# INTEGRATING THE FIRST SKA MPI DISH INTO THE MeerKAT ARRAY

S. N. Twum*, A. F. Joubert, K. Madisa, SARAO†, Cape Town, South Africa

*Abstract*

The 64-antenna MeerKAT interferometric radio telescope is a precursor to the SKA which will host hundreds of receptor dishes with a collecting area of 1 sq km. During the pre-construction phase of the SKA1-MID, the SKA DSH Consortium plans to build, integrate and qualify an SKA1-MID DSH Qualification Model (SDQM) against MeerKAT. Before the system-level qualification testing can start on the SDQM, the qualified Dish sub-elements have to be integrated into the SDQM and set to work. The SKA-MPI DISH, a prototype SKA dish funded by the Max Planck Institute, will be used for early verification of the hardware and the control system. This prototype dish uses the TANGO framework for monitoring and control while MeerKAT uses the Karoo Array Telescope Control Protocol (KATCP). To aid the integration of the SKA-MPI DSH, the MeerKAT Control and Monitoring (CAM) subsystem has been upgraded by incorporating a translation layer and a specialised SKA antenna proxy that will enable CAM to monitor and command the SKA dish as if it were a MeerKAT antenna.

## INTRODUCTION

### *MeerKAT Overview*

The MeerKAT telescope is a 64 dish interferometric[1] array which is located in the northern cape of South Africa. Lessons learned from the KAT-7 telescope, a seven dish interferometric array engineering prototype built prior to MeerKAT construction, were used to build a better and much more sensitive MeerKAT telescope. First light was received from the DEEP2 [1] commissioning field with 4 dishes in May, 2016. DEEP2 was then observed with a progression of 16, 32 and finally 64 dishes. The resolution achieved for these DEEP2 observations were distinctively sharp, exposing finer details for each array progression.

MeerKAT was launched in July 2018 revealing a very detailed image of our galactic centre and has since been fully operational doing different science studies with a variety of subarray configurations. The receptors, encompassing the antenna structure with the main reflector, sub-reflector and all receivers, digitisers and other electronics installed, are controlled and monitored using the KATCP - a communications protocol built on the TCP/IP layer (see Fig. 2). The CAM subsystem interfaces with all these subsystems to provide health, state and alarm information, and to execute overall control.

* stwum@ska.ac.za
† South African Radio Astronomy Observatory
1 Radio interferometry is a technique which simulates telescopes with diameters equal to the longest baseline

### *SKA Dish Prototypes*

MeerKAT is a pathfinder telescope and will become part of SKA Phase 1. Phase 1 marks the commencement of the construction of SKA1-MID Dish array. This array will constitute 133 15-m diameter dishes and 64 13.5-m MeerKAT dishes spread over approximately 150 km in the Karoo region.

The DSH consortium has delivered two prototype dishes for the SKA1-MID array. The first SKA dish prototype (SKA-P) was built and installed in China. It was put under a series of structural tests to identify discrepancies with the designs. The SKA-MPI dish, the second of the two prototype dishes, is funded by the Max-Planck Institute and is currently being assembled on the SKA site in South Africa.

The SKA DSH Consortium will integrate and qualify the SDQM on the Karoo site as part of the System Critical Design Review (CDR), [2]. Preceding this will be the verification of the dish sub elements using the prototype dish. Part of the SKA DSH consortium's effort in qualifying the SQDM is to make optimal use of the existing MeerKAT Receptor Test System (MKAT-RTS) [3] by leveraging the qualified SKA1-MID and MeerKAT hardware:

- SKA1-MID Dish Structure (DS)
- SKA1-MID Local Monitoring and Control (LMC)
- SKA1-MID Single Pixel Feeds (SPF) - Bands 1 and 2, SPF Controller and Services
- SKA1-MID Dish Fiber Network
- RTS Ku-Band Receiver
- MeerKAT L-band and UHF Digitisers
- MeerKAT CAM and Data Switches

In the following sections we explain the activities executed to deliver the MeerKAT CAM DSH proxy software which takes advantage of a translator to connect to the SDQM LMC whilst providing a common interface to the rest of CAM.

## INTEGRATING THE DSH LMC INTO THE MEERKAT RECEPTOR TEST SYSTEM

The SARAO CAM team has been tasked, by the SARAO Software Engineering (SE) team, to support the DSH consortium by developing a KATCP/TANGO translator that will be used between the MeerKAT Receptor Test System (MKAT-RTS) and the SDQM LMC in qualifying the prototype dish. This is one of the two translators that was first developed as part of a proof of concept whilst experimenting with the TANGO framework in the context of a real telescope system [4], MeerKAT.

### *The SKA-1 MID Architecture*

SKA is made up of multiple components, named elements. One of those elements, the Telescope Manager (TM), acts as the telescope's main control system. However, because managing the hierarchy of multiple devices of the telescope is such a complex task, each element has developed its own local control and monitoring component that monitors and gathers data from the element and sends it to TM. TM communicates directly with the elements' LMCs through the chosen TANGO Controls [5] protocol as shown in Fig. 1.

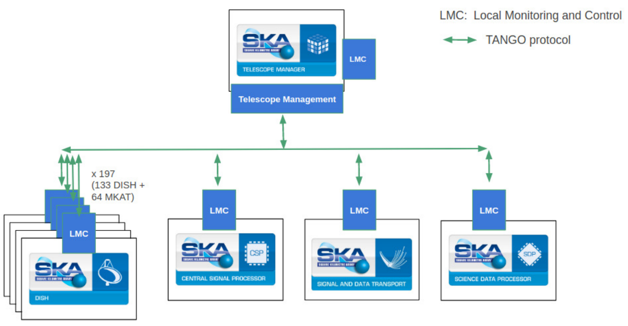

Figure 1: Simplified overview of the SKA architecture showing the TANGO monitoring and control interfaces from LMC to the different devices.

## *The MeerKAT Architecture*

The MeerKAT architecture is quite similar in terms of the subsystems of the telescope see Fig. 2, although it only has 64 antennas. MeerKAT CAM is to MeerKAT what Telescope Manager (TM) is to SKA. The only major difference is the communication protocol, KATCP, that is used for transmitting control and monitoring data between different subsystems.

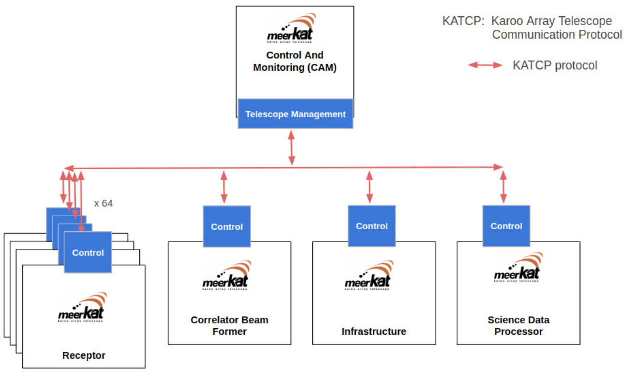

Figure 2: Simplified overview of the MeerKAT architecture showing the KATCP monitoring and control interfaces from CAM to the different devices

## *The MeerKAT + DSH Architecture*

The key idea in qualifying the SDQM is to leverage the existing MeerKAT system. To accommodate the new SKA dish prototype, the MeerKAT CAM has to emulate the SKA's TM by implementing a TANGO interface to be able to monitor and command the new SKA dish, see Fig. 3. The main point of entry on the DSH element is through its LMC component. This component is responsible for managing the DSH element's external CAM interface, and the DSH subsystem controllers, in order to command and monitor the antenna behaviour.

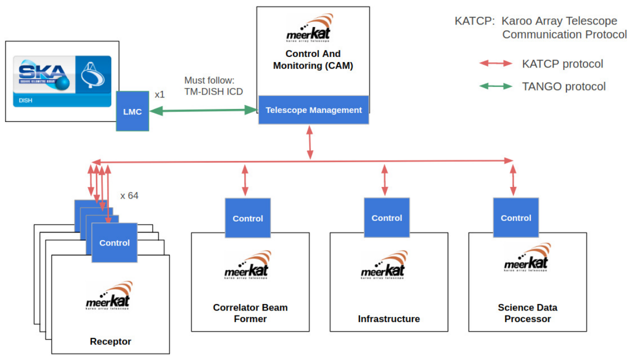

Figure 3: Simplified overview of the MeerKAT architecture showing the KATCP monitoring and control interfaces with the DSH LMC.

## *DSH LMC Simulation*

The DSH Consortium provided a virtual machine (VM) of the DSH LMC software running back-end simulators of the SKA1-MID hardware to use for testing when developing the DSH proxy. However, because the VM is extremely resource-heavy (~64 gigabytes in size), we had to create a light-weight simulated version of it in order to use it in the CAM's simulated environment. The light-weight DSH LMC simulator was developed using some of the tools from by the TANGO community, viz: tango-simlib [6] and fandango [7].

Fandango is a python library, written by Sergio Rubio from ALBA Synchrotron, for developing functional and multi-threaded control application and scripts. It was selected on the basis that it can be used to take a snapshot of a running TANGO device and export the device's configuration into a json file. The generated json file captures the same information as the POGO [8] generated xmi file (i.e., attributes, commands, and device/class properties). The fandango tool was used to export all of the configuration of the running DSH LMC into a single json file to use later on to generate a TANGO device simulator, using tango-simlib.

Tango-simlib is a data-driven library used for the development of TANGO device simulators. It uses configuration files that capture the device interface to generate a device simulator, making it easier for engineers to create simulators without having to write any code. With just an interface configuration file, a trivial simulator, with no behaviour, is created. Device simulator behaviour is captured in the Simulator Description Datafile (SimDD) and its accompanying override class module. The override class module is Python code. It defines the internal variables and the action handler methods that implement the simulator, thus allowing arbitrarily complex behaviour to be defined.

Much of the expected behaviour of the DSH LMC commands was captured in the TM-DSH Interface Control Document (ICD) and the dish movement behaviour was mostly copied from the MeerKAT antenna simulator.

### *KATCP/TANGO Translator*

Integration of the first TANGO based device into a simulated MeerKAT CAM system was done in [4]. This used a generic KATCP/TANGO translator which exposes the device's commands and attributes as KATCP requests and sensors, respectively. The schematics of the translator can be seen in Fig. 4 in that it has an in-built KATCP server device that the rest of the CAM system can connect to.

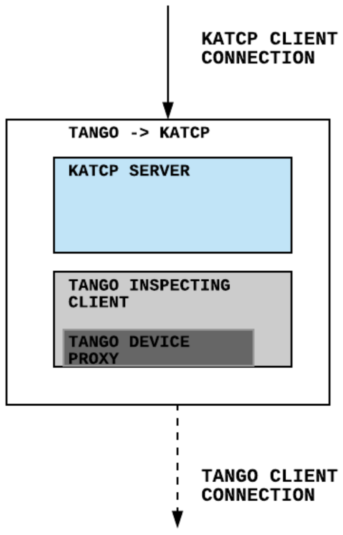

Figure 4: The KATCP/TANGO translator showing details of how it connects and communicates with a simple TANGO device

### *TANGOnised MeerKAT CAM Proxy*

The devices on the MeerKAT system are controlled through a layer of proxies. Using KATCP requests, the different proxies issue commands to control the underlying device. In order to control and monitor the prototype dish, another device proxy (DSH proxy) with the same high level KATCP interface had to be developed (see Fig. 5).

One of the objectives of this project was to develop a proxy for the SKA-MPI dish that would mimic the same behaviour as the MeerKAT receptor proxy. The DSH proxy has to expose the same set of sensors and higher-level requests to the rest of the CAM system so as to not break compatibility.

Most of the effort went into refactoring the existing MeerKAT receptor proxy so as to extract the common behaviour - this would form the basis for developing the DSH proxy. This common behaviour corresponded to the common MeerKAT subsytems (i.e. digitisers), that were leveraged for this qualification task. Moreover, the TANGO translator was used as device handler by the proxy to act as a TANGO client to the DSH LMC. A mapping of the dish modes and states had to be made to ensure the consistent

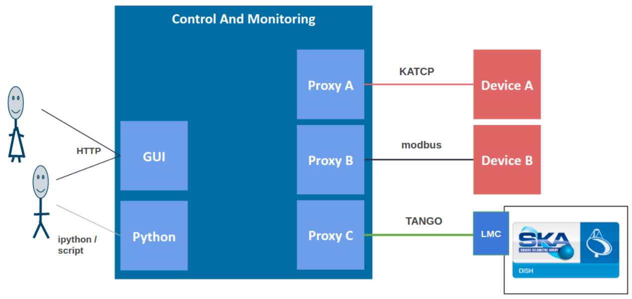

Figure 5: Different proxies communicating with respective underlying device using a specific protocol. Proxy (top) is the antenna proxy, Proxy B (middle) is the ancillary proxy and Proxy C (bottom) is the DSH proxy.

reporting from both the MeerKAT and SKA receptors. This also included having to map the DSH LMC commands to the proxy commands, and the attributes (not all) to some of the proxy sensors.

The DSH proxy also used the same implementation as the MeerKAT proxy to generate coordinates, apply pointing model corrections and the refraction correction when commanding the DSH LMC to simulate dish movements such as tracking and scanning.

To ensure that the DSH proxy had the same basic behaviour of the MeerKAT receptor proxy, it was put under a series of tests. These tests included running it against the real MeerKAT receptor observation scripts that are used in commissioning the receptors in the MeerKAT-RTS and also against our daily integration tests.

### *Integration Outcome*

Running the tests against the light-weight simulator served as an indication of confidence in the robustness of the DSH proxy ahead of the software readiness review meeting with the SE team and other stakeholders. That gave the team the go-ahead to integrate with the real DSH LMC software.

The integration effort of the MeerKAT-RTS CAM system and the DSH LMC component was a success. The first phase of the integration was done in the laboratory. There were some issues raised during the process, however the majority were resolved by the respective parties.

One of the issues that remains is that once the DSH LMC goes offline for an extended period of time the DSH proxy fails to re-connect with it as there are no more events being received. This may be a TANGO issue, or a problem in our translator.

## FUTURE WORK

Much of the integration testing was performed in the lab using the real SDQM LMC software prototype however with back-end simulators of the sub elements such as Dish Structure (DS), etc. There are plans in place later this year to have the final integration in the Karoo, with the LMC hardware and other components installed on the real SKA-MPI dish.

Further work is anticipated to happen at site. Six more SKA dishes are expected to be built there under the leadership of the SKA Organisation. These six dishes will form part of an Early Production Array (EPA), [9]. The EPA will be used to demonstrate a working array to allow engineers to identify any design and production defects before they go ahead with the full-scale production. This is will serve as an opportunity to integrate the other elements of the telescope and see how they work together.

There is also a plan to expand the MeerKAT array by 20 additional SKA dishes, [10]. This project will be jointly funded by the Max-Planck-Gesellschaft (MPG) and SARAO. This is intended to increase the capability of the MeerKAT array and the scope of scientific observations on MeerKAT whilst also keeping the SKA resources and facilities running during the SKA dormant period between the end of pre-construction and start of construction.

## CONCLUSION

Using the KATCP/TANGO translator, the DSH proxy exposed attributes from TANGO devices as KATCP sensors which could be monitored for the dish health. Also, KATCP requests were converted to TANGO commands for dish control and movement, and these were demonstrated with success during a software readiness review. The first integration effort with the LMC team also proved successful with a few modifications to be done by both teams based on encountered issues. The DSH proxy worked as intended and thus, provides the ability to drive both MeerKAT and the SKA-MPI dish as a single array.

## ACKNOWLEDGEMENTS

We would like to acknowledge the National Research Foundation (NRF), the Department of Science and Technology for awarding us the research grant to carry out this work and the team from Italian National Institute of Astrophysics (INAF), Osservatorio Astrofisico di Catania for collaborating with us during the integration session.

## REFERENCES

[1] T. Mauch *et al.*, "The 1.28 GHz MeerKAT DEEP2 Image", submitted for publication.

[2] SKA System CDR, `https://cdr.skatelescope.org/`

[3] N. Marais, "MeerKAT Control and Monitoring System Architecture", in *Proc. 15th Int. Conf. on Accelerator and Large Experimental Control Systems (ICALEPCS'15)*, Melbourne, Australia, Oct. 2015, pp. 247–250. `doi:10.18429/JACoW-ICALEPCS2015-MOPGF067`

[4] K. Madisa, N. Marais, A. J. T. Ramaila, and L. Van den Heever, "Integration of MeerKAT and SKA Telescopes using KATCP/Tango Translators", in *Proc. 16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'17)*, Barcelona, Spain, Oct. 2017, pp. 1964–1968. `doi:10.18429/JACoW-ICALEPCS2017-THSH201`

[5] TANGO, `https://www.tango-controls.org/`

[6] tango-simlib, `https://github.com/ska-sa/tango-simlib`

[7] fandango, `https://github.com/tango-controls/fandango`

[8] POGO, `http://www.esrf.eu/computing/cs/tango/tango_doc/tools_doc/pogo_doc/`

[9] Skatelescope, `https://www.skatelescope.org/news/first-ska-prototype-dish-assembled/`

[10] Department of Science and Technology, `http://pmg-assets.s3-website-eu-west-1.amazonaws.com/190911SKA.pdf`