ICALEPCS 2021 October 18, 2021

Readout, Control and Monitoring for the Medium-Sized Telescopes in CTA

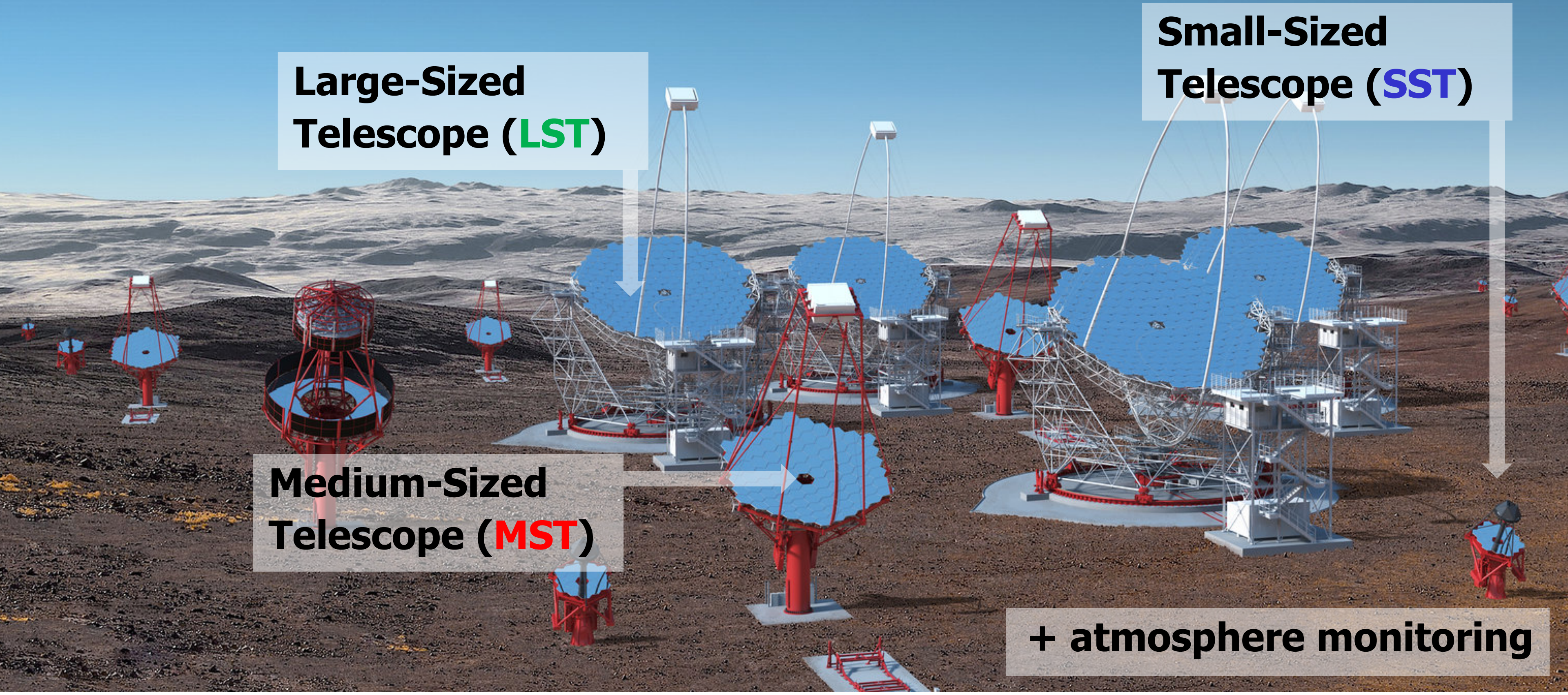
Ullrich Schwanke[1], T. Murach[2], P. Wagner[2], G. Spengler[1]
for the CTA MST Project, and
D. Melkumyan[2], I. Oya[3] and T. Schmidt[2]

[1]Humboldt University [2]DESY [3]CTA gGmbH

cherenkov telescope array

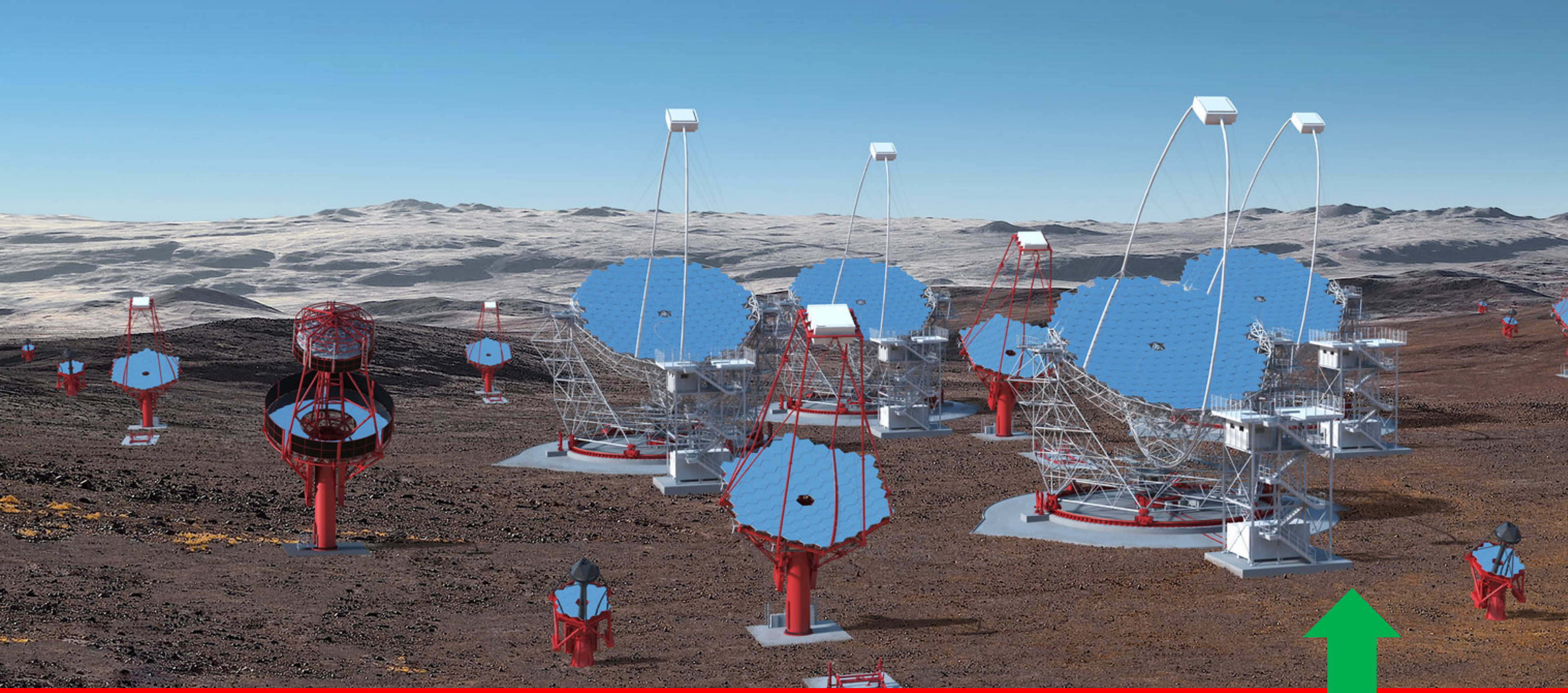
the observatory for ground-based gamma-ray astronomy

Cherenkov Telescope Array


Large-Sized Telescope (LST)
Small-Sized Telescope (SST)
Medium-Sized Telescope (MST)
+ atmosphere monitoring

α configuration		LSTs	MSTs	SSTs
CTA North	La Palma (Spain)	4	9	
CTA South	Paranal (Chile)	0	14	37

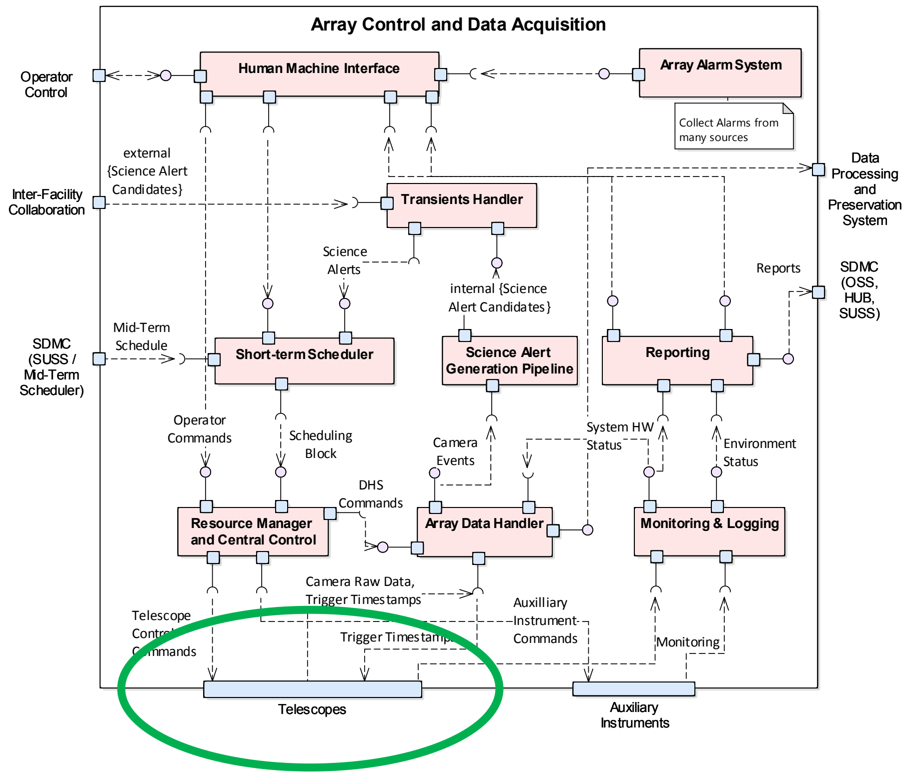
Array Control and Telescopes



Array Control and Data Acquisition System (ACADA)

- **concurrent operation of multiple telescope sub-arrays**
- **rapid re-scheduling in response to internal (real-time analysis) and external (other observatories) triggers**
- **different telescope types and atmospheric monitoring devices**

ACADA: Design and Implementation

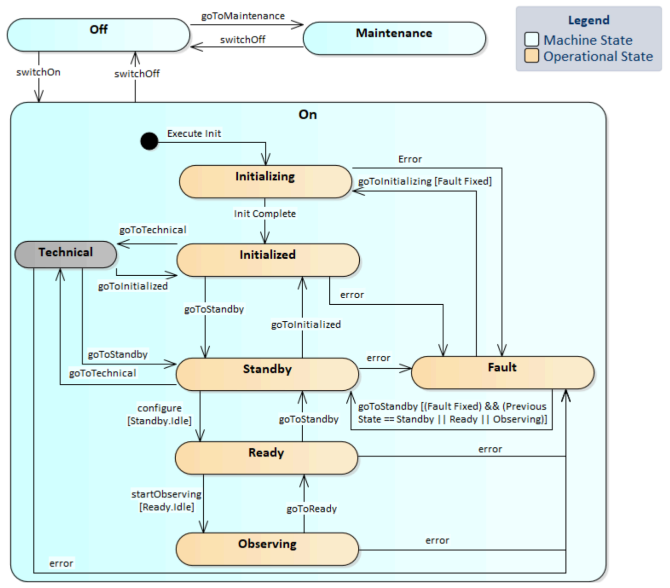


Array Control and Data Acquisition
Human Machine Interface
Array Alarm System
Collect Alarms from many sources
Operator Control
external {Science Alert Candidates}
Inter-Facility Collaboration
Transients Handler
Data Processing and Preservation System
Science Alerts
internal {Science Alert Candidates}
Reports
SDMC (OSS, HUB, SUSS)
SDMC (SUSS / Mid-Term Scheduler)
Mid-Term Schedule
Short-term Scheduler
Science Alert Generation Pipeline
Reporting
Operator Commands
Scheduling Block
Camera Events
System HW Status
Environment Status
DHS Commands
Resource Manager and Central Control
Array Data Handler
Monitoring & Logging
Camera Raw Data, Trigger Timestamps
Auxiliary Instrument Commands
Telescope Control Commands
Trigger Timestamps
Monitoring
Telescopes
Auxiliary Instruments

- **ACADA software is based on an architecture designed using the UML and SysML formalisms**
- **Implementation takes advantage of the Alma Common Software (ACS) framework**
- **OPCUA is the protocol of choice for communication with hardware devices**
- **Dedicated solutions (e.g. ZeroMQ) for bulk data transfer**

- **Will focus on one aspect here: the interface between ACADA and the telescopes (and its implementation for MSTs)**
- **Design principle: any telescope can be controlled by ACADA without knowing its exact type (LST, MST, SST)**

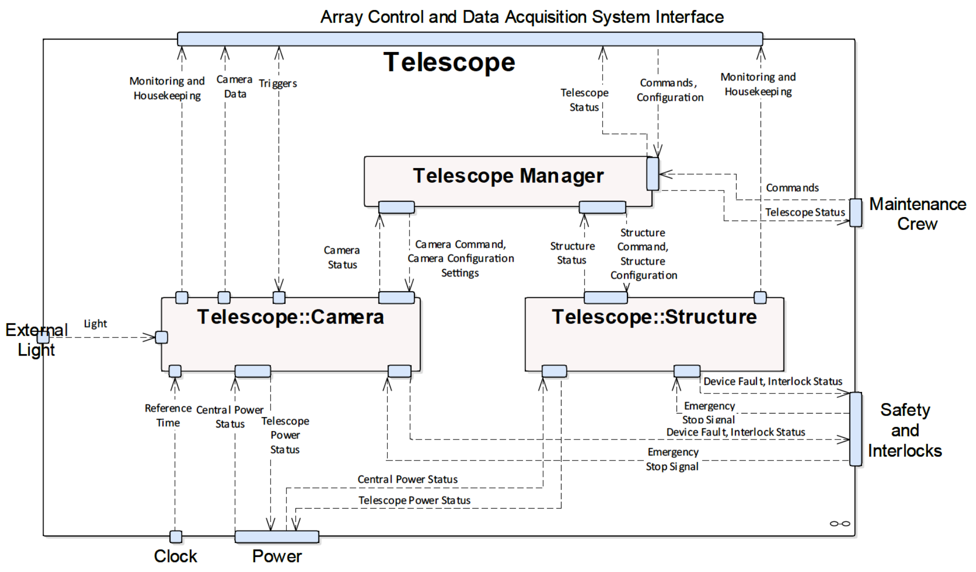
Telescope Interface (1/2)

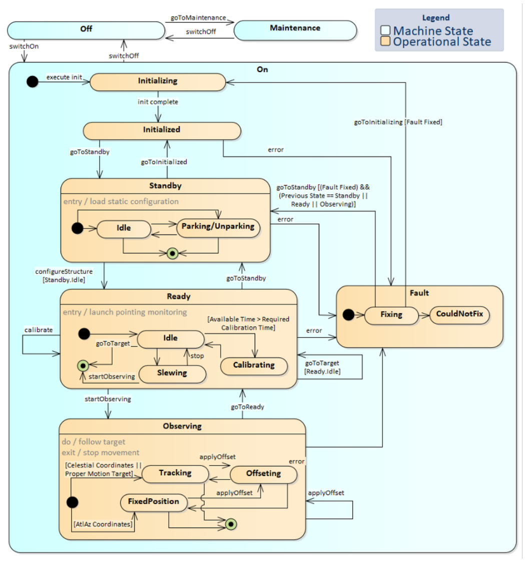


Transition	Description
goToInitialized goToStandby goToReady goToTechnical	Puts telescope to the Initialized/Standby/ Ready/Technical state
startObserving	Starts observation of a previously defined target
goToSkyTarget	Lets telescope track a celestial coordinate
goToProperMotionTarget	Lets telescope follow a target by providing a trajectory
goToFixedPosition	Points telescope to a fixed position

- **The generic telescope interface prescribes (i) a finite state machine, (ii) a set of high-level routines, and (iii) functions returning basic housekeeping information (e.g. telescope state, pointing position)**
- **Interface is defined/documented at the ACS level and uses ACS mechanisms (e.g. callbacks for asynchronous calls)**
- **Detailed hardware control is not intended here, but possible at a lower level**

Telescope Interface (2/2)





- **For standard observations, ACADA deals only with a single CORBA object (TelescopeManager) representing the entire functionality of the telescope**
- **For dedicated purposes (calibration, commissioning) the interface defines the functionality of the TelescopeCamera and the TelescopeStructure (all hardware except the Cherenkov camera)**
- **Telescope state is computed from camera and structure state**

MST Hardware

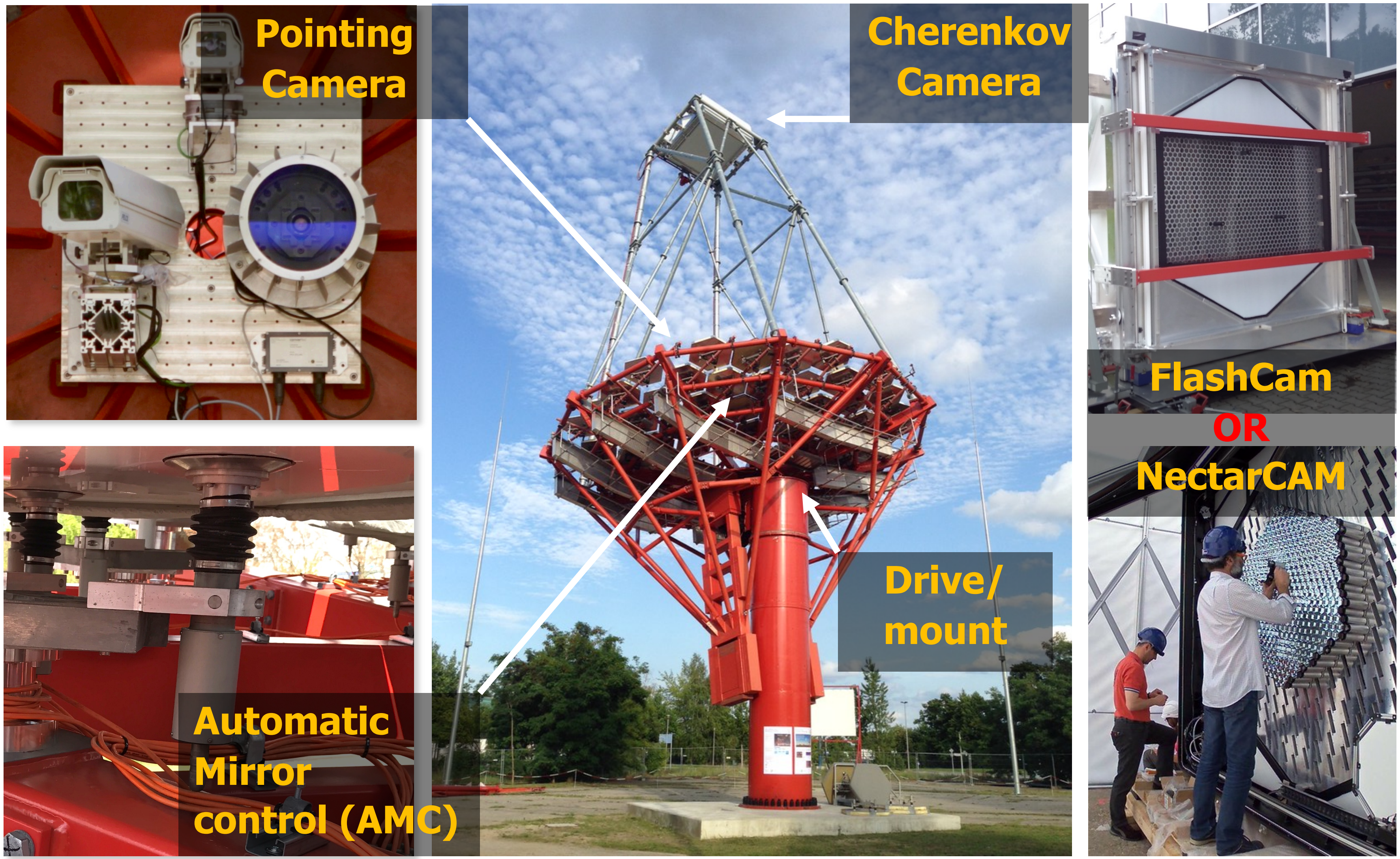
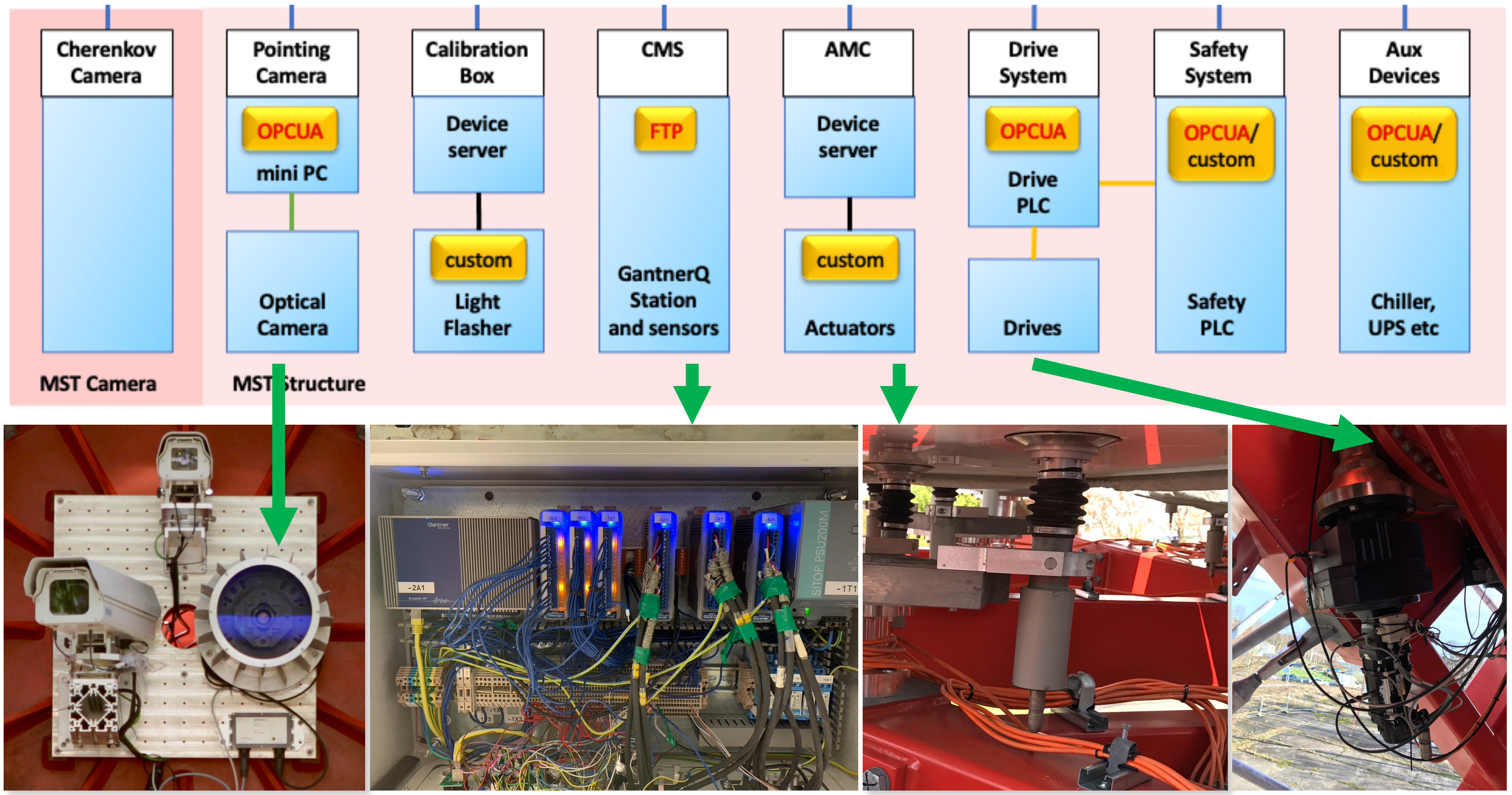


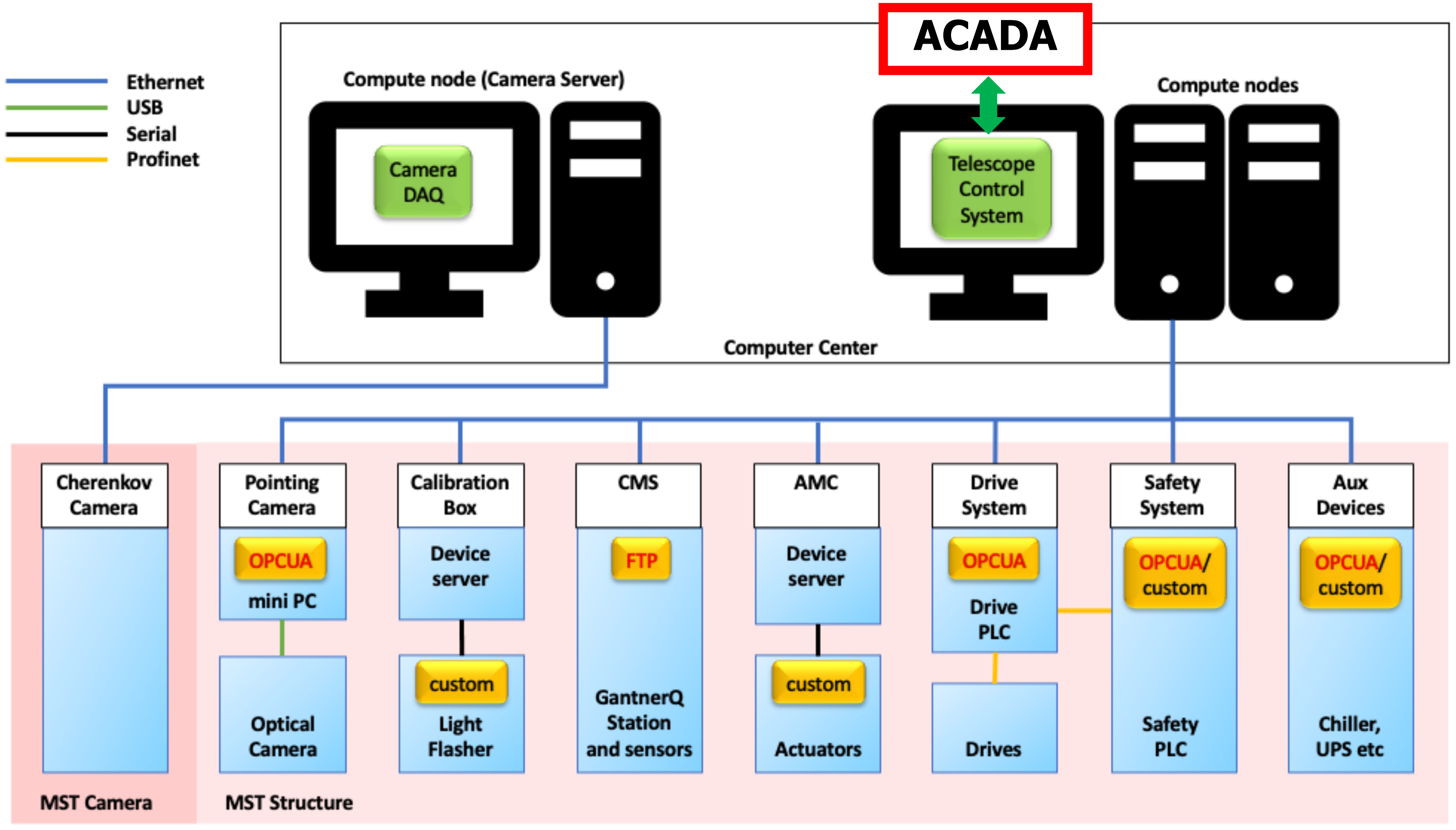
Image: MST Prototype (Berlin, 2012-2020)

MST (Structure) Subsystems

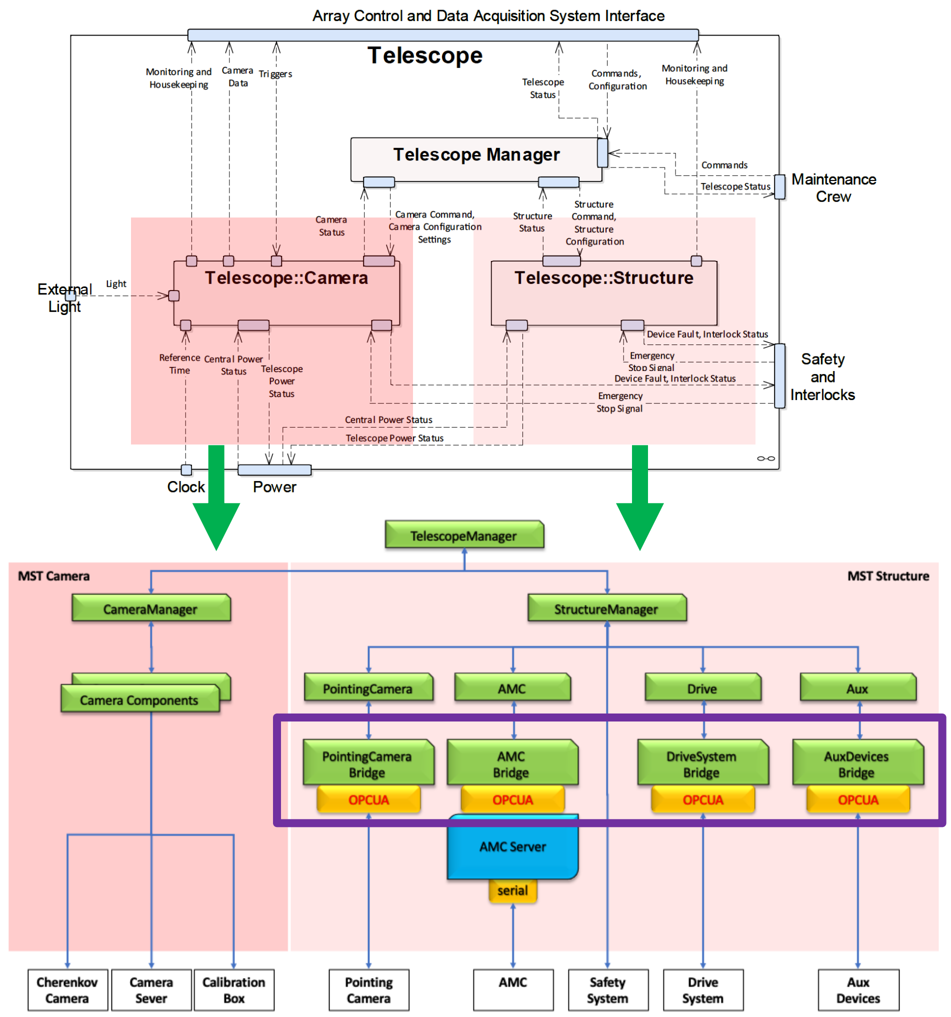
CMS = Conditions Monitoring System **(at the telescope)**



MST Control


Ethernet
USB
Serial
Profinet
ACADA
Compute node (Camera Server)
Camera DAQ
Telescope Control System
Compute nodes
Computer Center
Cherenkov Camera
Pointing Camera
OPCUA
mini PC
Optical Camera
Calibration Box
Device server
custom
Light Flasher
CMS
FTP
GantnerQ Station and sensors
AMC
Device server
custom
Actuators
Drive System
OPCUA
Drive PLC
Drives
Safety System
OPCUA/ custom
Safety PLC
Aux Devices
OPCUA/ custom
Chiller, UPS etc
MST Camera
MST Structure

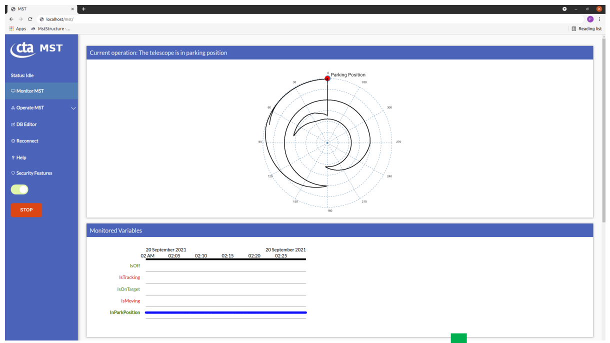
Telescope Control System (1/2)



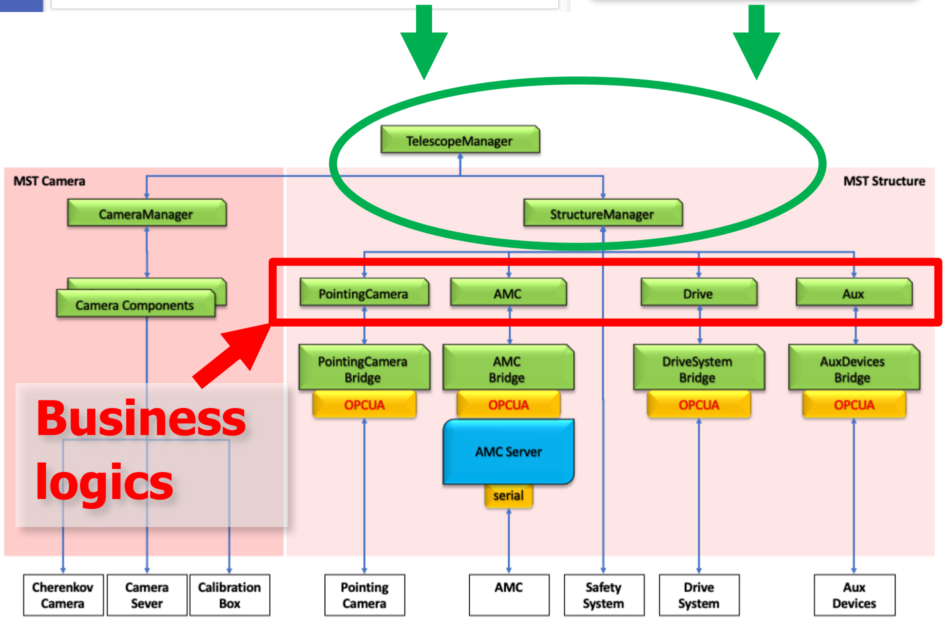
- A flat hierarchy of ACS components (in Java/python/ C++) colocated with ACADA in the array computer centre
- No supervision of components (unlike in ACADA)
- ACS configuration data base and encapsulation in TelescopeCamera is ideal for a telescope with two Cherenkov camera projects (FlashCam, NectarCAM)
- Use of an extended interface for the TelescopeStructure (also defined in IDL) for MST-structure-specific hardware
- OPCUA-to-ACS Bridges map ACS (properties) to OPCUA (nodes)

Telescope Control System (2/2)

Browser-based MST GUI



Expert scripts (e.g. mirror alignment)



- **The defined interfaces provide a clear entry point for GUIs and expert script**
- ***Technical State* decouples ACADA from telescope control**
- **Managers are concerned with state calculation and delegation of work**
- **Business logics implemented in few ACS components**
- **Example: Drive**
  - **Deals with astronomical coordinate transformations**
  - **Application of pointing model**
  - **Generation of track tables (t,az(t),el(t))**
  - **High-level safety (e.g. Sun avoid-ance)**

Summary and Outlook

- **The definition of a rather high-level interface between array control (ACADA) and the telescopes settled the division of labour between all involved parties**
- **Finite state machine and ACS-based interface are mandatory for all telescope projects, but telescope teams keep quite some freedom in the implementation**
- **For the MST (structure), most of the business logics is in the ACS layer (not in the OPCUA layer)**
- **Extended the ACADA-telescope interface to control MST-specific hardware items → clear prescription for the application of GUIs and scripting**
- **Looking forward to replace our development environment (mocks, CI system) with a real telescope in few years from now...**

- **Note: Have deliberately skipped over aspects of the monitoring. See talk by Alessandro Costa et al.**