of the parameters specified in the signature file (normally through the Phase 2 preparation software), not the contents of the script file. Normally the actions contained in a template script are commands to be sent to the various subsystems. The most typical commands are those needed to prepare and execute an exposure. Template scripts are the only authorized way to run, through the Sequencer, sequence of exposures, by sending the proper commands in the proper sequence to the subsystems.

[R-ICSS-282] / / / The following is a simplified example which shows how an exposure is executed:

1. The Sequencer extracts from the OB all the parameters and determines which have to be forwarded to which sub-system (FCS, DCS, and CCS or TCS in the case of VLT).

2. Sequencer sends one or more commands along with parameters to the various subsystems, as part of the execution of a template. The Sequencer refuses the setup when e.g an exposure is already active or the instrument is in a non-operational state.

3. Each subsystem performs checks whether a new setup is allowed and possible.

4. Each control system sets its own devices as much as possible in parallel to speed up the entire setup.

5. OCM process receives from Sequencer a START command. It notifies the sub-systems to start acquiring data.

6. Depending on the type of exposure, during the integration, The Sequencer may send to subsystems commands to perform special actions, e.g. grating step, as part of the template execution.

7. When the final detector data have been transferred to the IWS, OCM informs to DPM that all data are available.

8. The DPM collects and merges all information in the FITS header and informs the On-Line Archive.

5.8 Control of exposures

[R-ICSS-284] / / / The ABORT is a standard OCM command which is sent when the user presses the corresponding button in the OCS Control GUI or as part of a template. OCM does not support PAUSE and CONTINUE which might be supported by each DCS.

[R-ICSS-285] / / / The Sequencer GUI has also ABORT, PAUSE buttons. Their scope is however different: they apply to the running OB, while the OCS commands and corresponding GUI buttons apply to the running exposure.

5.9 Changes during an exposure

[R-ICSS-287] / / / Changes during an active exposure are possible only for a very limited set of parameters. The exact list depends on the instrument and on the current mode. A typical parameter which can be changed while an exposure is running is the integration time. Change of parameters is usually done by mean of the SETUP command and the appropriate parameters.

5.10 Exposure Types

[R-ICSS-289] / / / The list of possible and **allowed exposure types** is to be defined.

5.11 FITS header

[R-ICSS-291] / / / The requirements on the **format and contents of a FITS header** are given in [AD7].

5.12 Setup files

[R-ICSS-293] / / / The SETUP command allows specifying one or more setup files. The format and the type of setup files depends on each subsystem.

5.13 Templates

[R-ICSS-295] D/ / / OCM is in charge of executing and coordinating single data acquisitions. The **execution and coordination of** more **complex operations** is outside the scope of OCM and shall instead be **implemented in templates**. Exception to this rule is represented by **auto-guiding, active or adaptive optics functionality**, which some instruments are requested to implement, and which should better be **done in separate dedicated processes.**

[R-ICSS-296] / / / Examples of complex operations implemented in templates are:

- A telescope focus sequence

- A sequence of calibration exposures
- A sequence of telescope beam switch between object and sky while integrating.
- Execution of action B or C depending on the results of exposure A.

[R-ICSS-297] D/ / / On-line data processing required to be done in templates shall be performed with the support of an on-line data processing tool, such as ODP.

[R-ICSS-298] / / / All files related to templates, which build the Instrument Package, needed by the Phase 2 preparation software, are part of OCS. The only exceptions are the test and maintenance templates, which are instead part of MS (see section 7.2).

[R-ICSS-299] / /I/ The number and contents of the templates varies from instrument to instrument and the list of templates to be implemented shall be specified in the Instrument Software User Requirements Specification (ISURS) document or in a dedicated one.

[R-ICSS-410] D/ / / Templates shall provide a possibility to continue observations from the same point it was left in case of failure, obviously only when this does not affect scientific data quality. This could prevent science operations in losing additional time e.g., with offsetting templates, where failure on the last step could lead to repetition of the whole template.

[R-ICSS-411] D/ / /T Templates shall check the accessibility of subsystems that are needed for the correct execution, e.g. to verify that the telescope is not disabled in the acquisition templates. In case a subsystem is not enabled, the template shall attempt to enable it automatically after the confirmation from the user.

5.14 Graphical User Interface

[R-ICSS-301] D/ / / OCS shall provide at least two Graphical User Interface (GUI) panels:

[R-ICSS-302] D/ / / **1. OCS Control panel**. It normally occupies half of the screen at most (the other half being taken by the Sequencer GUI) and contains the essential instrument status information needed to follow the execution of OBs. It shall also contain the buttons (ABORT, PAUSE etc) needed to take an action on the running exposure**.**

[R-ICSS-303] D/ / / **2. OCS Status panel**. This panel contains detailed information of the status of the instrument, with particular emphasis on possible abnormal conditions, which may trigger alarms**.**

[R-ICSS-415] D/ /I/ The OCS Control panel shall contain at least the following information:

- Overall state of the Instrument
- Individual state of the subsystems.
- Exposure status of the detector system being used.
- Summary status of instrument alarms.
- Subsystem error conditions.
- Status of auxiliary systems like secondary guiding.

- Main settings of instrument hardware functions.

[R-ICSS-416] D/ /I/ The OCS Status panel shall contain at least the following information:

- Actual instrument mode.
- Actual settings of instrument functions.
- Updated values of instrument sensors.

[R-ICSS-304] D/ / / All OS GUIs shall be implemented with the Control UI Toolkit. OCS GUIs shall meet the requirements for graphical interfaces specified in [AD3].

5.15 Standards

[R-ICSS-306] / / / The **interface between OCS and the Observation Handling Tool** is described in [AD8] and **is applicable to all Instruments.**

[R-ICSS-307] / / / The **interface between DPM and the On-Line Archive System** is described in [AD9] and **is applicable to all Instruments.**

[R-ICSS-308] D/ / / The usage of *OCF* is mandatory to implement the core functionality of OCS.

5.16 Common Software

[R-ICSS-310] / / / The basic functionality of the OCM processes and DPM is provided by OCF.

5.17 Modules naming conventions

[R-ICSS-312] / / / **TBD**

6. ADAPTIVE OPTICS CONTROL SYSTEM (AOCS)

[R-ICSS-314] / / / The AOCS is the control system dealing with the internal AO capabilities of the instruments. It uses WFS cameras as sensors and deformable mirrors as actuators to compensate for the atmospheric turbulence effect on the light entering into the optical and infrared detectors. The main component of the AOCS is the an AO Real Time Computer (RTC). The AO RTC is primarily responsible of the implementation of the main AO control loops. ESO provides two frameworks aimed to help implementing the AO Control System: the ICS Framework [RD16] and the RTC Toolkit for [RD17]. More details of the specific RTC standards can be found in [RD17]. An AOCS can also be part of an independent AO module providing services to one or more instruments like MAORY in the ELT.

[R-ICSS-315] / / /

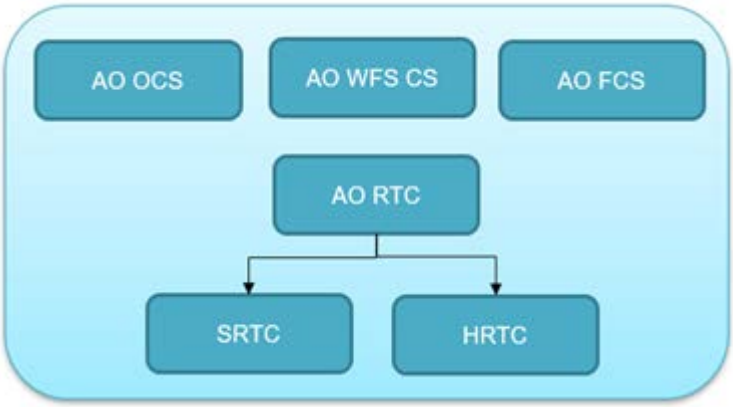

Figure 10: AO Control System product tree.

[R-ICSS-316] / / / Details about AO RTC are described in a separate document, see [RD15].

7. MAINTENANCE SOFTWARE (MS)

[R-ICSS-318] D/ / /
The Maintenance Software module (MS) shall enable the user to define and keep control over the Instrument configuration and to perform verification tests of the instrument.

7.1 Instrument Configuration

[R-ICSS-320] D/ /I/
The configuration of the instrument shall be defined by a set of files:

- Instrument configuration files in the format defined by the CII Configuration service.
- Detector configuration files.
- Dictionaries files for image header data.

7.1.1 Privileges

[R-ICSS-322] D/ / /
Observers shall never be able to accidentally modify or corrupt important configuration parameters (detector clock voltages, software limits, reference temperatures, etc.), however, they might be interested to display them**.**

7.1.2 Change Instrument Configuration Parameters

[R-ICSS-324] / / /
It is important to **keep control over changes to** any of the files defining the **instrument configuration**.

[R-ICSS-325] D/ /I/
Actions which **change the instrument configuration shall be additionally logged** so that this information is then automatically stored in the CII logging system**.**

7.2 Maintenance and Verification procedures

[R-ICSS-327] D/ / /
1. The procedures needed to perform proper maintenance of the instrument and verify its correct behavior shall be listed in the Instrument Software User Requirement Specification (ISURS) document. Examples of such procedures are:

- Monitoring of motor current consumption
- Monitoring of liquid nitrogen tank level for the detectors
- All kinds of instrument alignment measurements, e.g. detector column to slit position

- Instrument focus
- Measurement of the total instrument efficiency
- Detector linearity and saturation levels.
- Automatic measurement of bad detector pixels

[R-ICSS-328] D/ /I/ 2. Unless tools provided by ESO already exist, **all these procedures shall be implemented as technical templates** and therefore be executed using the Sequencer. MS is also responsible for providing a technical Instrument Package, which includes Science and MS templates. The technical Instrument Package is not supposed to be used during an observing run, but may be needed during daytime operations or in the AIV phase**.**

[R-ICSS-329] D/ /I/ 3. All measurements and activities (e.g. alignment, noise measurements etc.) carried out during AIV and Commissioning at the Observatory shall be performed by technical templates.

8. ALARMS

[R-ICSS-331] D/ /I/ The list of abnormal situations which must trigger alarms shall be specified in the Instrument Software Requirements Specifications.

Conditions which can seriously endanger people and/or expensive material, shall be handled to bring the system in a safe state by hardware of by the interlock and safety system without the need of human intervention. At the same time an Alarm shall be triggered to notify the operators and request their intervention to resolve or workaround the problem and bring back the system to normal conditions.

[R-ICSS-332] D/ / / Software **alarms** in Instrumentation Software **will be listed in the Instrument Software Functional Specification document and detailed in the Instrument Software Design Description document.**

[R-ICSS-333] D/ / / **Instrumentation Software shall implement alarms using the CII Alarm Service.**

[R-ICSS-334] D/ / / In addition to the general rules and definitions imposed by the CII Alarm System, the Instrumentation Software defines the following mandatory rules:

1. The Instrumentation Software shall make sure that an alarm is triggered only if the value of the associated on-line database attribute is healthy, in particular:

a. The related device controller is operational and monitoring the value.

b. No error was detected during the last read operation from the hardware.

What above can be achieved by applying the appropriate Calculation Engine formula to the database attribute.

2. Alarm conditions must be clearly visible (e.g. color red or bigger font) in the status panels**.**

[R-ICSS-335] D/ / / Alarms shall be defined at PAE including thresholds, possible associated audio resources and display color. The alarms definition shall allow an easy definition and refinement to facilitate its maintenance during the instrument life time.

9. INTERFACES

9.1 Graphical User Interface

[R-ICSS-338] D/ / / All GUIs provided with an instrument shall be built using the Control UI Toolkit [AD6], and follow the rules and guidelines specified in [RD7]. This document gives general guidelines and rules for user interaction, status display and "Look and Feel", which are applicable to all instrumentation software.

9.1.1 General Guidelines

[R-ICSS-340] / /I/ The following rules are given for the development of Instrumentation GUIs:

1. **Panels and windows shall not pop up and disappear automatically**. The display of new panels or the replacement of existing ones is usually performed after an action by the user and depend on the context

2. **The placement of panels shall be static**, i.e. panels appear, as much as possible, always at the same place on a screen

3. **A GUI must not automatically close a panel**

[R-ICSS-341] / /I/ The following general guidelines are given for the development of Instrumentation GUIs:

1. The number of panels which are displayed simultaneously shall be kept to a minimum. Only panels which are needed for the current context shall be shown, for example, when the user selects spectroscopy mode for a multi-mode instrument, then the panel for imaging mode may not be needed.

2. Minimum status information is to be shown automatically, while status of other parts of the system shall be displayed on request.

3. Action related status, which changes due to an action by the user, shall be displayed close to the input (e.g push button, radio button). For example, the status "exposure paused" is shown close to the "Pause" button.

4. Push buttons and other action widgets should be preferably automatically disabled and enabled depending on the status, e.g when an exposure is in status “paused”, the button "pause" must be disabled and "continue" enabled. Panels, which implement this optional feature, must also provide a way (e.g. a dedicated menu option) to disable this functionality (e.g. all buttons enabled, independently from the context), to get out of deadlock situations.

9.2 Performance Requirements

[R-ICSS-343] / / / The performance requirements for immediate and quick responses on a local user station are given in [AD3]

[R-ICSS-344] D/ / / Remote control via RAF or G-RAF is nowdays a normal activity to monitor or control the instrument software. Graphical interfaces shall avoid using high refresh rates whenever is possible to minimize latencies.

9.2.1 User Station

[R-ICSS-346] / / / VLT: The standard **User Station** of an instrument **consists of two screens** (typically the double-screen console of the User WS):

[R-ICSS-347] / / / 1. Screen 1. Sequencer GUI on the left and OCS Control GUI on the right.

This screen in general provides several workspaces, such that other GUIs can run on demand in one of the secondary workspaces. Example:

Workspace 1: Sequencer GUI and OCS Control GUI.

Workspace 2: OCS status GUI

Workspace 3: FCS stand-alone GUI

Workspace 4: DCS stand-alone GUI

[R-ICSS-348] / / / 2. Screen 2. Detector(s) DDT GUI and CCS Alarms GUI

An extension of the User Station to more than two screens requires a prior agreement with ESO.

9.3 Programmatic Interface

9.3.1 Interface to Observation Handling Tool

[R-ICSS-351] D/ / / The interface between the Phase 2 preparation software, running on the OHS WS, and the Sequencer, running on the IWS, is described in [AD8]. **The usage of this interface is mandatory for the INS.**

9.3.2 Interface to on-line Archive

[R-ICSS-353] D/ / / The interface between OS and the on-line Archive is described in [AD9]**. It is mandatory to follow it.**

9.3.3 Interface to TCS (only for VLT instruments)

[R-ICSS-355] D/ / / The interface between OS and TCS is described in [RD6]. **It is mandatory to follow it.**

9.3.4 Interface to CCS (only for ELT instruments)

[R-ICSS-357] D/ / / The interface between INS and CCS is described in [AD11]. **It is mandatory to follow it.**

10. INSTALLATION

[R-ICSS-359] / / /T 1 It shall be possible to rebuild and install from scratch the complete Instrument Software package, including the Instrument Configuration files. The *Git* is the version control system used for archiving operations. Instrument software shall be archived into the ESO Git server.

[R-ICSS-360] D/ / / 2 The installation procedure shall be based on waf/wtools.

10.1 Deployment

[R-ICSS-419] D/ / / Processes shall be deployed using Nomad [RD11].

10.2 Start-up / Shut-down

[R-ICSS-362] / /I/ 1 Standard procedures for start-up and shut-down of instrumentation software and individual processes or GUIs shall be provided. This procedures shall be implemented as Sequencer engineering scripts.

[R-ICSS-363] / / /T 2 It shall be possible to shut-down and re-start an INS subsystem (e.g FCS) without necessarily re-starting the whole INS package.

[R-ICSS-364] / /I/ 3 Some INS modules (FCF, DCS) shall provide for their own start-up/shut-down script when they are used in stand-alone mode.

11. SYSTEM ATTRIBUTES

11.1 Safety

[R-ICSS-368] D/ / / Measures shall be taken to meet any special safety requirements for personnel or plant, instrument and telescope, with acceptable probabilities of success. Topics to consider include:

- Validation of control inputs and outputs
- Safe states on failure
- High integrity alarm systems and interlocks
- Containment of errors
- Correct sequencing of operations
- Graceful degradation philosophy
- Compliance with ESO General Safety Requirements.

11.2 Security

[R-ICSS-370] D/ / / The following security aspects shall be considered:

- Restriction of unauthorized access
- Data security under fault, misuse and maintenance conditions
- Data recovery procedures
- Logging and archiving of historical data.

11.3 Availability

[R-ICSS-372] D/ / / The Technical Specification for the instrument shall specify the required level of availability and the mean time between failures (MTBF) of the instrument.

[R-ICSS-373] / / / For each instrument, define the levels of availability and MTBF required to the software and the measures to be taken to achieve them (redundancy, error checking, error correction, back-up or standby, rejection of human errors etc.).

11.4 Maintainability

[R-ICSS-375] / / / Cover all provisions made in the design to ensure that the software will comply to the required levels of availability and maintainability.

[R-ICSS-376] / / / **1. System Maintenance Philosophy**

The Maintenance Software (MS) module provides support to on-line or off-line maintenance activities of the instrument.

[R-ICSS-377] / / / **2. Maintenance Facilities and Procedures**

- Preventive maintenance schedules and effectiveness
- Fault traceability to lowest software level
- Resident test and diagnostic software
- Sampled and statistical outputs.

[R-ICSS-378] / / / **3. Resource Requirements**

- Test and simulation equipment
- Test software
- Manpower

[R-ICSS-379] / / / **4. Effectiveness.** Possibilities for degraded operation during fault conditions:

- Criteria allowing degraded operation
- Step-by-step procedures
- Recovery of full operation.

11.5 Adaptability and enhancement potential

[R-ICSS-381] / / / Adaptability will include ease of tuning for a variable demand and facility for incorporating foreseeable functional and performance extensions. Areas for concern are:

- Ease of optimization/tuning
- Storage flexibility
- Support of new mechanical, optical, data processing features
- Increase of automatic features.

[R-ICSS-382] / / / Describe in detail the facilities and capabilities, if any, which will enable the package to be adapted or extended, as already foreseen.

11.6 Documentation

[R-ICSS-386] / /I/ High quality documentation is required. The software of each instrument shall be documented at two levels:

1. For integration, operation and maintenance: this includes the internal organization of each module with the overview of the main software architectural issues, the services provided to users and to other software, all the interfaces, the installation, the troubleshooting, etc.

2. For final user: the user interface, the most typical procedures, including maintenance and training. Although the software is the main interface for the final user, such information should be arranged in the general User Manual of the Instrument, where all the operative aspects, involving software as well as mechanics, optics, etc., are described.

[R-ICSS-387] / /I/ Documentation shall be delivered in electronic form, according to the standard formats specified by ESO (at present MS Word and PDF). All documents, but in particular those for the final users (User and Maintenance Manual, Acceptance Test Plan), are subject to changes also after commissioning, i.e. after responsibility over the Instrument and its documentation has moved from the Consortium to the Observatory. **It is** therefore **mandatory that the electronic format of all documents is the same used internally at ESO at the time of commissioning.**

[R-ICSS-388] / /I/ Database documentation shall include purpose, units of each point, validity ranges and allowed values.

12. DEVELOPMENT FACTORS

12.1 Development Process

[R-ICSS-391] / / / It is recommended to follow an agile development process. Agile development is nowadays a common practice in software development. Some of the benefits are:

- It increases the quality of the deliverables.
- Provide better estimates.
- Be more in control of the project schedule and state.

12.2 Design considerations

[R-ICSS-393] D/ / /
The software architecture of each instrument shall be according to the model described in the present document. The use of the basic software stack and the ELT development standards, as well as the usage of the IFW, whenever possible and applicable, is mandatory.

[R-ICSS-394] D/ / /
The design of user interfaces shall be according to [AD6]

12.3 Test requirements

[R-ICSS-396] D/ /I/
As a general principle, each module shall be tested first independently by unit tests. Source code accessing external resources like databases or network devices shall be tested using mocking libraries. More complex tests including several entities shall be tested using integration tests.

12.4 Hardware Obsolecense

[R-ICSS-398] / /I/
The final target WSs shall be bought six months before PAE to minimize the soon obsolecense of the hardware.

12.5 Software Evolution

[R-ICSS-400] / /I/
During development, instrument software shall be regularly updated to the latest version of the ESO products including DevEnv, IFW and RTC Toolkit.

PAE shall be granted with the most recent version of the ESO Software products (DevEnv, IFW and RTC Toolkit) supported by the corresponding hardware if the official release is ready for 3 or more months in advance.

13. COMMISSIONING

13.1 Software Version

[R-ICSS-403] / /I/ Consortia shall update to the latest releases of the ESO products (DevEnv, IFW and RTC toolkit) before science verification if the official release is ready for two or more months in advance.

13.2 Device Configuration

[R-ICSS-405] / /I/ All device controllers should be fully and correctly parameterized before PAE. Some refinement for the motor tuning might be needed during commissioning but only when it depends on environmental aspects like temperature.

---- End of document ----