Status of the array control and data acquisition system for the Cherenkov Telescope Array

Füßling, Matthias, Oya, Igor, Balzer, Arnim, Berge, David, Borkowski, Jerzy, et al.

Matthias Füßling, Igor Oya, Arnim Balzer, David Berge, Jerzy Borkowski, Vito Conforti, Josep Colomé, Rico Lindemann, Etienne Lyard, David Melkumyan, Michael Punch, Ullrich Schwanke, Joseph Schwarz, Claudio Tanci, Gino Tosti, Peter Wegner, Ralf Wischnewski, Amanda Weinstein, "Status of the array control and data acquisition system for the Cherenkov Telescope Array," Proc. SPIE 9913, Software and Cyberinfrastructure for Astronomy IV, 99133C (8 August 2016); doi: 10.1117/12.2233174

SPIE.

Event: SPIE Astronomical Telescopes + Instrumentation, 2016, Edinburgh, United Kingdom

Status of the array control and data acquisition system for the Cherenkov Telescope Array

Matthias Füßling[a], Igor Oya[a], Arnim Balzer[b], David Berge[b], Jerzy Borkowski[c], Vito Conforti[d], Josep Colomé[e], Rico Lindemann[a], Etienne Lyard[f], David Melkumyan[a], Michael Punch[g], Ullrich Schwanke[h], Joseph Schwarz[i], Claudio Tanci[j], Gino Tosti[j], Peter Wegner[a], Ralf Wischnewski[a], Amanda Weinstein[k], and the CTA consortium[l]

[a]DESY, Zeuthen, Germany
[b]GRAPPA - Anton Pannekoek Institute for Astronomy, Netherlands
[c]N. Copernicus Astronomical Center, Poland
[d]I.A.S.F. di Bologna, Italy
[e]IEEC-CSIC, Spain
[f]University of Geneva - Departement dAstronomie, Switzerland
[g]APC, Paris
[h]Humboldt-Universität zu Berlin, Germany
[i]INAF - Osservatorio Astronomico di Brera, Italy
[j]University of Perugia, Italy
[k]Iowa State University, USA
[l]Full consortium author list at http://cta-observatory.org

ABSTRACT

The Cherenkov Telescope Array (CTA) will be the next-generation ground-based observatory using the atmospheric Cherenkov technique. The CTA instrument will allow researchers to explore the gamma-ray sky in the energy range from 20 GeV to 300 TeV. CTA will comprise two arrays of telescopes, one with about 100 telescopes in the Southern hemisphere and another smaller array of telescopes in the North. CTA poses novel challenges in the field of ground-based Cherenkov astronomy, due to the demands of operating an observatory composed of a large and distributed system with the needed robustness and reliability that characterize an observatory. The array control and data acquisition system of CTA (ACTL) provides the means to control, readout and monitor the telescopes and equipment of the CTA arrays. The ACTL system must be flexible and reliable enough to permit the simultaneous and automatic control of multiple sub-arrays of telescopes with a minimum effort of the personnel on-site. In addition, the system must be able to react to external factors such as changing weather conditions and loss of telescopes and, on short timescales, to incoming scientific alerts from time-critical transient phenomena. The ACTL system provides the means to time-stamp, readout, filter and store the scientific data at aggregated rates of a few GB/s. Monitoring information from tens of thousands of hardware elements need to be channeled to high performance database systems and will be used to identify potential problems in the instrumentation. This contribution provides an overview of the ACTL system and a status report of the ACTL project within CTA.

Keywords: Cherenkov Telescope Array, CTA, Central Array Control Software, Data Acquisition Software, Telescope Control Software, γ-Ray Astronomy, Cherenkov Telescopes, Alma Common Software (ACS), OPC UA

Further author information: (Send correspondence to M.F. and I.O.)
M.F.: E-mail: matthias.fuessling@desy.de, Telephone: +49 337 627 7457
I.O.: E-mail: igor.oya.vallejo@desy.de, Telephone: +49 337 627 7226

Software and Cyberinfrastructure for Astronomy IV, edited by Gianluca Chiozzi, Juan C. Guzman, Proc. of SPIE Vol. 9913, 99133C · © 2016 SPIE
CCC code: 0277-786X/16/$18 · doi: 10.1117/12.2233174

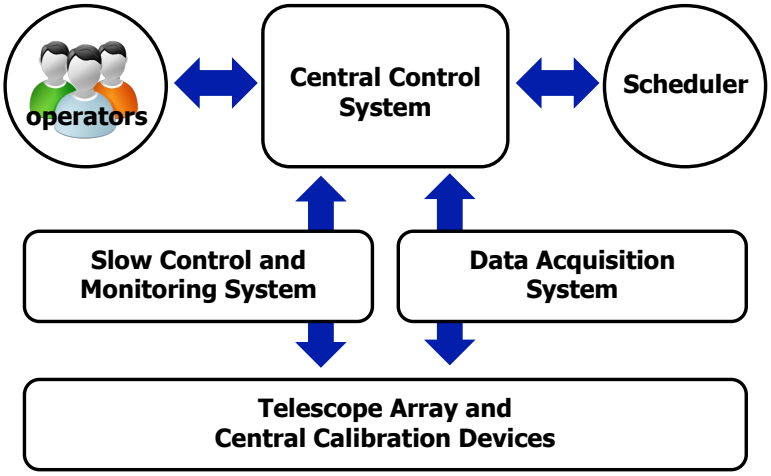


Figure 1: Schematic view of the main building blocks of the array control and data acquisition (ACTL) software system for the Cherenkov Telescope Array (CTA).

1. INTRODUCTION

The Cherenkov Telescope Array (CTA)[1] is planned as the next generation ground-based very-high-energy gamma-ray observatory. Currently considered array designs deploy about O(100) (O(20)) telescopes on an area of roughly $10\,\mathrm{km}^2$ ($1\,\mathrm{km}^2$) on a Southern (Northern) site. The increased complexity in operation, control and monitoring of a large distributed multi-telescope array with high reliability leads to new challenges in designing and developing the CTA array control and data acquisition (ACTL) software system. The ACTL software system (see Fig. 1 for the main building blocks) provides the software that is necessary to monitor and control all telescopes and auxiliary devices in the CTA arrays, to schedule and perform observations and calibration procedures, and to time-stamp, read-out, filter and store data.

In contrast to current-generation experiments CTA will be an open astronomical observatory. A significant fraction of the total available dark time (about 1300 h per year) will be filled with proposal-driven observations and all observations should be performed in a largely automatic fashion under the control of a very few professional operators. The ACTL system must support the optimal use of the observation time, diverse operation modes (surveys, pointed observations, multi-wavelength campaigns), the generation of outgoing alerts (by means of a pseudo real-time analysis of CTA data[2]), the prompt and proper reaction to alerts arriving from other observatories or to targets of opportunity, and the parallel operation of sub-arrays, groups of CTA telescopes operating independently of each other, by providing an automatic scheduling system. A central control system implements the execution of observations under automatic or local control (human operators via user interfaces). In addition, it handles the management of resources and provides mechanisms for the configuration and monitoring of all ACTL sub-systems.

The CTA observatory will be composed of Cherenkov telescopes of three different sizes (with typical reflector diameters of about 23 m, 12 m, and 4 m, respectively) whose cameras will comprise 1,000 to 10,000 pixels. The data rates of CTA that the ACTL system has to cope with are dominated by the acquisition of Cherenkov images selected by the local camera triggers during observations. The single telescopes are planned to be read out at single telescope trigger rates of about 0.5 kHz to 10 kHz. Along the way, ACTL provides array-level trigger schemes to suppress the background while selecting electromagnetic showers with high efficiency. For triggering and later association the ACTL system needs to provide a mechanism to timestamp data from the various telescopes with ns precision in an array whose extent implies signal round-trip times exceeding 10 µs. The ACTL data acquisition system has to be able to transport, process and store the resulting total data rates of about 30 GB/s (15 GB/s) for the Southern (Northern) CTA array.

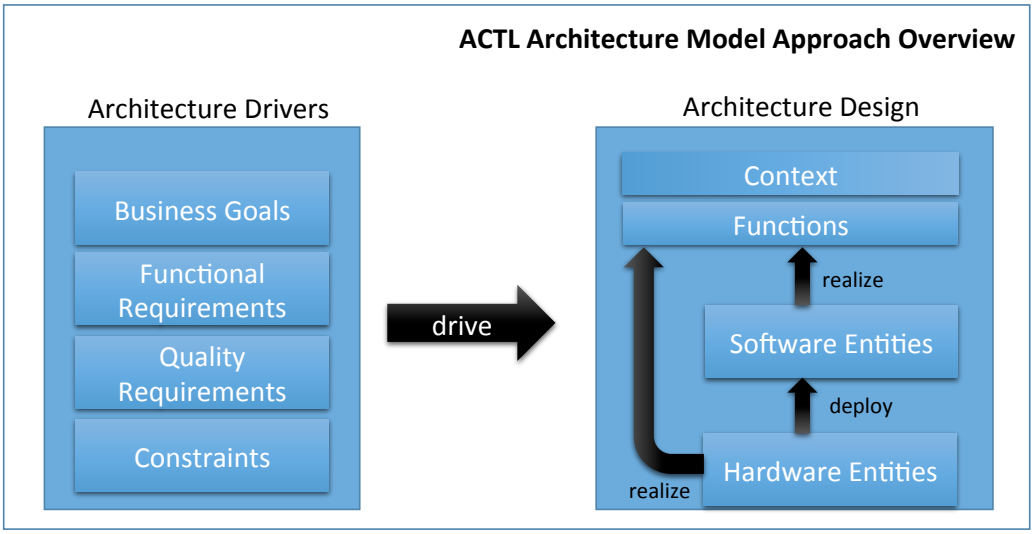


Figure 2: The architecture approach for the ACTL system[3] together with the architecture drivers and model viewpoints as well as the related design steps.

Besides the camera, telescopes comprise further systems (drive systems, CCD cameras, sensors, LED flashers, mirror control units etc.) that are needed for the operation and calibration of each Cherenkov telescope. In addition there are devices at the array level (trigger, LIDARs, weather stations, optical telescopes etc.) that must be operated in connection with the Cherenkov telescopes. The control and readout of these devices is another important ACTL task.

To build such a complex system, well-defined procedures supporting software development and project management are crucial. While prototyping activities play a major role in testing the feasibility of key concepts, a well-defined approach towards a coherent architecture model for the entire ACTL project forms the basis for all software development and project management activities.

In this proceeding, an overview of the architecture process and software frameworks is given in Sections 2 and 3, respectively, followed by an overview of the main building blocks of the ACTL system in the Sections 4 to 10.

2. ARCHITECTURE MODEL AND APPROACH

We have adopted a formal architectural approach for the entire ACTL project,[3] an approach appropriate for the high complexity of functionality and size of the project. The architectural approach is based on the SPES 2020 methodology for model-based engineering of embedded systems,[4] using a tailored SysML formalism[5,6] with tool support[7] to systematically describe the ACTL system and its relationship with the architecture drivers. The architecture drivers for the ACTL system are largely based on use cases motivated by the observational procedures expected from the CTA arrays and quality requirements (e.g. array up-time). The derived ACTL system model and architecture enables us to follow a model-driven software development and systems engineering approach and is the basis for addressing a large range of goals, both project-related (e.g. project setup and organization, effort estimation, project schedule) and software-related (e.g. design specification, developer guidelines, verification, validation and acceptance of software products). Throughout the lifetime of the project, the model will help to establish a common vision, which can be communicated to and is shared by all stakeholders. A decision-making process with reasoning on design and its impact on the ACTL software products can be established based on a coherent and common system model.

Figure 2 shows a schematic view of the design approach and the different viewpoints of the model. The design approach starts from customer requirements (functional requirements, quality requirements, use cases, business

goals) and, taking into account any constraints from e.g. chosen software frameworks (see Sec. 3), develops a fully traceable model. In an integrated approach, different stakeholders and different engineering techniques are reflected in a model with different views (context view, functional view, logical view and technical view), which focus on different aspects of the system to be developed (context, functionality, software and hardware). The dependency between the different views is fully modeled and documented, so that any refinement of one aspect of the model during the design phase is reflected in the overall model.

The software architecture is the key ingredient for the implementation of the ACTL software and is given priority in the project. As a reference for the entire project, the ACTL system model will be constantly refined throughout all project phases. The architectural artifacts and documentation will then serve as a basis for the refinement for the project planning and organization and will grow as the project evolves.

3. SOFTWARE FRAMEWORKS

The nature of CTA as a worldwide collaboration of universities and research centres, with different software development cultures, implies that the creation of the ACTL software can become a complicated task. A way to mitigate these complications from the beginning can be achieved by the selection of common software frameworks that promote merging the disparate styles and approaches, provide standard services and enforce standards and guidelines among the software developers. Two software frameworks have been chosen as the basis for the ACTL software: the ALMA Common Software[8] (ACS) as a higher-level software framework for supporting the application layer and OPC UA as lower-level software framework for standardised hardware access.[9]

3.1 ACS

The high-level ACTL software implements the software architecture using ACS, a software framework for the implementation of distributed data acquisition and control systems. ACS was developed as a common software framework for the Atacama Large Millimeter/sub-millimeter Array[10] (ALMA), which is a joint project of a worldwide collaboration and an installation with many similarities in operation and complexity to those of CTA. ACS is a distributed middleware framework and is based on a container-component model. It supports the programming languages C++, Java and Python (in particular for scripting). ACS provides a well-tested platform that embeds standard design patterns and services.

3.2 OPC UA

The CTA arrays will comprise a multitude of different hardware devices (CCD cameras, mirror actuators, drive systems, high-voltage systems, calibration units, weather stations, etc.) that have to be controlled and/or monitored. Given the many different hardware devices with different hardware interfaces, a well-defined and standardized software bus to interface the devices in a common way is chosen. Thereby, the details of the underlying hardware and operating system as well as of the low-level programming are hidden and possible hardware changes will not affect the higher layers of the communication and control software.

For CTA, the Object Linking and Embedding for Process Control Unified Architecture (OPC UA) has been chosen. This interoperability layer unifies information exchange and provides a common interface for controlling processes. OPC UA is an industrial standard, designed to be independent of the operating system, and comes already with many devices such as automated systems driven by PLCs (e.g. the telescope drive systems) or can be developed by CTA teams using well-defined development kits.

4. CENTRAL CONTROL SYSTEM

The central control system allows the execution of observations according to a program given by the automatic scheduler or the operator via an user interface. In addition, the central control system manages the resources of the system, i.e. keeps track of the availability of sub-systems including telescopes and provides the monitoring and configuration services for all hardware and software independent of on-going observations.

The central control interacts with all entities in the system to organize the acquisition of Cherenkov images observed by more than one telescope. A simplified view of an execution of an observation is shown in Fig. 3. After the properties of the observations have been provided by the central scheduler (see Sec. 6), the central control

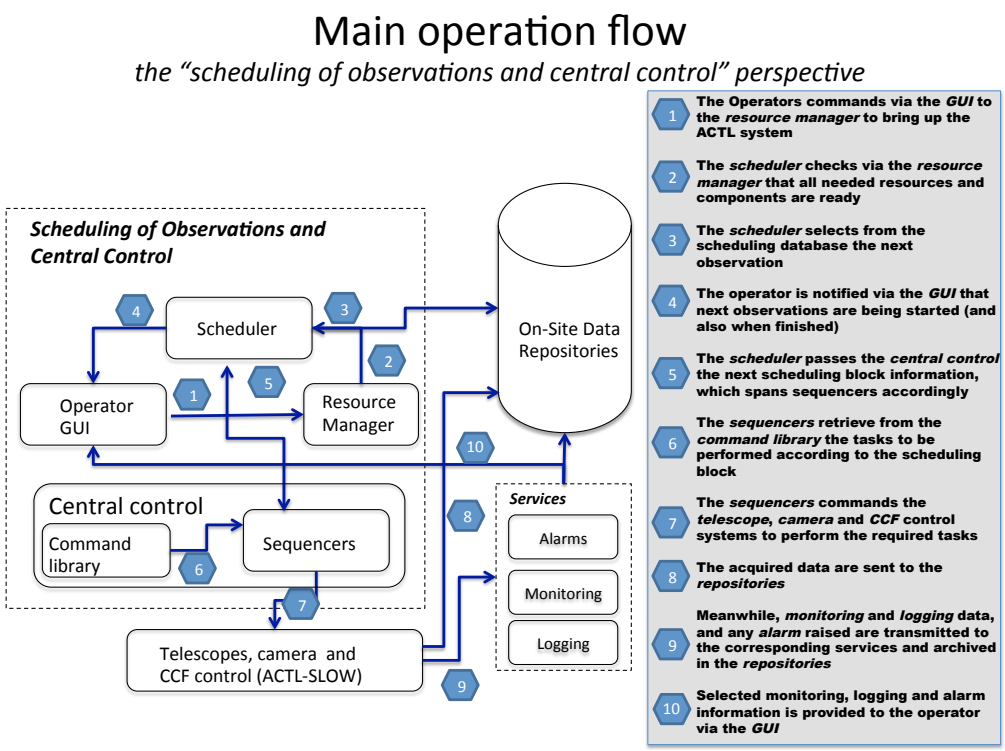


Figure 3: Schematic view of the interaction of the central control components and other ACTL elements during a simplified operation sequence in an observation night. The sequence of events is shown for a sequence free of errors.

creates a sub-array of telescopes in run-time following the observation request. It decomposes the observation details in a sequence of commands from sub-array to telescope level using a library of ACS python scripts, to be executed by the instrument slow control software (see Sec. 7). A first prototype of the central control system is currently under construction and will enable us to integrate and test several ACTL systems in one common environment for a full emulated observation run.

The operator is constantly informed about the on-going observations and the status of all sub-systems via the operator graphical user interface (GUI). The design of the operator GUI is done considering not only the implementation technology and the look and feel, but more importantly the usability by and the interaction with the operator. The large scale of the array to be controlled (approximately 100 telescopes in the Southern array) will require a careful design of the system and choice of technology. For the wealth of accessible information and controllable parameters, human factors in the interaction of the operator with such an interface have to be considered early in the design phase. A collaborative effort has been established with experts in Human-Computer Interaction (HCI) to address these points.[11]

5. MONITORING AND CONFIGURATION

Logging and monitoring data is taken continuously throughout on-going observations. Monitoring data is taken for both hardware and software, where the latter could be the status of a process or of a database service. Current estimates for the individual items to be monitored in CTA are around 10^5, which are usually sampled with a regular rate on the order of Hz or whenever a value changes more than a predefined threshold. For an effective analysis and interpretation of this large quantity of data, it is crucial to maintain links between an element's monitoring and configuration data, a concept already introduced by ALMA. From the configuration data one can determine e.g. when the configuration has been changed and which parts of the system are affected. Prototyping

efforts in CTA for the monitoring and configuration part are on-going, using the telescope prototype projects as test facilities. Two prototypes for the front-end of the monitoring system have been successfully tested,[12,13] the blobber-collector mechanism provided by ACS and used by ALMA as well as the property recorder. The data is provided to the back-end, which takes care of storing the gathered data into the CTA repositories. To solve the problem of gathering and storing data for the large expected number of monitoring points, NoSQL databases are evaluated. MongoDB and Cassandra are currently considered as the most promising candidates. The choice of the technology will follow the detailed specification of the CTA needs, including the description of the data and computing model as well as the planned usage of the data, and a thorough testing with real-world examples.

6. CENTRAL SCHEDULER

The observation program on different time scales of the CTA observatory is planned to be automatically created by a scheduler software that ensures that observations can be performed without human intervention, at high efficiency and optimized for maximum scientific output of CTA. The observation plan will be produced from the available and accepted proposals and those constraints given by the CTA observatory that can be computed beforehand. This scheduling process is devoted to translate the scientific prioritization, based on the proposal priority and on the CTA observatory policies, into a feasible and optimized real-time observation schedule on timescales from several months to single nights. To maximise the scientific return, the scheduler supports the splitting of the CTA arrays into sub-groups of telescopes for simultaneous observations. The scheduler will be tightly coupled with the ACTL system to be able to react dynamically to changes of observation conditions, e.g. weather or the state of the telescope array, with reaction times in the range of minutes. The challenge for the scheduler in producing the new observation schedule is to take the changed observation conditions into account while keeping the long-term scientific goals. Another important application of the central scheduler is the fast, i.e. on timescales of seconds, reaction to internal and external alerts e.g. due to high priority transient events. With the advent of upcoming high sensitivity instruments across all wavebands, a smart filtering and combination of external and internal triggers is of increased importance and is planned as part of the scheduling procedure.

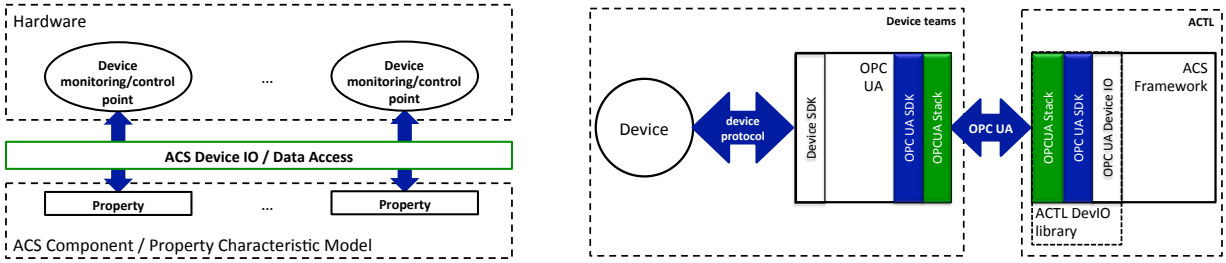
The scheduling processes described above are impractical for manual planning due to the complexity in computing the enormous amount of possible combinations, especially in the search for a near-optimal solution in the short timescale range. Several mathematical approaches are being explored for the different scheduling processes[14] and, depending on its time-criticality, range from simple heuristics to more complex algorithms based on artificial intelligence.

Prototypes for the scheduler, providing long-term and short-term plans, have already been developed together with a simulator and user interface, which can run multiple scheduling simulations in parallel and automatically summarize the results obtained for an agile performance analysis. The prototype is being tested using real-world examples from current-generation Cherenkov experiments as well as key science programs of the CTA observatory to gain valuable information to improve the observation strategy for higher scientific return. Revision of these prototypes is currently on-going on the basis of top-level science use cases[15] and using the recent added functionality of multiple sub-arrays.

7. INSTRUMENT SLOW CONTROL SOFTWARE

The task of the instrument slow control software is the integration of telescopes with their Cherenkov camera sub-systems and the auxiliary devices, used e.g. for the telescope array calibration, into the overall ACTL framework. The main challenge results from the complexity of devices and protocols as well as from the sheer number of telescopes coupled with the need for a highly stable and performant operation of the multi-telescope array.

The usage of software frameworks such as ACS and OPC UA mitigates the challenges of diversity of hardware technologies. ACS already provides a mechanism to encapsulate the communication with the processes corresponding to the hardware devices with its "DevIO" mechanism, see Fig. 4(a). The base DevIO was extended to allow using a standard software layer based on the OPC UA standard as the access point to the devices to avoid specialized DevIO classes depending on environment and device firmware, see Fig. 4(b). The DevIO library



(a) (b)

Figure 4: (a) The ACS device integration concept based on the 'DevIO' bridge. The DevIO bridges the device monitoring points (e.g. for a sensor) and control points (e.g. for a power switch) to the ACS standard elements labeled 'properties'. (b) The standard DevIO library for bridging OPC UA to ACS as used in the ACTL system. The device is accessed from the OPC UA server via any device specific protocol (e.g. a serial line communication) while the OPC UA server communicates with the ACTL ACS-based software via the standard ACTL DevIO library.

was extended to cover communication aspects, e.g. method calls and alarm transmission, in a standard manner additional to those of the standard DevIO implementation.[9,16] This paradigm for the integration of devices in the ACTL system has been successfully tested on telescope prototype projects,[12,13] which have constituted an important driving force in the evaluation of the concepts and proposed software technologies. The telescope prototype projects also make it possible to develop and test solutions to steer, monitor and readout the telescope prototypes in the context of ACTL.

While the main concepts have been successfully tested, the standardization of concepts and solutions to arrive at a uniform telescope integration model, including uniform interfaces and control hierarchy, among all prototype projects remains the major challenge for the next steps of developing common slow control software components.

8. CLOCK DISTRIBUTION AND TIME-STAMPING SYSTEM

Data from a wide variety of sources (e.g. telescope cameras, weather stations) distributed throughout the CTA array must be correlated in time. Successful association of these data requires sufficiently precise timestamping mechanisms. For quantities such as weather data that need only be known on time scales larger than typical network latency times on the order of ms, adequate timestamps can be obtained by timestamping after the data are read out, using system clocks synchronized with e.g. the Network Time Protocol (NTP). For these cases, round-trip times in the array (typically 10 µs for a distance of 2 km at fiber speed) are negligible.

The requirements are far more stringent for the Cherenkov event data read from the individual telescope cameras. Here relative telescope-to-telescope timing accuracy of the order of 10 ns is required in order to correctly associate events which are seen by several telescopes in coincidence and to efficiently reject non-coincident events. If the time-resolved Cherenkov wavefront is used for reconstruction methods such as the shower direction, and background rejection, then a relative timing accuracy of 2 ns is desirable. Studies of pulsars, active galactic nuclei (AGN), and other periodic and transient phenomena also require that an absolute time be attached to each reconstructed photon candidate. This is most easily done via the same mechanism that provides the timestamps to the telescopes, and imposes an additional requirement that these timestamps provide an absolute time good to 1 µs. These more stringent requirements mean that a specific solution must be found to distribute a clock and timestamp information to the telescope cameras. The backbone is based on a White Rabbit network,[17,18] which provides clock synchronization capability with ns accuracy and sub-ns precision for more than 1000 nodes, connected by fibers of up to 10 km in length.

The timing distribution system delivers precision clock signals to all devices (e.g. the Cherenkov cameras) in the array as needed, where triggers are timestamped and the information is forwarded, together with additional information about the nature of the trigger and the status of the telescope, to any client which may require it. An overview of the timing distribution system is shown in Fig. 5. It is supported by a dedicated White Rabbit optical fiber network over which the reference time signal and the timestamped event triggers from each

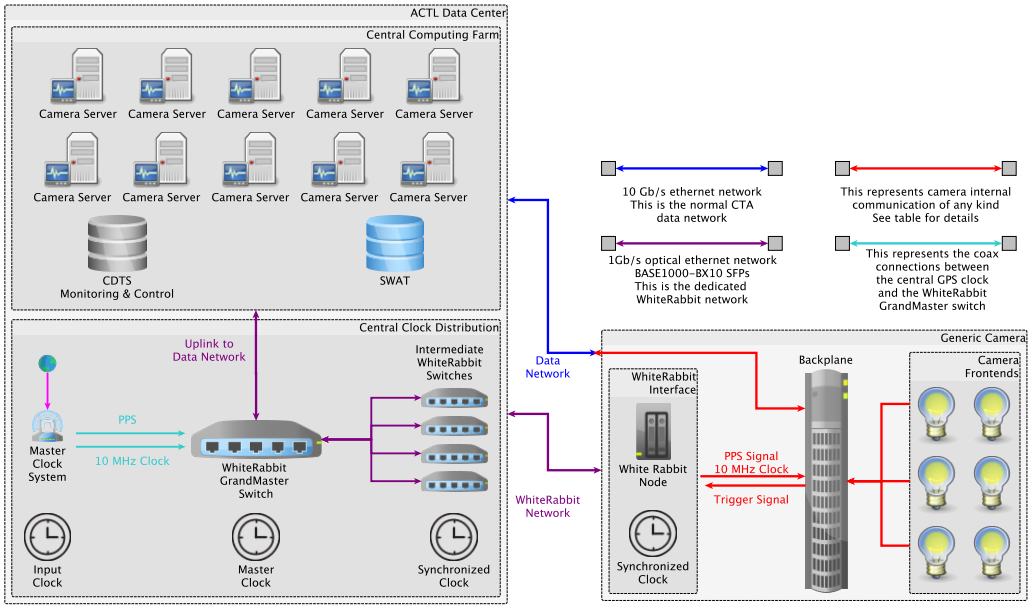


Figure 5: Scheme of the proposed White Rabbit optical fiber network for the timing distribution system together with the main elements (White Rabbit nodes, White Rabbit switches, GPS clock system, computing farm), which are connected via the White Rabbit network.

Cherenkov camera are distributed. Since its bandwidth per fiber connection is limited to 1 Gb/s, special care has to be taken to avoid bottlenecks within the network. Therefore, a strict separation between the data network, which is used for the transfer of the bulk Cherenkov data (see Sec. 6), and the White Rabbit network is the favored solution. The network itself consists of a single master switch that provides the time and frequency reference for the entire network, connected to a set of slave switches and the White Rabbit nodes (e.g. the Cherenkov cameras). For the CTA application, the time and frequency references provided by the master switch are planned to be disciplined by an external GPS clock system which provides the absolute timing information. Dedicated software processes monitor all White Rabbit nodes and switches, as well as the master clock system, which is reported, as the system status and performance statistics (packet rate, dropped packets, etc.), to the instrument monitoring system.

9. ARRAY TRIGGER

The overall data stream that the ACTL system has to cope with is dominated by the telescope Cherenkov cameras whose local camera triggers select events at rates that are assumed to vary between 1 kHz to 10 kHz, leading to individual telescope data rates in the range of 10 MB/s to about 5 GB/s. The requirement of stereoscopic time coincidence (i.e. ≥ 2 telescopes have triggered simultaneously) as realized by a central array trigger allows stabilization of data readout rates and reduction of the data volume. The array trigger acts on information sent by the local telescope triggers to select shower signatures and distinguish background processes (obvious hadrons and muons). The full trigger system, including the array trigger, must be flexible enough to pass through events such as muon rings that are useful for calibration while keeping the overall volume of recorded data under control.

For the CTA application, the current design of the array trigger favors a software-based solution due to the relatively large round-trip signal propagation times ($\approx 10\,\mu s$) and coincidence windows used in CTA. A software-based array trigger relaxes the demands on the required depth of camera electronics buffers ($\approx 4\,\mu s$ for a hardware-based trigger). The software-based array trigger is a dedicated process that runs in the near real-time regime. It receives and collects the input record stream of trigger timestamp and other identifying information, e.g. the type of the trigger, from each telescope and searches for coincidences in a coincidence time window, configurable for each telescope pair. The resulting list of coincidence-flagged events is then sent back

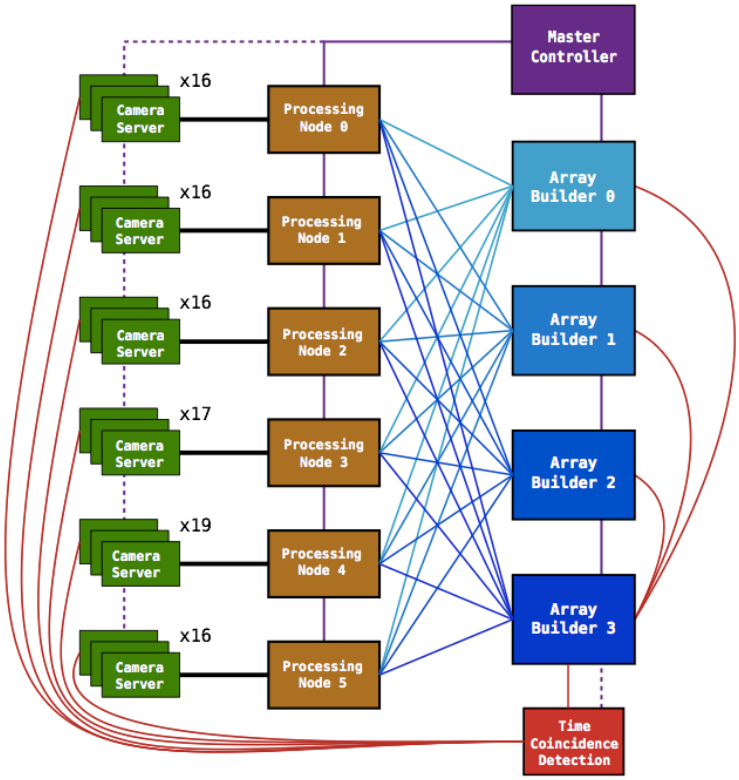


Figure 6: Simplified scheme of a prototype data acquisition system.

to all clients (e.g. Cherenkov cameras, data acquisition system) to enable the overall data readout and related cross-checks.

First demonstrator systems based on this concept show promising performance. A first prototype system, based on a multi-threaded C-process, has been developed and tested. It uses Linux-specific epoll() system calls to efficiently manage hundreds of simultaneous network connections using the standard TCP protocol to the telescopes. To achieve the maximum performance, all the main operations on data streams (data reception, data merging, data sorting and array trigger detection) were performed in parallel by dedicated threads. The prototype successfully processed trigger input records delivered at rates up to 2 MHz and is capable of generating array triggers at rates up to 200 kHz to 400 kHz. The maximum array trigger rates are achieved for the minimum array trigger requirement of two coincident telescopes. Coincidence windows ranging from 50 ns to 10 µs have been tested. Next steps will include a full integration of the software array trigger prototype into the overall ACTL ACS framework, where an optimized balance for performance, reliability and maintainability has to be found.

10. DATA ACQUISITION SYSTEM

The data streams generated by the cameras after local camera triggers dominate the overall data rate of CTA of up to 30 GB/s (for the Southern array and after filtering of stereoscopic events). The data acquisition system is responsible for the readout, buffering, transfer and storage of this bulk data. In addition, it processes the individual events for data volume reduction and preparation of the data for the real-time analysis. Since the science return of the CTA observatory depends on the quality and efficiency of the recorded data, the data acquisition system has to fulfill stringent requirements on stability, performance and flexibility.

The design of the data acquisition system[19] follows a modular architecture, which has several advantages compared to a monolithic approach. Modularity allows for complex processing of event data to be decomposed

into simple and exchangeable tasks, thus simplifying the development and validation procedures. The data acquisition system is decomposed into several components, where each component has a simple task to perform while more complex tasks are created by plugging several components together. In addition, it enables load-balancing across several physical nodes and thereby helps to achieve the high availability required by allowing the reconfiguration of the system at any time. Specific master components are responsible for monitoring the system and for taking appropriate actions if necessary (e.g. optimization of the load-balancing).

The data acquisition is currently in the prototyping phase and a simplified view for one prototype is shown in Fig. 6. The general concept for camera data readout involves transmission of the bulk event data stream from the camera electronics to a camera server process running at a central location in the on-site data center. After initial camera event building (and processing), the next step is the transmission of the data to the central computing farm and data acquisition system, where further buffering, processing and storage of the event data is done. As part of this processing step, different data volume reduction scenarios will be applied and tested, including possible data volume reduction. Reducing this large data set should not discard scientifically useful data, thus each new data trimming process will be carefully evaluated and tested off-line before it is introduced in the data acquisition system. This can be achieved by the proposed modular architecture. Various data reduction strategies are under consideration, including compression, waveform reduction and zero-suppression. In the last step, the events are prepared for further processing of the real-time analysis[2]

The main design challenges for the data acquisition system are those of choosing the data format and the transport protocol. A compact and flexible data format is needed since the amount and content of data to be read out depends on the camera type. The transport protocol needs to be highly reliable and allow for the modular structure of the data acquisition system. A current prototype, built up to process data of 100 simulated cameras, is based on ZeroMQ[20] as the transport protocol and protocol buffers[21] as the data format. ZeroMQ implements features that include automatic connect/reconnect of new peers, automatic load-balancing and shared memory between processes or threads which support the current design of the data acquisition system. Protocol buffers are a standardized way of handling the serialization problem with its built-in forward and backward compatibility of the format, which allows for easy modification of the data content. While the current prototype focused on load-balancing and automatic failure recovery procedures, a maximum data throughput of beyond 10 Gb/s could be reached using data compression. While these results are promising, the final choice of data format and transport protocol will be done after further detailed specifications and testing, followed by a full integration and thorough testing of the data acquisition system with the timing system and the array trigger in a common environment.

11. SUMMARY

To build the array control and data acquisition (ACTL) system of CTA is a challenging task. Not only is the CTA observatory a complex installation of multiple (O(100)) telescopes of different types distributed over a large area, producing a relatively large amount of data, that needs to be controlled, monitored and readout, but is also being planned as an observatory with a lifetime spanning decades, resulting in stringent requirements for availability, reliability and maintainability for the ACTL software.

To mitigate these challenges, the ACTL project has decided to use software frameworks (ACS and OPC UA) throughout the project and has adopted a formal approach to establish a software architecture model as the basis for software development and organization of the ACTL project and as a reference for the entire CTA. While the model is further detailed, the links and run-time behaviour between the main building blocks will be specified. The model will then allow ACTL to establish methods and standards for the integration and testing of these building blocks.

Prototypes and concepts for the main building blocks of the ACTL system have been successfully tested in recent years. The main challenges in the next phases will be to integrate these prototypes in a coherent system, which works as a whole, and to create standardized and re-usable solutions for the entire project. In addition, in many of these building blocks important technology choices have yet to be made. But the established architecture model and procedures together with thorough tests with the prototype systems will support the decision-making process.

ACKNOWLEDGMENTS

We gratefully acknowledge support from the agencies and organizations listed under Funding Agencies at this website: http://www.cta-observatory.org/.

REFERENCES

[1] Acharya, B. S., Actis, M., Aghajani, T., et al., "Introducing the CTA concept," *Astroparticle Physics* **43**, 3–18 (Mar. 2013).

[2] Bulgarelli, A., Fioretti, V., Zoli, et al., "The On-Site Analysis of the Cherenkov Telescope Array," in [*Proceedings of the 34th International Cosmic Ray Conference (ICRC2015), The Hague, The Netherlands*], (2015).

[3] Oya, I., Füßling, M., et al., "The Software Architecture to Control the Cherenkov Telescope Array," in [*Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*], *Proceedings of the International Society for Optical Engineering (SPIE), These Proceedings* (2016).

[4] Pohl, K., Honninger, H., Achatz, R., and Broy, M., [*Model-Based Engineering of Embedded Systems: The SPES2020 Methodology*], Springer, Berlin, Heidelberg (2012).

[5] "SysML - OMG Systems Modeling Language 1.4." OMG. 2015.

[6] Kuhn, T. and Antonino, P. O., [*Model-Driven Development of Embedded Systems*], Embedded Software Engineering Congress, Sindelfingen, Germany (2014).

[7] "Enterprise Architect – Compendium of Enterprise Architect from Sparx Systems – Trainung Document." SparxSystems Software GmbH.

[8] Chiozzi, G., Jeram, B., Sommer, H., et al., "The ALMA common software: a developer-friendly CORBA-based framework," in [*Advanced Software, Control, and Communication Systems for Astronomy*], Lewis, H. and Raffi, G., eds., *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series* **5496**, 205–218 (Sept. 2004).

[9] Füßling, M., Oya, I., Schwanke, U., et al., "Towards a global software architecture for operating and controlling the Cherenkov Telescope Array," in [*Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*], *Proceedings of the International Society for Optical Engineering (SPIE)* **9152** (Aug. 2014).

[10] Wootten, A. and Thompson, A. R., "The Atacama Large Millimeter/Submillimeter Array," *IEEE Proceedings* **97**, 1463–1471 (Aug. 2009).

[11] Sadeh, I., Oya, I., Pietriga, E., et al., "Prototyping the graphical user interface for the operator of the Cherenkov Telescope Array," in [*Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*], *Proceedings of the International Society for Optical Engineering (SPIE), These Proceedings* (2016).

[12] Tosti, G., Schwarz, J., Antonelli, L. A., et al., "The ASTRI/CTA mini-array software system," in [*Ground-based and Airborne Telescopes V*], *Proceedings of the International Society for Optical Engineering (SPIE)* **9152**, 915204–915204–9 (July 2014).

[13] Oya, I., Anguner, E. A., Behera, B., et al., "The control system of the 12-m medium-size telescope prototype: a test-ground for the CTA array control," in [*Ground-based and Airborne Telescopes V*], *Proceedings of the International Society for Optical Engineering (SPIE)* **9152**, 91522G–91522G–8 (July 2014).

[14] Colomé, J., Colomer, P., Campreciós, J., et al., "Artificial intelligence for the CTA Observatory scheduler," in [*Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*], *Proceedings of the International Society for Optical Engineering (SPIE)* **9149**, 0 (Aug. 2014).

[15] Bulgarelli, A., Kosack, K., Hinton, J., et al., "The CTA Observatory Top Level Use Cases," in [*Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*], *Proceedings of the International Society for Optical Engineering (SPIE), These Proceedings* (2016).

[16] Oya, I., Füßling, M., Schwanke, U., et al., "Status and Plans for the Array Control and Data Acquisition System of the Cherenkov Telescope Array," in [*Proceedings of the 34th International Cosmic Ray Conference (ICRC2015), The Hague, The Netherlands*], (2015).

[17] Serrano, J., Alvarez, P., Cattin, M., et al., "The white rabbit project," in [*Proceedings of the ICALEPCS 2009, Kobe, Japan*], **TUC004**, 1–3 (2009).

[18] "White Rabbit." http://www.ohwr.org/projects/white-rabbit. (Accessed: 31 May 2016).
[19] Lyard, E., Walter, R., Kosack, K., et al., "Modern middleware for the data acquisition of the cherenkov telescope array," in [*Proceedings of the 34th International Cosmic Ray Conference (ICRC2015), The Hague, The Netherlands*], (2015).
[20] "ZeroMQ - Distributed Messaging." http://zeromq.org. (Accessed: 31 May 2016).
[21] Google Inc., "Protocol Buffers." http://developers.google.com/protocol-buffers. (Accessed: 31 May 2016).