Evaluating the control software for CTA in a medium size telescope prototype

2012 J. Phys.: Conf. Ser. 396 012037

(http://iopscience.iop.org/1742-6596/396/1/012037)

Evaluating the control software for CTA in a medium size telescope prototype

I Oya[1], B Behera[2], E Birsin[1], H Koeppel[2], D Melkumyan[2], T Schmidt[2], U Schwanke[1], P Wegner[2], S Wiesand[2] and M Winde[2] for the CTA Consortium[3]

[1] Institut für Physik, Humboldt-Universität zu Berlin, Newtonstrasse 15, D-12489 Berlin, Germany [2] Deutsches Elektronen-Synchotron, DESY, Platanenallee 6, D-15738 Zeuthen, Germany [3] see http://www.cta-observatory.org/ for full author and affiliation list

E-mail: oya@physik.hu-berlin.de

Abstract. CTA (Cherenkov Telescope Array) is one of the largest ground-based astronomy projects being pursued and will be the largest facility for ground-based γ-ray observations ever built. CTA will consist of two arrays (one in the Northern hemisphere and one in the Southern hemisphere) composed of telescopes of several sizes. A prototype for the Medium Size Telescope (MST) of a diameter of 12 m will be installed in Berlin by the end of 2012. This MST prototype will be composed of the mechanical structure, drive system and mirror facets mounted with powered actuators to enable active control. Five Charge-Coupled Device (CCD) cameras and a weather station will allow the measurement of the performance of the instrument. The Atacama Large Millimeter/submillimeter Array (ALMA) Common Software (ACS) distributed control framework is currently being considered by the CTA consortium to serve as the array control middleware. In order to evaluate the ACS software, it has been decided to implement an ACS-based readout and control system for the MST prototype. The design of the control software is following the concepts and tools under evaluation within the CTA consortium, like the use of a Unified Modeling Language (UML) based code generation framework for ACS component modeling, and the use of OPen Connectivity-Unified Architecture (OPC UA) for hardware access. In this contribution, the progress in the implementation of the control system for this CTA prototype telescope is described.

1. Introduction

Very High Energy (VHE, energies $>$ 100 GeV) γ-rays are produced in astronomical sources like pulsars, pulsar wind nebulae, supernova remnants and active galactic nuclei. Thanks to the success of Imaging Atmospheric Cherenkov Telescope (IACT) instruments like H.E.S.S. [1], MAGIC [2] and VERITAS [3] the field has experienced a major boost during the recent years, with more than 100 sources already established as VHE emitters. The technique used by IACTs is based on the fact that VHE γ-rays interact with the atmosphere producing electromagnetic showers, the charged components of which generate Cherenkov light in a narrow cone. An IACT consists of a reflector that concentrates the Cherenkov light in a camera composed of sensitive photo-detectors, usually photomultiplier tubes (PMTs).

In order to fulfill its scientific requirements, CTA will comprise two arrays (one in the Northern hemisphere and one in the Southern hemisphere), and more than 50 telescopes of three different sizes will be deployed in the arrays. A few large size telescopes (LST), with dishes of 23 m diameter, are designed to achieve a good sensitivity in the energy domain from a few tens of GeV up to about a hundred GeV. In order to achieve a good sensitivity at energies above a few TeV a large number of small-size telescopes (SSTs) will be deployed. For the core energies (around 100 GeV to 1 TeV) a considerable number of MSTs, of 10-12 m diameter each, will be used. By extrapolating the distribution of known VHE objects, it is expected that CTA will detect around 1000 new objects. Details on the science motivation, design concepts and expected performance of CTA are provided in [4].

2. The medium size telescope prototype

A prototype of one of the design concepts of the MST is under development [6] and will be deployed in Berlin by the end of 2012. The MST prototype has a Davies-Cotton [5] type reflector with diameter of 12 m, and a focal length of 16 m. The design of the telescope structure can be seen in Fig. 1. The main goal of this prototype is to test a design of the mechanical structure and drive system, but other prototype instruments like the Active Mirror Control (AMC) will also be tested. No real PMT camera will be installed on this prototype, but a dummy camera of 2.5 tons will resemble the structural effect of the real PMT camera, allowing functions such as the operation of the protective lids and temperature sensors to be tested. The prototype will also contain a drive system, five Charged-Coupled Device (CCD) cameras and a Weather Station (WS), plus some sensors designed to test the behavior of the structure. Emulated data sources will be employed to generate the expected data from PMT cameras of an array of IACTs. Time stamps of events will be obtained either via the Network Time Protocol (NTP) or a Global Position System (GPS) receiver.

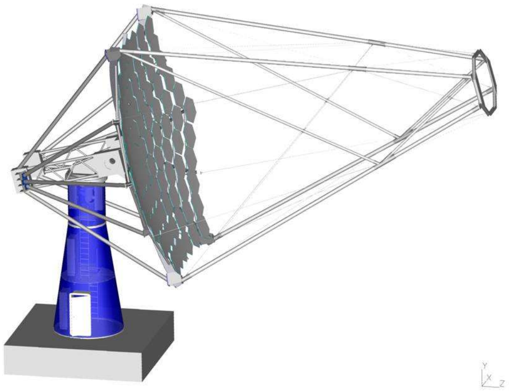

Figure 1. View of the design of the mechanical structure of the MST, composed of a quadrupod (camera support), a dish structure, a counterweight, the "head" (connecting the dish and the tower), and the tower with the foundation.

3. Control software for the MST prototype

The MST prototype constitutes a very good opportunity inside the CTA consortium to integrate a considerable number of hardware devices and to exercise the control software. Due to the similar level of complexity and requirements of ALMA [7] and CTA, the ALMA Control Software [8] (ACS) is currently being considered by the CTA consortium members to serve as the control middleware (see [9, 10] for more details). It has been decided to use ACS for controlling the MST prototype because it will provide an excellent platform to test the behavior of ACS with CTA hardware, while serving as a learning platform.

In order to standardize the interface to the hardware, a lower level software layer provided by the OPC UA is under evaluation by the CTA consortium. Some preliminary studies and developments aiming at the illustration of the OPC UA standard have taken place at LAPP[1] in order to illustrate the implementation of the OPC UA standard [10]. In particular these implementations concerned some electronic components and related functionalities, devoted to the online security control of the H.E.S.S. 2 camera [11]. A similar approach has been adopted for the MST prototype: OPC UA servers for hardware access are included in the drive system, CCD cameras and weather station. A standard ACS access method (a Java DevIO class) has been developed (see [10] for details) with the support of the ACS development team from ESO[2], complementing the existing C++ DevIO for OPC UA [12]. In the context of the CTA MST prototype, the OPC UA servers are being modeled using the *Prosys* OPC UA Java Software Development Kit (SDK).

A software development, integration and test environment for the MST control software exists in DESY Zeuthen, comprising two Dell PowerEdge R510 machines with 12 fast disks, 12 Nehalem Cores, 36 GB RAM memory and 10 Gbit Ethernet interface using the same layer-2 switch. An RPM package manager based installation of 64-bit ACS releases has been developed. An archive for permanent storage of the prototype measurements is under development, which will also serve as a *test bench* for the CTA archive. Different technologies for archiving data, as well as their integration in the control middleware, are being tested. In particular, the use of relational databases (MySql) and document-oriented databases (e.g. MongoDB) are under consideration. The DESY data-center will be connected to the MST prototype site, ca. 15 km away, by a broadband connection with a bandwidth of about 40 MByte/s. To exercise the transport of large data volumes, different technologies like the Data Distribution Service (DDS), Common Object Request Broker Architecture (CORBA) mechanisms, Zero Message Queueing (ØMQ) and other ones are being evaluated by the CTA consortium (details are provided in [10]).

To facilitate the creation of ACS components of the MST prototype, the *Comodo* framework [13], based on UML code, is being tested. The UML models are created using MagicDraw UML Standard [14] (v.17.0 or higher) on Linux, which is used to create the input required by *Comodo*. From the UML model, the interface definition language (IDL), configuration database files and Java implementation skeleton are created. The Java skeleton can be extended, for example to provide an OPC UA client functionality, in order to communicate with an OPC UA server.

4. Prototype instrumentation

4.1. Drive system

The drive system of the MST prototype is designed to resemble the expected operation modes of the CTA telescopes, allowing pointing of the prototype to any position and to track any astronomical object. The telescope will operate with two motors, one for azimuth and one for elevation.

The drive system of the prototype is composed, at a lower level, of the control of 6 drives (2 for azimuth and 4 for elevation) communicating via a *Bosch-Rexrot* programmable logic controller (PLC). The PLC operates VxWorks real-time operating system and hosts an OPC UA server. This OPC UA server can be accessed from an ACS component (currently under development) by using the OPC UA DevIO mechanism mentioned before.

As a preliminary step to operate the drive system with the prototype, a testing program is being carried out by using two drive test stands (see Fig. 2), allowing the evaluation of the

[1] Laboratoire d'Annecy-le-Vieux de Physique de Particules http://lapp.in2p3.fr/

[2] European Southern Observatory http://www.eso.org/

drive concept, different algorithms to operate with two drives in synchronous operation and exercising the safety aspects. After the prototype is deployed, a first phase of commissioning will take place, when the control software will be used to tune the system by using some fiducial marks. In a later phase, the control system will be used to track astronomical objects and to check the pointing of the telescope by taking images of the field of view with a CCD camera and using the *astronomy.net* software to find the actual targeted position. The final goal is to check if the mechanical tracking accuracy (without any pointing model correction) is better than of 1.2 arcmin (corresponding to the CTA specifications), and to estimate the post-pointing-model systematic error, that should be lower than ∼10 arcsec.

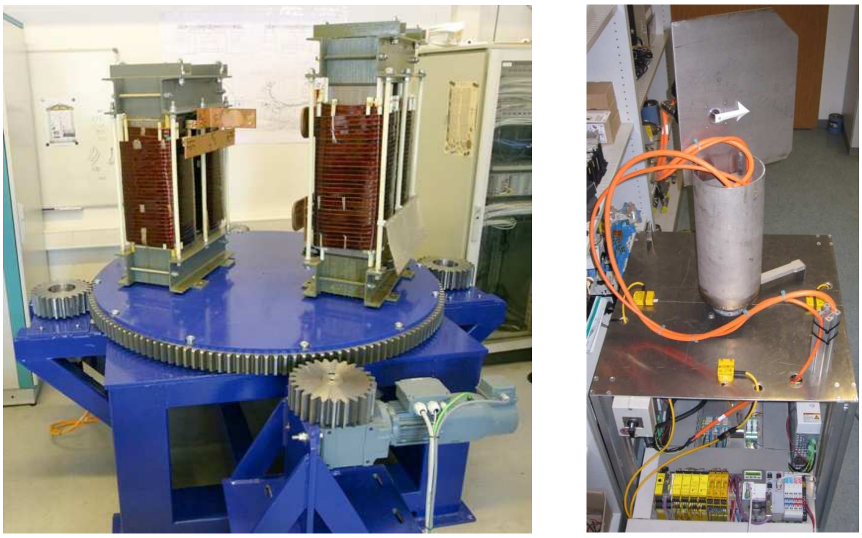

Figure 2. The two MST prototype drive test stands that allow different aspects of the drive system to be evaluated.

4.2. The CCD cameras

A crucial tool to check different aspects of the MST prototype will be a set of five CCD cameras. These cameras will take pictures of Light-Emitting Diodes (LEDs) located in the dummy camera, fiducial marks located near the telescope and astronomical objects. One goal of the CCD cameras will be to test the structural design of the prototype and, with the help of a WS, the effect of the temperature, wind and other environmental factors. Additionally, the CCDs will be used to perform mirror adjustments, and will allow for measurements of the optical point spread function. The output images will be stored in the standard astronomical Flexible Image Transport System (FITS) data format for later image processing. These CCDs will operate at a rate up to 10 Hz and, excluding the emulated data sources, they will generate the largest fraction of data volume of the prototype.

The chosen CCD camera model is a Prosilica GC 1350 which has a resolution of 1.4 Mpix (1360 x 1024) and a pixel size of 4.65 μm. These CCD cameras are interfaced via GigE Vision interface (allowing up to 1000 Mbit/s on Gbit Ethernet). The Allied Vision Technologies (AVT) PvAPI SDK allows to control and capture images from GigE Vision CCD cameras in a Linux environment. It is accessible by most programming languages such as C++ and Java (via JNI). The evaluation of this software has been positive, working correctly in the test machines in both implementation languages.

Fig. 3 shows a schematic view of the CCD camera software design. OPC UA servers will be used as the low level access to the CCD cameras. In the current stage, a preliminary server

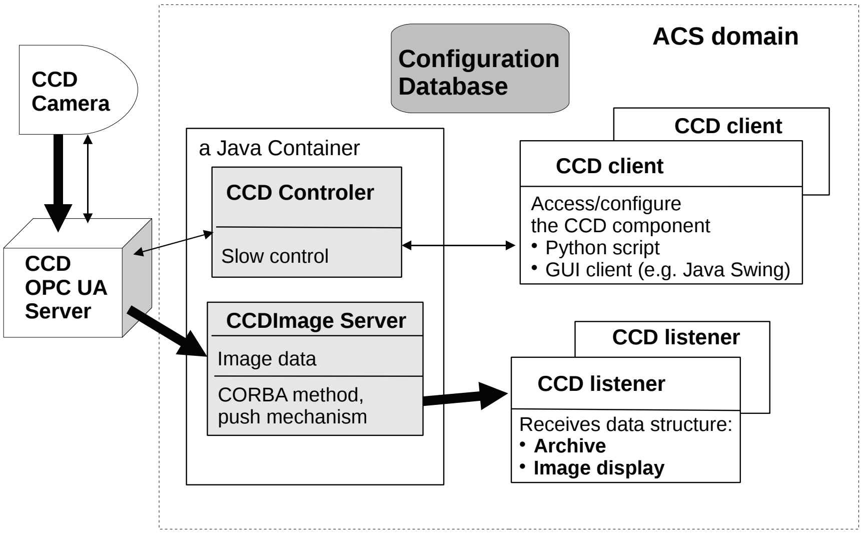

Figure 3. Schematic view of the design of the CCD control and acquisition software. The thick arrows represent the image data flow whereas the thin arrows represent slow control and monitoring information flow.

has been used to interact with a CCD unit. In parallel, an OPC UA server that simulates the CCD operation by generating frames and by reacting to slow control commands has been used to check the interaction with ACS. On a higher level, the ACS part of the controlling software is being created by using a Java implementation of some ACS components:

- **CCD controller.** It serves to interface to properties and methods on the OPC UA Server by using the Java DevIO mechanism. The ACS configuration database is used to define the default values and parameter limits of each CCD unit.
- **CCD image server.** It obtains the CCD images from the OPC UA server by using the subscription mechanism of the Java DevIO. By separating the image server from the controller, it will be possible to test different data transport mechanisms with the CCD frames in the distributed system. In the existing implementation and as a preliminary solution, a CORBA method is used to transfer a data structure via a push method to a data listener (data writers and image displays).
- **Image file writers.** Simple ACS components that write the image data to disk in different image file formats have been created. So far two components were created, one saving image data in FITS format by using the `nom.tam.fits` library and the other that can write to disk with usual image file formats (e.g. .png). For the final implementation, a solution to write the meta-data and links to the CCD images into a document-oriented database (e.g. MongoDB) is being developed.
- **Image display.** A graphical user interface to display the received frames has been created.
- **Python ACS scripts.** Used to activate and test the interaction of the different components and the simultaneous operation of several CCD units.

A system containing the simulation OPC UA server and the described ACS components and clients, including the image display, has been successfully set.

4.3. The active mirror control

The design of the CTA telescopes makes use of a tessellated reflector composed of individual mirror facets (see [15] for details). Each individual mirror facet will be attached to a triangular support, with two power actuators and a fixed support (an Active Mirror Control (AMC) unit) which has the functionality to enable perfect mirror alignment. This will allow online re-alignments of the LSTs reflectors, where the deformation caused by the weight of the telescope will cause misalignments depending on the telescope elevation. It will also allow, for larger timescales, realignments in MSTs and SSTs to be performed. The dish of the MST prototype will be completely covered by a combination of real and dummy mirrors. Several AMC units of two different designs will be installed in the MST prototype (see [4] for details in the design concepts). One type of unit communicates via XBee radio modules [16], creating a Wireless Personal Area Network (WPAN) that is accessed via a XBee receiver connected to a PC via USB or RS-232 serial interface. The other type of unit is interfaced via Controller Area Network Bus (CAN-Bus), accessed via an Ethernet-CAN-Bus gateway.

In order to simplify the control of the AMC system and the mirror alignment procedures, the higher level interfacing to both unit types will be unified. This can be achieved with a higher level ACS component which sends the common instructions for the alignment procedures, acting as client for the lower level ACS components that will be different for each of the two AMC concepts. The mirror alignment procedures use as input the images obtained with the CCD cameras described above, and therefore the higher level component will be synchronized with the operation of the CCDs.

For the AMC units using the XBee communication, a preliminary version of a C++ ACS component using the *libxbee* library has been implemented. The AMC hardware has been successfully tested using C++ under ACS.

4.4. The Weather Station

A WS will be installed nearby the MST prototype to continuously monitor the weather parameters, allowing correlation with the behavior of the structure of the MST with changes in the environmental status like, for example, the wind speed or temperature.

The chosen WS model is a *Davis Vantage VUE* [17]. This WS is able to measure the wind speed and direction, as well as other quantities with the required accuracy and measurement rate. The instrument is composed of an outdoor unit communicating via a WPAN with an indoor unit, including a data-logger with some limited internal storage capacity and equipped with a RS-232 serial interface. The WS is temporally installed in DESY Zeuthen, allowing to check the developed software.

Since the WS is the least complex component of the MST prototype, it was the first device used to experiment the software development procedures. At the lowest software level, an OPC UA server has been implemented with the OPC UA Java SDK which uses the RXTX library to communicate with the WS *datalogger* via the serial line. At a higher level, the data from the WS is accessed via a Java ACS component. With the actual implementation, the data are stored on disk by using an ACS Python client. A general mechanism to insert slow control data from the MST prototype into a MySql relational database (using Django and/or Hibernate open source frameworks) is under development and is being investigated first with the WS.

5. Conclusions

The MST prototype is providing a realistic *test bench* scenario for the control software for the CTA array. Equipped with complex systems like the drive system, CCD cameras and the AMC, the MST prototype will allow the evaluation of the ACS control middleware while the hardware design concepts are being tested, providing input and expertise to the CTA array control software developers. The use of tools like the UML code generator, OPC UA servers and different transport mechanism like DDS, will eventually allow the creation of an efficient and reliable framework for the CTA array control.

Acknowledgments

We thank the ACS experts from ESO Garching, in particular G. Chiozzi and H. Sommer, for their excellent support. We gratefully acknowledge support from the agencies and organisations listed at the following URL: http://www.cta-observatory.org/?q=node/22

References

[1] Aharonian F et al (H.E.S.S. Collab.) 2006 Observations of the Crab nebula with HESS *A&A* **457** 899

[2] Albert J et al (MAGIC Collab.) 2008 VHE γ-Ray Observation of the Crab Nebula and its Pulsar with the MAGIC Telescope *ApJ* **674** 1037

[3] Acciari V A et al (VERITAS Collab.) 2008 VERITAS Observations of the γ-Ray Binary LS I +61 303 *ApJ* **679** 1427

[4] CTA Consortium 2011 Design concepts for the Cherenkov Telescope Array *Experimental Astronomy* **32** 193

[5] Davies J M and Cotton E S 1957 Design of the quartermaster solar furnace *J. Solar Energy Sci. and Eng.* **1** 16

[6] Davis E et al 2011 Mechanical design of a medium-size telescope for CTA *CTA internal note*

[7] Wootten A and Thomson A R 2009 The Atacama Large Millimeter/submillimeter Array *IEEE Proc.* **97** 1463

[8] Chiozzi G et al 2004 The ALMA Common software: a developer friendly CORBA based framework *SPIE Proc.* **5496** 205

[9] Oya I et al 2011 A readout and control system for a CTA prototype telescope *Proc. of ICALEPCS 2011* Grenoble, France, October 2011

[10] Wegner P et al 2012 Simultaneous Operation and Control of about 100 Telescopes for the Cherenkov Telescope Array *these proceedings*

[11] H.E.S.S. webpage http://www.mpi-hd.mpg.de/hfm/HESS/

[12] Di Marcantonio P et al 2011 Evaluation of software and electronics technologies for the control of the E-ELT instruments: a case study *Proc. of ICALEPCS 2011* Grenoble, France, October 2011

[13] Troncoso N et al 2010 A code generation framework for the ALMA common software, *SPIE Proc.* **7740** 121

[14] MagicDraw UML http://www.nomagic.com/products/magicdraw.html

[15] Föster A et al 2011 Mirror development for CTA *Proc. of the 32nd International Cosmic Ray Conference* Beijing, China, 2011

[16] XBee from Digi International http://www.digi.com/

[17] Davies Vantage Vue weather station http://www.vantagevue.com/