ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD057

OVERVIEW AND CURRENT STATUS OF THE SKA-LOW MONITORING, CONTROL AND CALIBRATION SUBSYSTEM (MCCS)

E. L. Arandjelovic*, A. Child†, G. Chira‡, A. Clemens§, T. Moynihan¶
Observatory Sciences Ltd., UK
D. Devereux[1‖], M. Waterson**, SKA Observatory, UK
M. Davies††, J. Harvey‡‡, S. Melhuish§§, University of Manchester, UK
[1]also at CSIRO, Australia

Abstract

SKA-Low is the low frequency radio telescope currently under construction in Western Australia. At its final extent, it will consist of 512 stations up to 74 km apart, each containing 256 antennas which can be used in different combinations to digitally "point" the telescope. The Monitoring, Control and Calibration Subsystem (MCCS) is responsible for performing calibration and providing local monitoring and control of all of the LFAA (Low Frequency Aperture Array) hardware components. This includes managing the allocation of resources for an observation, and the aggregation of health status. SKAO has adopted the Tango control system framework, and the MCCS software comprises upwards of 18 different Tango devices, some of which are replicated dozens of times for a single station. This complexity poses a significant challenge to computing resources and reliability when considering how to scale up the system, first to the 16 stations to be constructed and integrated by January 2026, then 68 stations at the end of 2026, and targeting 307 stations by mid-2028. This paper will describe the MCCS architecture, report on our latest performance profiling, and discuss how we are preparing for the AA 2 construction milestone which will need to support 68 stations.

INTRODUCTION

The stated mission of the Square Kilometre Array Observatory (SKAO) is to provide two world-leading radio telescopes: SKA-Low in Western Australia's Shire of Murchison on the traditional lands of the Wajarri Yamaji targeting the 50-350 MHz observing range, and SKA-Mid located in the Karoo Region, South Africa, targeting 350 MHz - 15 GHz. This paper focuses on the Monitoring, Control, and Calibration Subsystem (MCCS) being developed to support SKA-Low, a "mathematical" telescope which when complete will comprise 131,072 log-periodic antennas to receive the radio signals [1], grouped into 512 stations and distributed in three spiral arms radiating from a central core. The signal is first boosted by a low noise amplifier and then converted to an analogue optical signal for transmission to a processing facility. Here the RF signals are digitized and transmitted 800 km off-site to the correlator / beamformer (CBF) where signals from different stations are combined, allowing high-resolution sky images to be formed from the interferometric visibilities or high-sensitivity narrow beams to be directed for deep pulsar searches [2].

This paper provides a brief overview of the MCCS software and its role in the operation of the SKA-Low telescope. It begins with a high-level description of the purpose of MCCS and how it fits into the overall telescope control system. The next section outlines the software architecture, and then provides an overview of some critical functionality including calibration and health monitoring. The paper is concluded with a discussion about performance profiling and how we are tackling the challenges involved in scaling up to 68 stations by the end of 2026.

A BIRD'S EYE VIEW OF MCCS

Functionally, MCCS is part of the Low Frequency Aperture Array (LFAA) telescope subsystem, and sits underneath the Telescope Manager (TM) which integrates all the engineering entities in the system to form the telescope device [3]. MCCS interfaces to the Telescope Monitoring and Control (TMC) software used by the telescope operators to control the telescope. The purpose of MCCS is to:

- Control and manage the LFAA receiver electronics
- Acquire, process and apply signal chain calibration data
- Acquire, process and provide diagnostic data products
- Monitor, aggregate and provide resource health to TM
- Provide an interface to TMC to acquire and transmit science data to the correlator and beamformer in the Central Signal Processor (CSP)

This context is illustrated in Fig. 1. SKAO has adopted the Tango [4] control system framework, and at the time of writing MCCS consists of 18 different Tango devices which are deployed using Kubernetes [5]. These Tango devices implement the SKA Control Model, a standard set of state transitions and mode indicators developed to unify the design of control system components across the telescope [6]. The device hierarchy that has been implemented reflects the layout of the physical components of the SKA-Low telescope for which MCCS is responsible, namely:

* ela@observatorysciences.co.uk
† abc@observatorysciences.co.uk
‡ gsc@observatorysciences.co.uk
§ ajc@observatorysciences.co.uk
¶ tdm@observatorysciences.co.uk
‖ drew.devereux@skao.int
** mark.waterson@skao.int
†† mark.davies-6@manchester.ac.uk
‡‡ joseph.harvey@manchester.ac.uk
§§ simon.melhuish@manchester.ac.uk

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD057

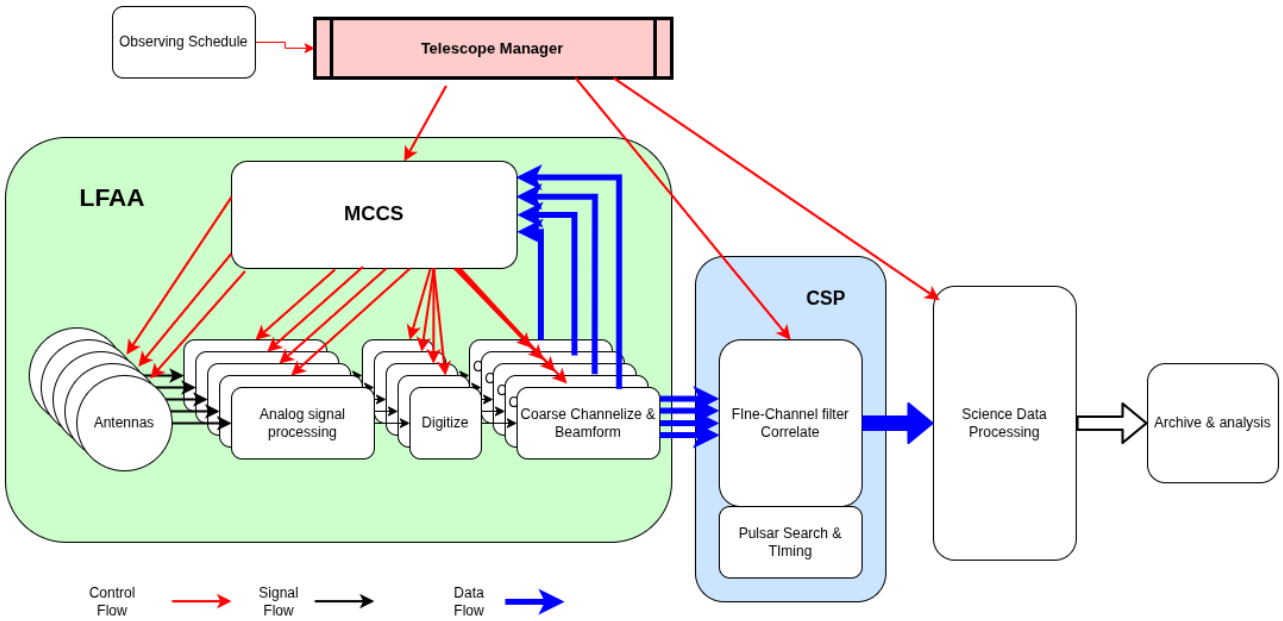

Figure 1: SKA-Low telescope context.

- Field station – contains signal capturing and front-end components.
- Signal Processing Subsystem (SPS) – digitizes and combines the field station signals, forming "station beams".

These are now discussed briefly in turn.

Field Station

Each field station contains a 256 antenna-element array, a Power and Signal Distribution System (PaSD), and fibre cable assemblies to transmit the signals to either the Central Processing Facility (CPF) or a Remote Processing Facility (RPF). The PaSD consists of one Field Node Distribution Hub (FNDH), one Field Node Communications Controller (FNCC) and 24 local Smartboxes, and is controlled and monitored by MCCS via a Python Modbus interface [7]. Each station deployment includes a Fieldstation Tango device to manage the low-level Tango devices which interface to the hardware.

Signal Processing System

Each station has 16 Tile Processing Modules (TPMs) which perform digitization and signal processing of the antenna signals using FPGAs. This includes channelization of the sampled data, applying delay corrections, and combining the streams to form beams [8]. Intermediate data products are also provided to MCCS as well as station beams. MCCS controls FPGA operation over a UDP interface fronted by a Python API. The TPMs are housed in subracks alongside a subrack management board (SMB) which is also under MCCS control. Clean clock signals are essential to SPS operation and this is in part a responsibility of the SMB, coordinated between stations using the White Rabbit protocol. The SMB also performs various other ancillary control and monitoring functions. Each station deployment includes an SPSStation Tango device to manage the Tile and Subrack devices.

In addition to controlling these physical components of the telescope, MCCS must also manage and represent logical concepts such as beams (signals combined from multiple antennas and/or stations) and subarrays. Subarrays contain subsets of the telescope's signal collecting resources, and are effectively operated as independent telescopes [2].

ARCHITECTURE AND DESIGN

A high-level architectural diagram of MCCS is shown in Fig. 2. The main MCCS software components (which each have their own Gitlab repository) are as follows:

- *ska-low-mccs*: the top-level component which provides Tango devices exposed directly to the telescope operators for interacting with MCCS.
- *ska-low-mccs-pasd*: the interface to the fieldstation hardware components, including a top-level Fieldstation device which manages them.
- *ska-low-mcccs-spshw*: the interface to the SPS hardware components, including a top-level SPSStation device which manages them.
- *ska-low-mccs-daq* and *ska-low-mccs-daq-interface*: the data acquisition system which captures data from the TPMs for calibration, diagnostic and testing purposes.
- *ska-low-mccs-calibration*: the subsystem responsible for station calibration.
- *ska-low-mccs-common*: contains common functionality and utilities that are used across the other repositories.

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD057

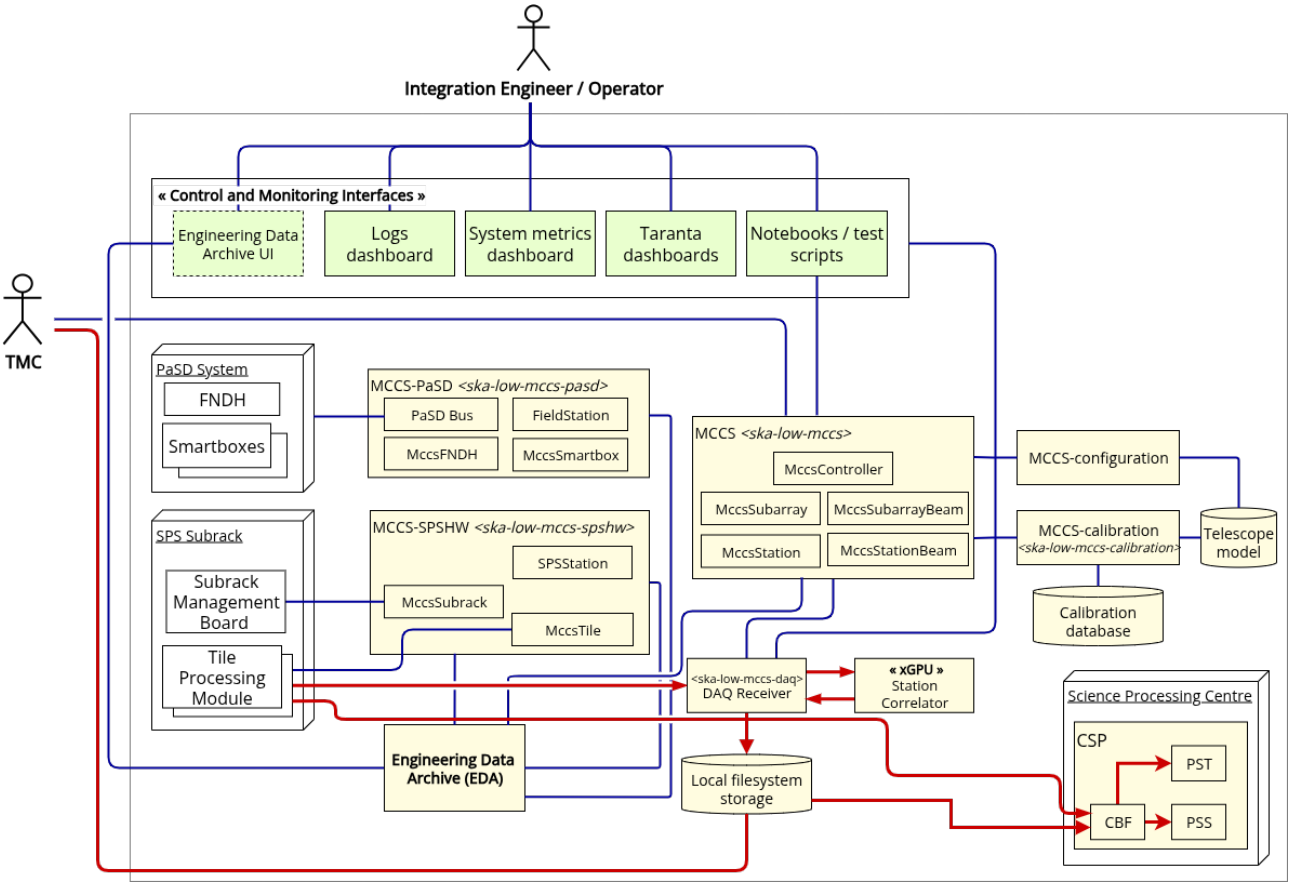

Figure 2: MCCS software architecture.

The block diagram shown in Fig. 3 illustrates how MCCS interacts with the Telescope Manager via the top-level `MccsController` (a Tango device which manages all the resources in the MCCS system), and a fixed number of `MccsSubarray` devices which can be used for observations. These in turn interact with devices lower down in the hierarchy: the `MCCSStation` which manages the `MccsFieldStation` and `MccsSPSStation` devices, and the `MccsSubarrayBeam` and `MccsStationBeam` devices which are responsible for configuring the TPMs to do beamforming.

Tango Device Design

In order to facilitate consistent design across the telescope and reduce code duplication, SKAO have developed some base classes with skeleton implementations of Tango devices that follow the SKAO control model [9]. MCCS builds on this approach, and some additional functionality has been developed in `MccsBaseDevice`, the base class for all MCCS devices, to provide an inheritance mechanism for propagating control through the device hierarchy. Since this hierarchy from `MccsController` down to the hardware-facing devices such as `MccsSmartbox` has multiple layers, the process for turning them all ONLINE is quite involved. To solve this problem, each device is configured with the Tango Resource Locator (TRL) of its parent device, and subscribes to change events on its operational mode (a Tango attribute defined in the control model called `AdminMode`). It then acts on these change events to push itself into the parent's new state automatically.

For hardware-facing devices, a common approach to the design has been taken to separate the details of the hardware interface from Tango. Each such device has a component manager which is responsible for establishing and maintaining communication with the component. This typically initiates polling of the hardware, implements any commands, and communicates data and status back to the Tango device through the use of callbacks.

Calibration

For SKA-Low, calibration refers to the process of collecting and processing sky data into signal chain parameters (delays / phases and gains), and applying these on command to correct for local variations in the performance of each individual signal chain. The calibration process has a number of steps outlined next.

Step 1: Acquire calibration data For a specified coarse channel, the TPMs send channelised antenna signals to a DAQ receiver running on the MCCS cluster, where a GPU-based correlator is used to form visibilities between

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD057

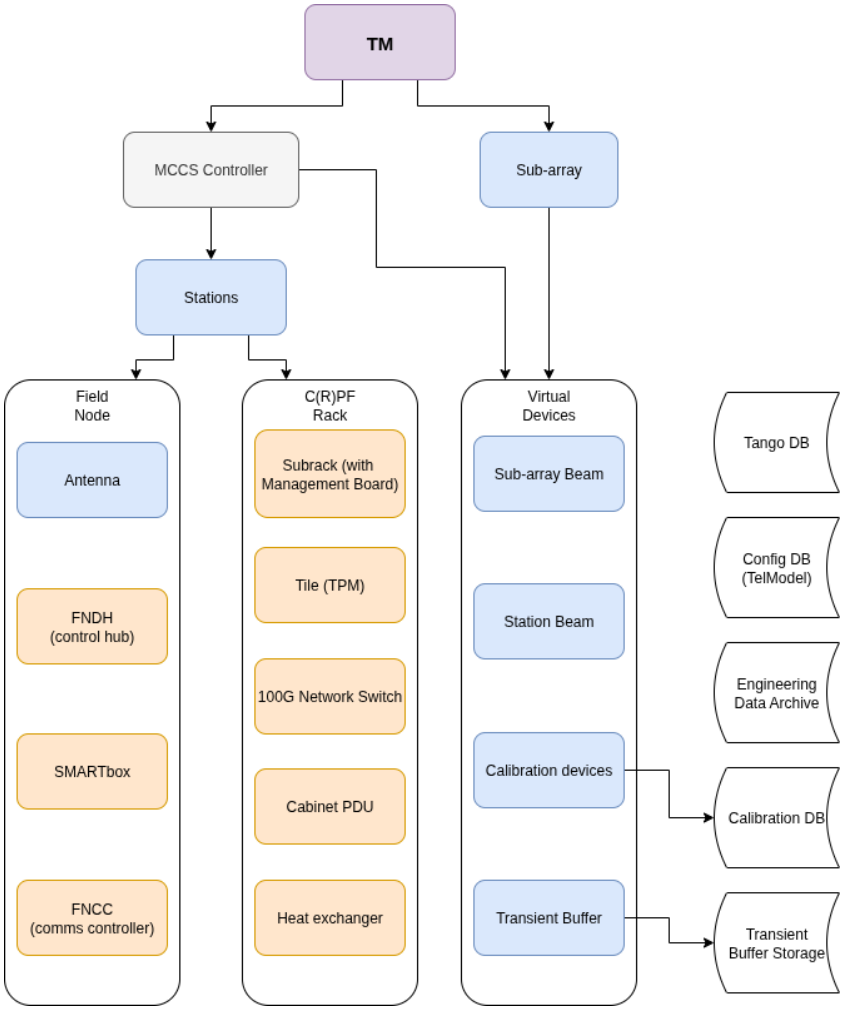

Figure 3: MCCS block diagram.

all pairs of antennas. After data collection visibility files are sent to a calibration solver.

Step 2: Perform and store the calibration The calibration solver compares the gathered visibilities with computed visibilities from a model sky. This uses a combination of the global diffuse sky model provided by PyGDSM [10] and a point-like source to represent solar emission. The solver uses this to calculate a 'calibration solution' (HDF5 file [11]) for the given data, and stores it in an artefact repository along with metadata about the observation context e.g. outside temperature.

Step 3: Retrieve and apply the calibration During the MCCS `Configure` command which prepares a subarray for observation, a calibration solution appropriate for the current operating conditions is retrieved from the artefact repository and applied.

A `StationCalibrator` Tango device is responsible for orchestrating the retrieval and storage of calibration solutions. This delegates to additional Tango devices `StationCalibrationStore` and `CalibrationSolver`. A database is used to store the calibration solution metadata and path to the solution in the artefact repository.

Health Monitoring

For a system as large and complex as SKA-Low, it is essential for the operators to be able to ascertain the status of each component and how this impacts on the telescope's readiness for observing. The SKA Control Model defines four possible health states for a device to be in:

1. OK - the device is working normally.
2. DEGRADED - only part of its functionality is available.
3. FAILED - it is unable to perform core functionality.
4. UNKNOWN - the system is unable to ascertain health status.

Each MCCS Tango device has a health model which stores and manages health on behalf of the device. It can work in two ways:

1. If the device manages hardware, then the health model obtains an assessment of hardware health from the device's hardware manager.
2. If the device manages subservient devices (for example the FieldStation device which manages an FNDH and

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-WEPD057

set of smartboxes), then the health model obtains an assessment of aggregate subservient device health.

For the latter case, the model is instantiated with a set of thresholds which define the number of subservient devices which need to be in each health state for this to cause a cascade upwards. For example, this would include the number of smartboxes which need to be in the DEGRADED state for the Fieldstation to become DEGRADED. This is a challenging area to get right, and at the time of writing there are still some unanswered questions about how to deal with devices that are in an UNKNOWN state because they have been switched off.

PREPARATION FOR AA 2

SKAO is constructing and verifying the two telescopes in stages referred to as array assemblies (AA). At the time of writing, a test array of four SKA-Low stations, AA 0.5, is undergoing science commissioning. The next milestone is AA 1 (16 stations) planned for the end of 2025, and following that AA 2 will deliver 64 stations at the end of 2026. Given that an MCCS deployment for one station requires over 100 Tango device servers, scaling the system up to 64 stations poses some significant reliability and performance challenges. To prepare for this, we have invested some time in profiling the system using py-spy [12]. The hardware-facing devices were evaluated when polling the devices at the default rate, and other devices were evaluated during functional test runs in a Kubernetes cluster and some specific calibration functions. This investigation identified some redundant tasks, as well as efficiency improvements that could be made by modifying the polling strategy to better match the needs of operators. The "flame-graph" visualization was particularly useful in helping to identify time-expensive operations such as the Python deep copy. Some work has also been done on the deployment side, for instance making more efficient use of Kubernetes by reducing the number of pods required per station.

CONCLUSION

MCCS is a core part of the SKA-Low control system, responsible for monitoring and control of the LFAA receiver electronics as well as calibration, diagnostic data acquisition, and health aggregation. Building on the framework provided by the SKA Tango base classes and control model, MCCS implements a hierarchy of Tango devices from hardware-facing devices such as those controlling the TPMs and PaSD, up to the top-level `MCCSController`. A total of more than 100 devices are deployed per station, with 4 stations currently undergoing science commissioning and preparation underway for scaling up to 64 stations by the end of 2026.

ACKNOWLEDGEMENTS

A great many people have been involved in the evolution of the MCCS software including SKAO architects, domain specialists, and external collaborators. We would also like to acknowledge the contributions of the integration and test teams based in Australia, as well as the firmware development team who we work closely with at the MCCS hardware interface level.

REFERENCES

[1] L. Sevgi, "The antenna as a transducer: Simple circuit and electromagnetic models", *IEEE Antennas Propag. Mag*, vol. 49, pp. 211–218, 2008. doi:10.1109/MAP.2007.4455907

[2] M. G. Labate *et al.*, "Highlights of the Square Kilometre Array Low Frequency (SKA-LOW) Telescope", *J. Astron. Telesc. Instrum. Syst.*, vol. 8, no. 1, 2022. doi:10.1117/1.JATIS.8.1.011024

[3] Y. Gupta, V. Mohile, J. Kodilkar, R. Uprade, Y. Wadadekar, and S. R. Chaudhuri, "Telescope Manager for the SKA", *J. Astrophys. Astron.*, vol. 44, no. 1, 2023. doi:10.1007/s12036-023-09908-0

[4] Tango Controls Collaboration, *Tango Controls*. https://tango-controls.readthedocs.io/en/latest/overview/overview.html

[5] Kubernetes. https://kubernetes.io/

[6] S. Vrcic and T. Juerges, "SKA telescope control system design and status", in *Proc. SPIE 13101, Software and Cyberinfrastructure for Astronomy VIII*, 2024. doi:10.1117/12.3018723

[7] E. L. Arandjelovic, D. Devereux, J. Engelbrecht, and U. K. Pedersen, "Tango Integration of the SKA-Low Power and Signal Distribution System", in *Proc. ICALEPCS'23*, Cape Town, South Africa, pp. 1526–1528, 2023. doi:10.18429/JACoW-ICALEPCS2023-THPDP077

[8] G. Comoretto *et al.*, "The Signal Processing Firmware for the Low Frequency Aperture Array", *J. Astron. Instrum.*, vol. 6, no. 1, 2017. doi:10.1142/S2251171716410154

[9] S. N. Twum, D. Devereux, T. Juerges, J. A. Venter, T. Ives, and J. Engelbrecht, "SKA base classes: a case study of reusability of code in a large-scale telescope control system", in *Proc. SPIE 13101, Softw. Cyberinfrastruct. Astron. VIII*, 2024. doi:10.1117/12.3018765

[10] PyGDSM. https://github.com/telegraphic/pygdsm

[11] HDF5. https://www.hdfgroup.org/solutions/hdf5/

[12] py-spy. https://pypi.org/project/py-spy/