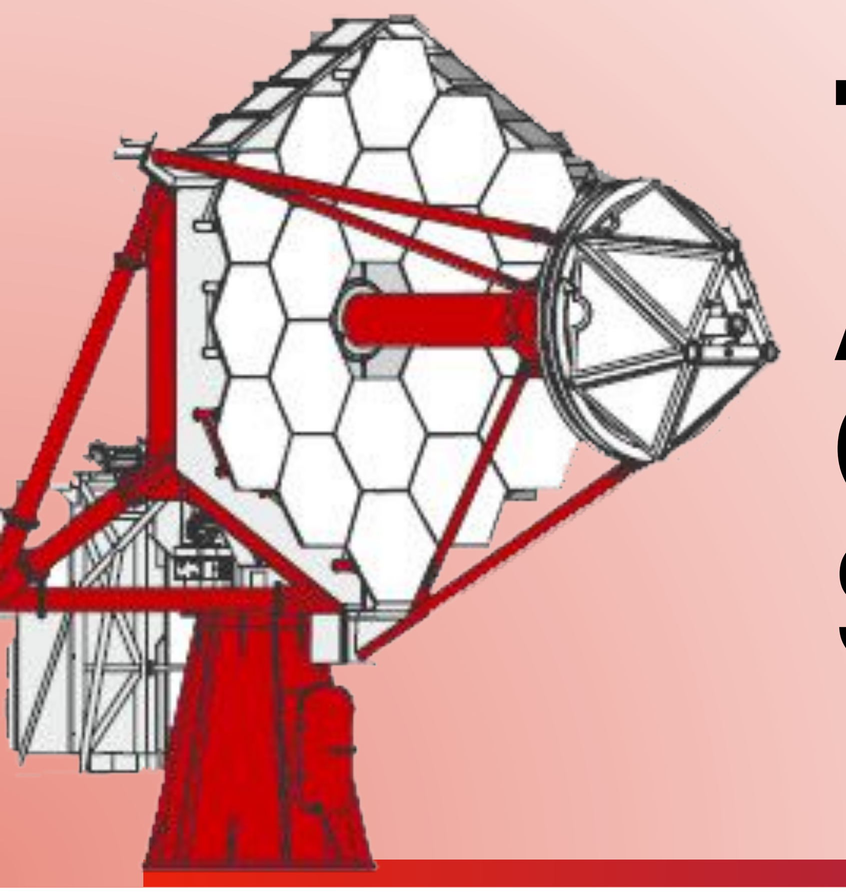

The Architecture of ASTRI Mini Array Cherenkov Camera Software Supervisor

M. Corpora (*INAF* - Istituto di Astrofisica Spaziale e Fisica Cosmica di Palermo),
A. Grillo, P. Sangiorgi, M. Capalbi, O.Catalano, G. Sottile, G. Tosti, A. Bulgarelli, F. Lucarelli,
N. Parmiggiani, J. H. Schwartz, F. Russo, S. Scuderi for the ASTRI project.
ICRC 2021 - 37th International Cosmic Ray Conference
12 - 23 July 2021, Berlin.
mattia.corpora@inaf.it

Introduction

The **ASTRI Mini-Array (MA)** is an INAF project to construct and operate an experiment for the study of gamma-ray sources emitting up to the TeV spectral band. The ASTRI MA consists of an array of nine Imaging Atmospheric Cherenkov Telescopes that will be deployed at the Observatorio del Teide (Tenerife, Spain). These telescopes will be an evolution of the two-mirror ASTRI-Horn telescope, successfully tested since 2014 at the Serra La Nave Astronomical Station of the INAF Observatory of Catania. Each telescope will be equipped with the new version of the ASTRI Silicon Photo-Multiplier (SiPM) Cherenkov Camera. The ASTRI-MA will be monitored and controlled by a Supervisory Control And Data Acquisition (SCADA) system which consists of different software subsystems. One of these is the **Cherenkov Camera Supervisor (CCS)**, that is the part of SCADA that interfaces with each Cherenkov Camera. The CCS provides the services to control and monitor the Camera through the Alma Common Software (ACS). This is a framework based on object-oriented CORBA middleware, which gives the infrastructure for the exchange of messages between distributed objects and system-wide-services. The CCS is based on the Open Platform Communications - Unified Architecture (OPC-UA) protocol, a platform-independent service-oriented architecture. This work presents the design and the technologies used by the ASTRI Camera team to implement the CCS. It describes architecture and functionalities starting from the definition of the use cases and the system requirements.

Mini-Array Software System Structure

To describe the CCS architecture, it is useful to explain some definitions and the software organization. A *system* is an arrangement of parts that together exhibit behaviour or meaning that the individual constituents do not. A subsystem is a system in its own right, except it normally will not provide a useful function on its own, it must be integrated with other subsystems (or systems) to make a system. The term assembly is used to indicate a hardware and software part of a subsystem. The term device is used to indicate a part of an assembly.
The ASTRI MA software is organized into 5 logical layers:

- **layer 0,** assembly: an assembly represents a collection of hardware (sensors, actuator);
- **layer 1,** Local Control Software (LCS): the system directly connected with the hardware and used to switch-on/switch-off, control, configure and get the status, monitoring points of all parts of the assembly;
- **layer 2,** OPC-UA interface with SCADA system;
- **layer 3,** control software: performs tasks of control and synchronization of the actions of the assemblies to accomplish the tasks. It must also manage the state machine of the assemblies, detect abnormal conditions or alarms from assemblies, and acquire monitoring points useful to detect events and the health of the assemblies.
- **layer 4:** Central Control System, is the layer that coordinates all the subsystems of the MA system.

SCADA shall switch-on, switch-off, control, configure, get status and manage the subsystem state machine and assembly state machine, acquire monitoring points that can generate warnings or critical event notifications, errors and alarms of the assemblies of the ASTRI Mini-Array Observing Subsystems. The software components belonging to layer 3 manage the subsystems. Among these, there is the Telescope Control System (TCS), which manages a single Telescope. The TCS shall be responsible for coordinating all telescope assemblies, starting up, configuring and shutting down the assemblies of the Telescope, supervising optical system control, telescope mount control and instrument control (Cherenkov Camera, Optical Camera,). The main components of the TCS shall be:

- **Cherenkov Camera Supervisor:** the software component that controls, monitors and provides an interface to the Camera LCS;
- **SI3 Supervisor:**the software component that provides an interface to the SI3 LCS;
- **Mount Supervisor**: the software component that controls, monitors the mount LCS and other auxiliaries;
- **Optics Supervisor:** the software component that controls and monitors the Optics LCS and the Optical Camera LCS;
- **Telescope Manager:** the software component responsible for coordinating all TCS subsystems and starting up and shutting down the system.

All Supervisor components interface with Connectors and Telescope Manager. A Connector is a low-level software component whose function is to communicate with LCS via OPC-UA to control devices. On the other hand, the TM is a software component that uses the business logic of the various supervisors and offers high-level services to the Central Control System. The Fig. 1 shows a generic component diagram which contains a supervisor, some connectors, the telescope manager and the relative interfaces.

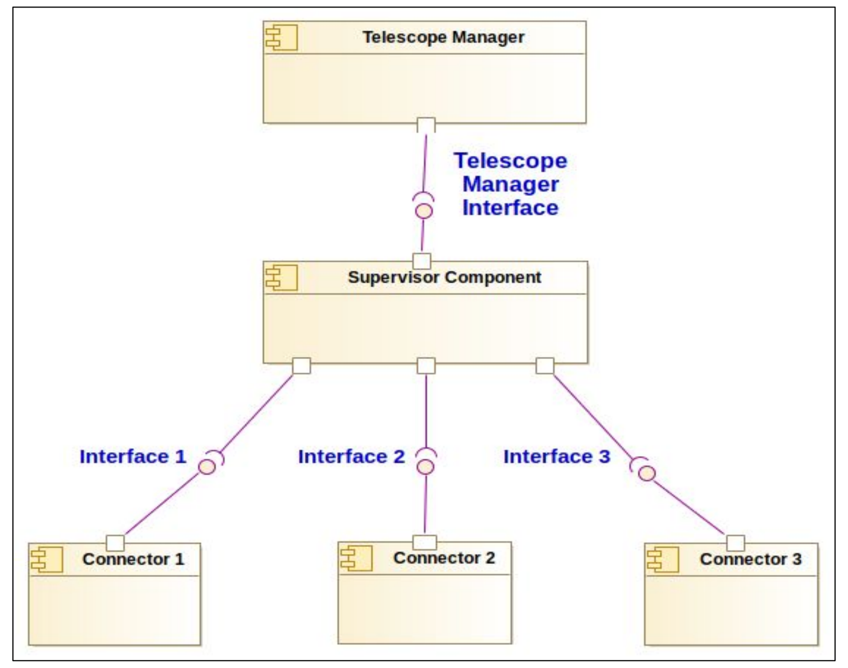

Figure 1: UML Component Diagram of Telescope Control System

Cherenkov Camera Supervisor Architecture

The CCS has two main features:

- **combines the services offered by Camera Connectors in order to perform high-level operations;**
- **provides an high-level interface to Telescope Manager**.

Fig.2 shows a context component diagram of the CCS. It interfaces with the following Connectors:

- Timing Connector: a component to manage the time synchronization device;
- VDB Connector: a component to monitor and control the voltage distribution device;
- LID Connector: a component to monitor and control the LID motion;
- TEC Connector: a component to monitor and control the thermal system;
- Focal Plane Connector: a component to manage the Camera Focal Plane;
- Calibration System Connector: a component to manage the Calibration System;
- Camera Environment Connector: a component to monitor the Camera environmental parameters, such as temperature and humidity.

The Connectors offer services for accessing OPC-UA nodes such as read/write an OPC-UA variable or call an OPC-UA method. They also provide a range based alarm system, ensuring that the values of the variables remain within the predetermined thresholds. The CCS offers some high-level services to Telescope Manager through the logic offered by ACS (based on CORBA). These services consist of the composition of primitive services of the lower layers. Moreover, the CCS exhibits the state machine (that is described below) of the Cherenkov Camera to the Telescope Manager. The latter can request or monitor the state transitions between those possible.

CCS implementation

The CCS shall be developed in Java Programming Language. It consists of some classes interacting with each other. In particular, a CCS main class instantiates objects of the controller classes. Each controller class manages a single device through the relative Connector. Each Connector is created using ACS (Alma Common Software) starting from a document that describes its functions, the so-called IDL (Interface Description Language). Through this document, the component skeleton is generated, which is then completed with the implementation part, i.e. the part of code that carries out the communication with the various nodes of the OPC-UA server.
Fig. 3 shows the CCS class diagram. The ***CameraSupervisorImpl* class** implements the ***CameraSupervisor* Interface** which contains the high-level operations (such as *performAcquisition*, *performCalibration* or *initCherenkovCamera*). The Telescope Manager shall use these ones according to the requests of the Central Control System. Each high-level operation corresponds to a sequence of calls to operations offered by the controllers. The latter will use the connectors services to communicate with the OPC-UA server, which will interact with the Cherenkov Camera LCS.

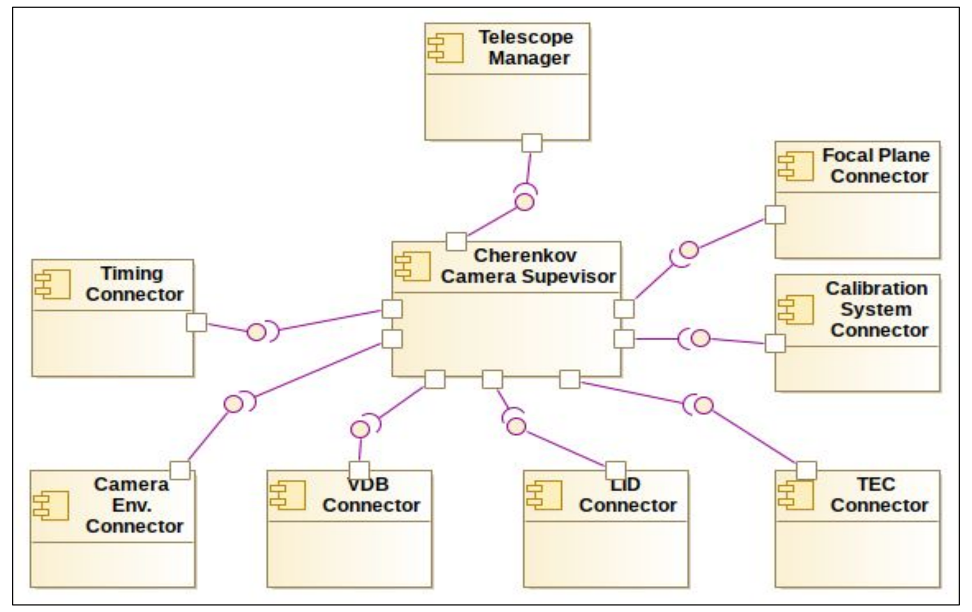

Figure 2: Cherenkov Camera Supervisor UML Component Diagram

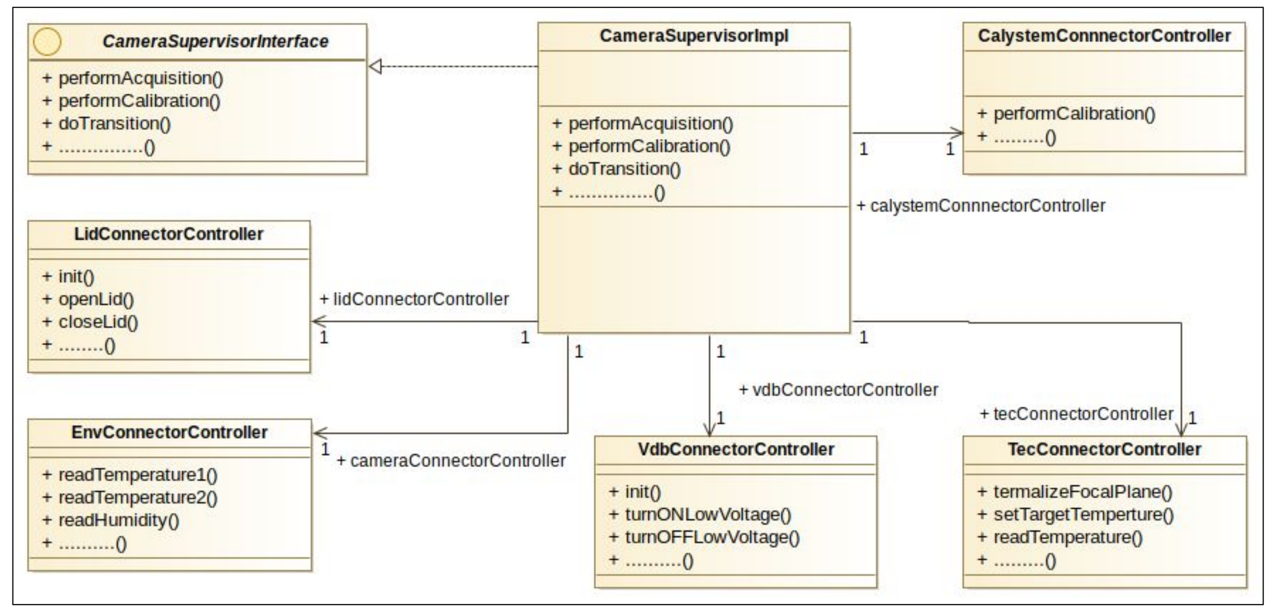

Figure 3: Cherenkov Camera Supervisor UML Class Diagram

In order to provide its state to the Telescope Manager, Cherenkov Camera Supervisor also must implement a common assembly state machine shown in Fig. 4 . Camera states are defined as follows:

- **Off State:** the Camera is entirely without electrical power.
- **On State:** with the following sub-states:
  - **Initialised State:** the state of the Camera after power on.
  - **Standby State:** a state that is still safe with respect to extreme conditions, but has all components activated, with preparations for observation initiated. The Camera is prepared for rapid entry into the Operational State.
  - **Operational State:** a state associated with operations (e.g. data taking), with configuration dictated by performance requirements. Two sub states could be present:
    - **Nominal:** the Camera can be operated with full performance;
    - **Degraded:** the Camera can be operated with reduced performance.
  - **Safe State:** if dangerous conditions are present, the Camera goes into a configuration where the object is considered exposed to "normal" risk for damage or loss.
  - **Fault State:** the Camera has encountered a serious problem, which means it is currently unable to meet the requirements associated with one of the standard states. Alarm shall be generated. The transition to Fault State shall generate an alarm signal for the Operator.
  - **Engineering Mode State**: a state designed to facilitate maintenance and engineering activities, and is unavailable for routine operations.

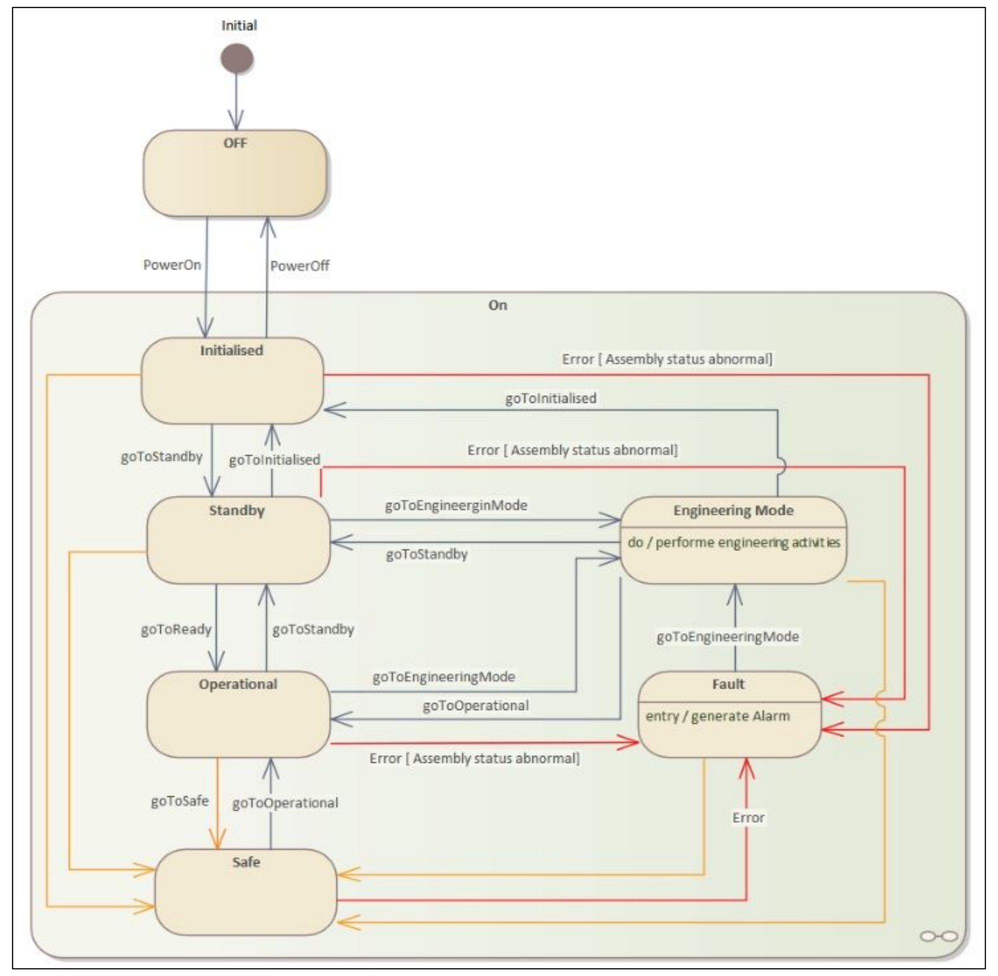

Figure 4: Cherenkov Camera Supervisor State Machine

The *CameraSupervisorImpl* class shall also provide some services regarding **logging** and **alarm management** to Telescope Manager. In particular, it must be able to raise alarms and warnings whenever a device or its relative controller faults. On the other hand, it must be able to listen to alarms raised by other assemblies or by the Central Control in order to execute the relative operations.

Conclusion

This design will be the basis for the software development of the Cherenkov Camera Supervisor in ASTRI Mini Array software. Even if the architecture is designed independently of the hardware that will be used, it could be changed and updated when the first prototype of the Cherenkov Camera will be built.
As the system was designed, it respects software engineering criteria, such as modularization, granularity and object orientation. This could favor future hardware and software integration, as well as changes to existing modules.