INTEGRATION OF MeeRKAT AND SKA TELESCOPES USING THE KATCP/TANGO TRANSLATORS

K. Madisa*, L. van den Heever, N. Marais, A. J. T.Ramaila
SKA SA, Cape Town, South Africa

Abstract

The MeerKAT radio telescope control system uses the Karoo Array Telescope Control Protocol (KATCP) protocol and technology stack developed at Square Kilometer Array South Africa (SKA SA). The future SKA project chose the TANGO controls technology stack. However, MeerKAT and phase 1 of the SKA-mid telescope are intimately related: SKA-mid will be co-located with MeerKAT at the SKA SA Karoo site; the first SKA-mid prototype dishes will be tested using the MeerKAT systems; MeerKAT will later be incorporated into SKA-mid. To aid this interoperation, TANGO to KATCP and KATCP to TANGO translators were developed. A translator process connects to a device server of protocol A, inspects it and exposes an equivalent device server of protocol B. Client interactions with the translator are proxied to the real device. The translators are generic, needing no device-specific configuration. While KATCP and TANGO share many concepts, differences in representation fundamentally limits the abilities of a generic translator. Experience integrating TANGO devices into the MeerKAT and of exposing MeerKAT KATCP interfaces to TANGO based tools are presented. The limits of generic translation and strategies for handling complete use cases are discussed.

INTRODUCTION

The Square Kilometer Array (SKA) [1],a large multi radio telescope project is to be built in the co-hosting countries, Australia and South Africa. The telescope will have a total collecting area of approximately one square kilometer. The huge telescope will be made up of a collection of dishes, antennas and aperture arrays.

Currently the project is still in the design phase with construction scheduled to commence in 2019. The project will be developed over two phases - SKA1 and SKA2. SKA1 will be built over 2019 to 2027.

SKA1 will include two telescopes - SKA1 MID and SKA1 LOW - observing the Universe at different frequencies. South Africa will host the mid-frequency telescope, while Australia will be hosting the low-frequency telescope.

The MeerKAT (Karoo Array Telescope) telescope [2] is a interferometric radio telescope. SKA-SA is coordinating its construction in the Karoo, Northern Cape, South Africa. It is regarded as a precursor to the SKA radio telescope. On its completion, it will contain 64 receptors. It will be integrated into the SKA1 MID telescope - more details on the status of project is available [3].

The rest of the paper is will delve into details about the different technology stacks that the two telescope system use, with more focus on the MeerKAT telescope and how it was modified to accomodate these generic translators and the importance of these translators in the SKA project.

TANGO CONTROLS FRAMEWORK

The TACO Next Generation Objects (TANGO) was chosen as the SKA phase 1 common framework for all the telescope's element LMCs (Local Monitoring and Control). It was considered the most promising framework to meet the SKA requirements after thorough comparative tests between several major control systems such the Experimental Physics and Industrial Control System (EPICS), and the ALMA Common Software (ACS).

TANGO control system [4] is a free open source device oriented controls toolkit for controlling any kind of hardware or software and building SCADA (Supervisory Control and Data Acquisition) systems. It is a distributed system that uses two network protocols - the omniorb implementation of CORBA (Common Object Request Broker Architecture) and ZeroMQ.

TANGO is being developed in collaboration between several research institutions primarily including; ESRF (European Synchrotron Radiation Facility), SOLEIL (Soleil Synchrotron), ELETTRA (Elettra Synchrotron), and ALBA (Alba Synchrotron), with their main goal being to guarantee the successful development of TANGO.

KATCP PROTOCOL

The MeerKAT radio telescope control system uses the Karoo Array Telescope Control Protocol (KATCP). KATCP is a communications protocol based on top of the TCP/IP (Transfer Control Protocol/Internet Protocol) layer [5]. It is a syntax specification for controlling devices over a TCP or RS-232 link. It is the preferred remote command and control interface for open source FPGA (Field-Programmable Gate Array) platforms (e.g. ROACH and SKARAB also used by KAT-7/MeerKAT) developed by the CASPER (Collaboration for Astronomy Signal Processing and Electronics Research) collaboration [6].

KATCP was developed by SKA South Africa in the CASPER collaboration. KATCP is open-sourced and has been adopted by a few projects that use the open source FPGA platforms.

The KATCP protocol is specified as the MeerKAT Control and Monitoring (CAM) interface [7] for all subcontracted and internal hardware devices and subsystems, as well as internal communication between CAM components. In cases where the subcontractor cannot deliver a KATCP interface, a device translator is implemented by the CAM team to trans-

* kmadisa@ska.ac.za

late its specific protocol (e.g., modbus, OPC, web-services, Ganglia metrics) to KATCP.

INTEGRATION OF THE TANGO DEVICE

As part of the prototyping effort in developing the LMC interface simulation framework [8], a simple TANGO based simulator was implemented. The objective was to investigate how the TANGO device simulator could be incorporated in the existing fully functional KATCP based MeerKAT CAM environment.

As a proof of concept, a weather device simulator was implemented. The device simulator was configured to match the functionality of the existing MeerKAT weather station. In order to aid the interoperability of the TANGO device simulator in the MeerKAT radio telescope system, a TANGO/KATCP translator was developed.

The TANGO/KATCP Translator

The translator is generic, it has no precoded idea of the TANGO device it is translating. The translator is made up of two major components. The first component is a *tango_inspecting_client wrapper*. This wrapper consists of a TANGO *DeviceProxy* component. The TANGO *DeviceProxy* object is responsible for making client connections to the TANGO devices. It is a high level class which provides the client with an easy-to-use interface to TANGO devices. The second component is the KATCP server.

The translator connects to the TANGO device as a client and interrogates the device for TANGO attributes and commands, and exposes them over a KATCP server interface as KATCP sensors and requests, respectively. The translator allows a KATCP client to communicate with a TANGO device. Once the KATCP server inside the translator has been initialized, to remain synchronised with TANGO device, the TANGO *DeviceProxy* sets up attribute polling and subscribes to most, if not all, the attribute events.

The MeerKAT CAM Architecture

The MeerKAT CAM architecture is divided into three layers of components as shown in Figure 1 with connections strictly from higher to lower levels, and a fourth category of CAM controller components that collaborate as models and controllers in the Model-View-Controller software architectural pattern.

Apart from the katportal clients that use http and websockets, all connections are made using KATCP over TCP/IP; each directed arrow starts at a client making a TCP connection to a KATCP server running at a well known IP and TCP port [9]. For subsystems external to CAM the IPs and ports are assigned by the System Engineering team, while CAM components have IPs and ports assigned internally by the Control and Monitoring team within the CAM IP range.

The hardware devices and the MeerKAT subsystems are protected from direct access through a layer of proxies implemented by the CAM subsystem. All the system components

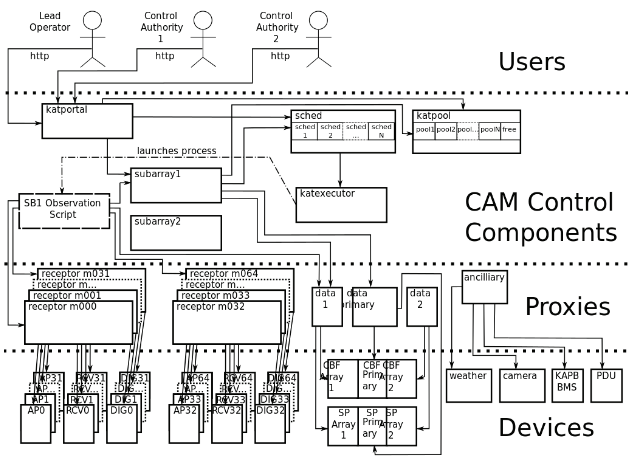

Figure 1: CAM Layers and Observation Control.

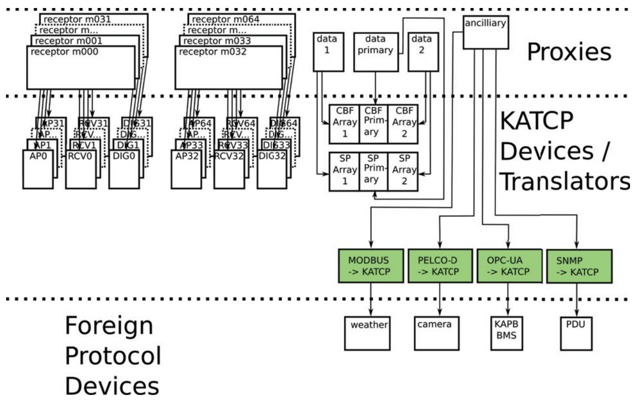

Figure 2: The MeerKAT CAM proxy and device/translator layer.

connect via the proxy layer and not directly to hardware devices/subsystems.

At the lower level is the device layer which may comprise real hardware or simulators. For devices that do not have a KATCP interface, a CAM device translator instance is implemented by the CAM subsystem as shown in Figure 2.

The CAM subsystem implements a fully simulated system up to the KATCP interface of each hardware device and subsystem [7]. It is possible to run a configuration including only simulated devices, or any combination of real and simulated devices combined. This allows full software development, unit testing and integration testing, including CAM subsystem qualification testing without dependency on the availability of hardware.

Modifying the MeerKAT CAM Configuration

Before a TANGO device can be started, it needs to be registered in the TANGO configuration database (with a MYSQL backend). The TANGO configuration database is the central registry for all the TANGO devices that make up a specific control system instance. Once the device is registered, the device is exported by its TANGO device server. It is then dynamically assigned a unique port to listen to.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH201

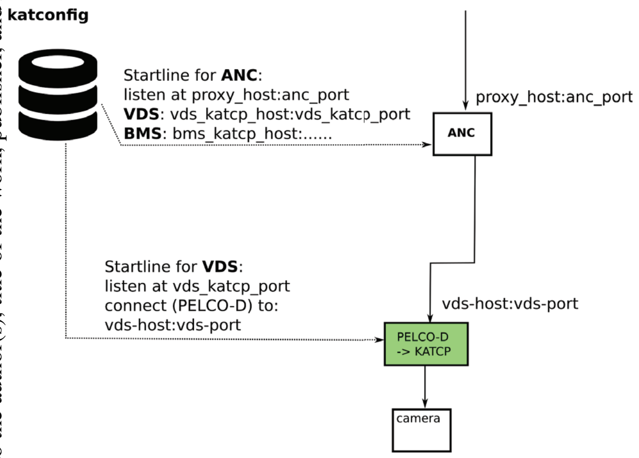

Figure 3: The MeerKAT CAM system configuration.

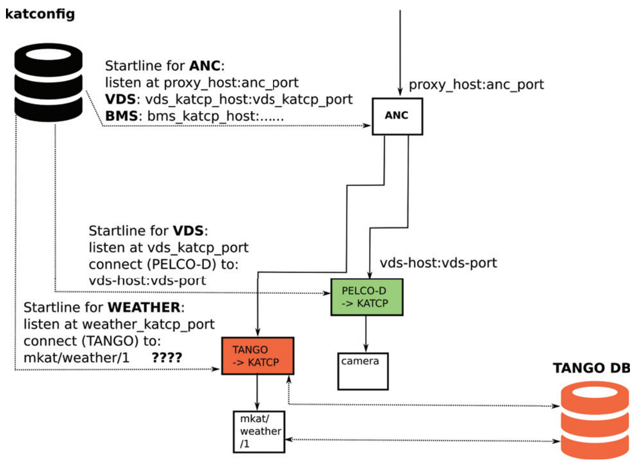

Figure 4: The MeerKAT CAM system configuration change with the TANGO device and translator..

The MeerKAT CAM keeps its own configuration for the telescope system (katconfig and katcamconfig) as shown in Figure 3. This is a run-time static configuration. It maintains all of the CAM components and telescope subsystems IPs and ports.

To work around these differences both methods were used. The information required to start up a TANGO device server (the device server name, server class name, and device and class properties) were added to the CAM configuration. During system startup katconfig generates command-line instructions on how to launch a server processes, which IP and port number it should listen to, and which lower-level devices it should connect to. These processes are then launched by a supervisory service on each compute node in the CAM system.

Launching the TANGO/KATCP Translator + Device

After configuring the CAM system, the next step was integrating the translator into the simulated telescope system launch. The command-line instructions for the weather simulator ensures that TANGO device is registered in the TANGO DB before the server process is started as shown in Figure 4. The MeerKAT system has no direct understanding of the TANGO DB and manages system interconnections through the command line parameters when starting the telescope processes. MeerKAT has its own TCP port allocation method, which could conflict with the TANGO system's automatic port allocation. For this reason the command-line instructions for starting up the TANGO device requires a –port flag to be passed, controlling the TCP port where the TANGO device server will listen.

Once the TANGO device is up and running, the translator command-line instructions tell the *DeviceProxy* to which TANGO device server to connect to. Using the name of the device, the *DeviceProxy* queries the database for the IP and port number where the weather simulator device is running and the translator connects to its device. Figure 5 show the successful integration of the TANGO/KATCP translator

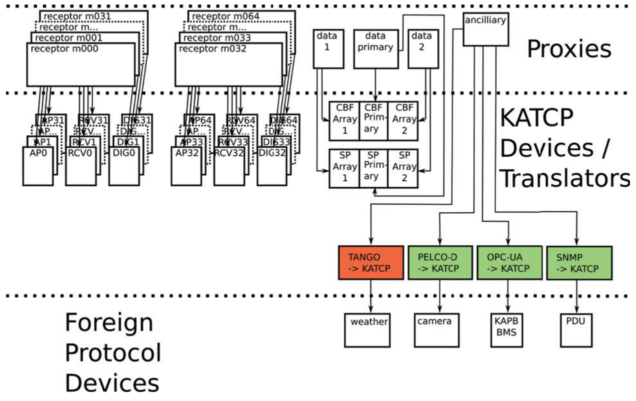

Figure 5: The MeerKAT CAM proxy and device translator layer with the TANGO/KATCP translator and the TANGO device

together with the TANGO weather simulator once the system is done starting up.

Issues With the Generic Translator

Although the translator does perform the desired objective, it still has some limitations. It does not handle TANGO commands that take or return arrayed values, however this can be handled by simply using kattype with multiple=True. The other problem with the translator is that it only supports scalar attributes. KATCP does not define how sensors with 1-D or 2-D arrayed values should be handled. The translator does not know to map the reply-informs in KATCP to TANGO. At the moment we have not yet implemented the a way to handle a TANGO device that dynamically changes its attributes and/or commands.

EXPOSING MEERKAT KATCP INTERFACE TO TANGO BASED TOOLS

Continuing with the LMC interface simulation framework [8], the next task was to evaluate some of the functionalities of the TANGO tools in the context of a real telescope

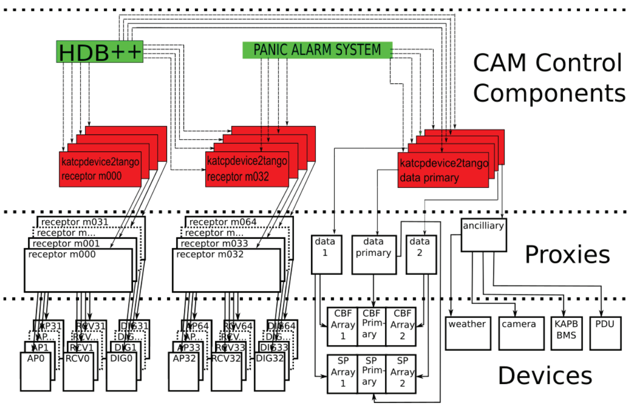

Figure 6: The MeerKAT CAM proxy and device translator layer showing the use of the TANGO tools in MeerKAT system.

system. The TANGO community is growing quickly and the development of the tools is supported by many research institutions and industry.

The idea was to have to try out some of these tools features to see if they could meet some of the MeerKAT requirements. The tools that were chosen was the Historic Database++ (HDB++) [10] and the PANIC Alarm System [11].

The major output from this work was the development of the KATCP/TANGO translator and evaluation reports on the tools [12–14]. This TANGO translator enabled the process of pluging in these two TANGO based tools in the MeerKAT CAM system. The procedure for installing and configuring the translator is similar to the way we installed the TANGO/KATCP wrapper.

Connecting the TANGO Tools

The KATCP/TANGO translator is made up two major components, the *ktcp_inspecting_client* and the TANGO device server. The *katcp_inspecting_client* interrogates the KATCP device simulator, or in some cases, a real KATCP hardware device for its sensors (monitoring points) and requests. The *katcp_inspecting_client* gathers all this information and populates data structures describing the sensors and requests. The *katcp_inspecting_client* makes use of the KATCP client instance to establish a connection with the KATCP device/simulator.

The TANGO device server component in turn exposes these KATCP sensors and requests as TANGO attributes and commands, respectively. This component also sets up sensor event sampling strategies to keep the TANGO device server synchronized with the KATCP device.

Issues With the Translator

At the moment, we have not implemented a way to map the KATCP discrete data type to the equivalent TANGO DevEnum type. We had to find a way around to map the KATCP sensor status values to only a selected few values of the TANGO attribute quality, as there was no direct mapping.

FUTURE WORK

The translators are still a work in progress. There are plans in place to make improvements support commands with array input/output. The other feature to be implemented is to handle TANGO device interface changes, this feature has already been added to the new PyTango version 9.2.5 [15]. The open-sourcing of this library is in the pipelines and once the improvements are complete it will be made available to the public.

The SKA-mid Dish Prototype

There is a plan in place to have the SKA-mid prototype dishes installed and tested on the MeerKAT telescope as part of the SKA Dish Qualification Model (SDQM). Currently the Dish foundation has been laid as construction is underway in the Karoo to have the Dish pedestal ready for the Dish structure.

The SKA DSH Consortium will build, integrate and qualify the SDQM on site in the Karoo. The integration and on-site qualification scheduled take place around the second quarter of 2018.

The SKA SA CAM role in this project will be of a supporting nature to the SKA DSH Consortium in qualifying the SDQM. It will make use of the existing MeerKAT Receptor Test System (RTS) [9].

This, once the DSH LMC simulator has been developed, will entail the improvement/development of the KATCP/TANGO translator for the MeerKAT RTS and the SDQM LMC.

CONCLUSION

The TANGO weather device simulator was successfully integrated into the MeerKAT CAM system. The changes made to the system's configuration makes it easier to launch a TANGO device together with its translator. The TANGO weather simulator has show to be a drop-in replacement for the MeerKAT KATCP weather simulator.

The TANGO to KATCP translator has proven to be working well. It has shown that integrating more TANGO devices into the MeerKAT system seems plausible.

ACKNOWLEDGEMENT

We would like to acknowledge the National Research Foundation (NRF), the Department of Science and Technology, the Government of India for awarding us the joint research grant to carry out this work under the Indo-South African Flagship Programme in Astronomy.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THSH201

REFERENCES

[1] *SKA*. Oct. 2017. http://skatelescope.org

[2] *SKA-SA*. Oct. 2017. https://www.ska.ac.za

[3] L.R.S. Brederode and L van den Heever. 'MeerKAT: Project Status Report'. In: *16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'17), Barcelona, Spain, this conference*. JACOW, Geneva, Switzerland. Oct. 2017.

[4] *TANGO*. Oct. 2017. https://www.tango-controls.org

[5] *KATCP*. Oct. 2017. https://www.pythonhosted.org/katcp

[6] *CASPER*. Oct. 2017. https://casper.berkeley.edu/wiki/KATCP

[7] L van den Heever. 'MeerKAT control and monitoring-design concepts and status'. In: *SKA SA, October* (2013).

[8] A.J.T. Ramaila et al. 'Control System Simulation Using DSEE High Level Instrument Interface And Behavioural Description'. In: *16th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'17), Barcelona, Spain, this conference*. JACOW, Geneva, Switzerland. Oct. 2017.

[9] N. Marais. 'MeerKAT Control and Monitoring System Architecture'. In: *15th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'15), Melbourne, Australia*. JACOW, Geneva, Switzerland. Oct. 2015, pp. 247–250.

[10] *HDB*. Oct. 2017. http://www.tango-controls.org/community/projects/hdbplus/

[11] *Panic*. Oct. 2017. http://www.tango-controls.org/community/projects/panic-alarm-system/

[12] *TANGO Evaluation: Historic Database plusplus*. Oct. 2017. https://docs.google.com/document/d/1qgpn54w3Igs-lmFWsv2Nusuo3NTcfclAM-3Q17yXeYQ/edit?usp=sharing

[13] *TANGO Evaluation: Panic Alarm System*. Oct. 2017. https://docs.google.com/document/d/1X3UB5-zLQKBnWjJ-WEQnXyaW-ggtgPz0I8fAGZraZII/edit?usp=sharing

[14] *TANGO Evaluation: TANGO Database*. Oct. 2017. https://docs.google.com/document/d/1DSBNdPhcnkD80FpoE4eHzONlgNDF_MZoJP5nLwaR7ik/edit?usp=sharing

[15] *PyTango*. Oct. 2017. https://github.com/tango-controls/pytango