ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-THMG012

A MULTI-LEVEL MONITORING INTERFACE FOR THE SKA CENTRAL SIGNAL PROCESSOR USING THE TARANTA SYNOPTIC VIEW

G. Marotta*, C. Baffa, E. Giani, INAF-OAA, Firenze, Italy
V. Alberti, INAF-OATs, Trieste, Italy
M. Canzari, INAF-OAAb, Teramo, Italy
G. Brajnik, University of Udine and IDS, Udine, Italy

Abstract

Graphical user interfaces (GUIs) play a critical role in the operation and maintenance of large-scale distributed control systems. In this work, we present a synoptic-based visualization for the Central Signal Processor (CSP) of the SKA (Square Kilometre Array) telescope, developed using Taranta, a web-based visualization tool for TANGO systems. The synoptic view provides an intuitive, multi-level representation of CSP, from the upper-level control managed by CSP.LMC down to individual data processing subsystems, including CBF, PSS, and PST devices. The synoptic diagrams are created using Inkscape and exported as SVG files, enabling flexible and straightforward integration with Taranta. A key focus of this work is the collaboration between different subsystem teams, which was essential to accurately model and visualize the full CSP hierarchy. We also define a strategy to validate the effectiveness and the usability of the GUI, ensuring it delivers tangible value in improving the monitoring and troubleshooting capabilities of complex control systems such as SKA.

INTRODUCTION

The Square Kilometre Array (SKA) is currently under construction and represents the most ambitious radio telescope ever built, both in scale and in scientific potential. When finalised, it will consists of two large interferometers located in South Africa and Australia, that will search the cosmos at Mid (350 MHz – 15.3 GHz) and Low (50 MHz – 350 MHz) frequencies ranges respectively [1].

With the first milestones already achieved and the system progressing through its commissioning phase, the project is now transitioning from the earliest integration stage, known as Array Assembly 0.5 (AA0.5), toward a gradual scale-up of its full capabilities. In this context, Graphical User Interfaces (GUIs) play a crucial role as tools to support the testing and commissioning of software components prior to their deployment in production. They provide developers with control panels and monitoring screens that deliver timely feedback during exploratory testing sessions, helping to identify unexpected code behavior. At the same time, they serve Assembly, Integration, and Verification (AIV) teams as collaborative platforms for synchronizing and validating system functions. Furthermore, the feedback from this early user experience will support the development of interfaces tailored for operators and maintainers, facilitating their work with the telescope during astronomical observations.

UI development has also proven to be an effective tool for communication between teams working on different software components, as it encourages the exchange of knowledge, discussions about interfaces, and clarifications of code behavior. In previous phases, collaborative GUI development was successfully adopted with limited telescope configurations, demonstrating its value for both technical validation and team coordination. [2, 3].

THE CENTRAL SIGNAL PROCESSOR

At the core of the SKA lies a highly complex computing system designed to process and reduce, in real time, the immense flow of data generated by receptors. The Central Signal Processor (CSP) constitutes a sophisticated computing infrastructure within the project, integrating multiple software and hardware components to enable the first stage of raw antenna data reduction [4].

The main design of the CSP is shared between the two interferometers (Mid and Low) and comprises the following:

- CSP.LMC (Local Monitoring and Control): the CSP's control entry point, which interfaces with the Telescope Manager and orchestrates control over the entire system.
- CBF (Correlator and Beam Former): processes raw antenna signals into scientifically usable data streams.
- PSS (Pulsar Search): an engine dedicated to identifying new pulsars and transient signals.
- PST (Pulsar Timing): optimized for precise timing of known pulsars.

In this context, CBF, PSS, and PST represent the subsystems dedicated to data reduction and analysis, while CSP.LMC coordinates requests received from the higher-level Telescope Monitoring and Control (TMC) system and provides responses on behalf of the entire CSP.

A fundamental feature of SKA operations is the ability to partition telescope resources, such as antennas and computing hardware, into subarrays, allowing up to 16 simultaneous and independent observations or technical/engineering activities. To achieve this, a large portion of CSP resources must operate flawlessly in parallel, and any error or disruption must be detected and resolved as quickly as possible.

For these reasons, the design and implementation of GUIs must be regarded as a comprehensive and integral part of the scaling process toward the full implementation of CSP

* gianluca.marotta@inaf.it

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-THMG012

system. Their development should enable testers and developers to rapidly identify and expose anomalies and bugs.

As the telescope evolves to support its maximum capacity of 16 subarrays, UIs will not only remain essential for developers and AIV teams but will also become key enablers of efficient operations, system reliability, and collaborative workflows across the international project.

VISUALIZING SKA SOFTWARE COMPONENTS: TARANTA SYNOPTIC VIEW

Software components in the SKA control system are implemented as TANGO devices. TANGO Controls [5] is widely adopted in large-scale scientific facilities, designed to provide a standardized architecture for device communication, monitoring, and control.

Part of the Tango ecosystem is Taranta, a web-based, no-code platform that allows for the creation of intuitive and accessible GUIs for monitoring and controlling TANGO-based systems [6]. Taranta is based on a drag and drop mechanism: users with no experience in web development can quickly select the appropriate widgets, position them on a canvas and connect them to the relevant Tango devices, easily creating their own GUI.

The synoptic view concept plays a central role in control systems, providing an intuitive and comprehensive visualization of the system's state and functions. Taranta implements this concept by allowing the display of SVG files in its dashboards. The characteristics of vector files—such as resolution independence and editability, along with the availability of highly usable free software for editing them —make this file type an ideal choice for synoptic view dashboards.

Taranta documentation recommends Inkscape [7] as a software to create svg files and gives clear guidance on how to configure it to best exploit synoptics functionalities. SVG elements can be linked to TANGO device states and attributes by configuring the object properties in Inkscape. In particular, two main possibilities are supported:

1. **Text boxes** can be configured to display the value of a specific attribute from a specific TANGO device;

2. **SVG layers** can be configured to be shown or hidden based on logical conditions applied to TANGO attributes, with CSS configurations also modified under the same conditions.

In addition, the use of layers enables another key feature of Taranta's synoptic view: **zoom levels** in SVG. Zoom levels provide dynamic scaling and selective visibility of SVG elements, creating an interactive user experience. As users zoom in and out, different parts of the SVG are displayed, allowing a single view to present multiple layers of information with varying levels of detail.

DASHBOARD DEVELOPMENT PROCESS

The possibilities enabled by the new Taranta synoptic view not only enrich the visualisation capabilities to explore the control system but also suggest a structured, collaborative and iterative process for the development of GUIs within the SKA project.

The workflow followed for this study greatly benefits from SKA Taranta and CSP.LMC developers belonging to the same Agile team. This arrangement has always fostered close collaboration between the two groups, with CSP.LMC developers also acting as early adopters and beta testers of new Taranta functionalities [8].

The process is often triggered by CSP.LMC developers producing a new dashboard or, when additional functionalities are needed to realise the desired effect, mock up drawings. This phase is critical to systematically collect feedback and identify missing functionalities and improvements, which are carefully documented and prioritised for future development cycles.

In addition to this core interaction, UI/UX experts play a key role in coordinating the process by providing feedback on design guidelines, usability, and consistency across interfaces. After this, the involvement of teams developing CSP subsystems ensures that the dashboards integrate domain-specific requirements and provide detailed operational information.

THE SYNOPTIC VIEW FOR CSP SUBARRAYS

General View

In Fig. 1 we present the first mock-up of a comprehensive dashboard for monitoring up to 16 independent observations or technical operations, carried out via subarrays at the CSP.LMC level. The dashboard is organized into a 4×4 grid, with each box reporting details on a single subarray.

The purpose of this view is to provide the user with an immediate impression of the status of the different CSP subarrays. Although the dashboard may appear information-dense, its design highlights the primary features of each subarray and facilitates comparisons across the 16 instances.

The most immediate information is the distinction between active and inactive subarrays, i.e., those currently engaged in an observation versus those that are not. In the software interface, this status is determined by the attribute *ObsState*, which reflects a finite state machine describing the observational state of each subarray. For instance, a subarray's ObsState is IDLE when resources have been assigned, READY when it is configured, and SCANNING when recording observational data. The default value EMPTY indicates that no observational procedure has been initiated for that instance.

In the synoptic view, inactive (EMPTY) and active (non-EMPTY) subarrays are differentiated by borders and color coding. Active subarrays are shown in yellow, inactive ones in light blue. Additionally, active subarrays are given an

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-THMG012

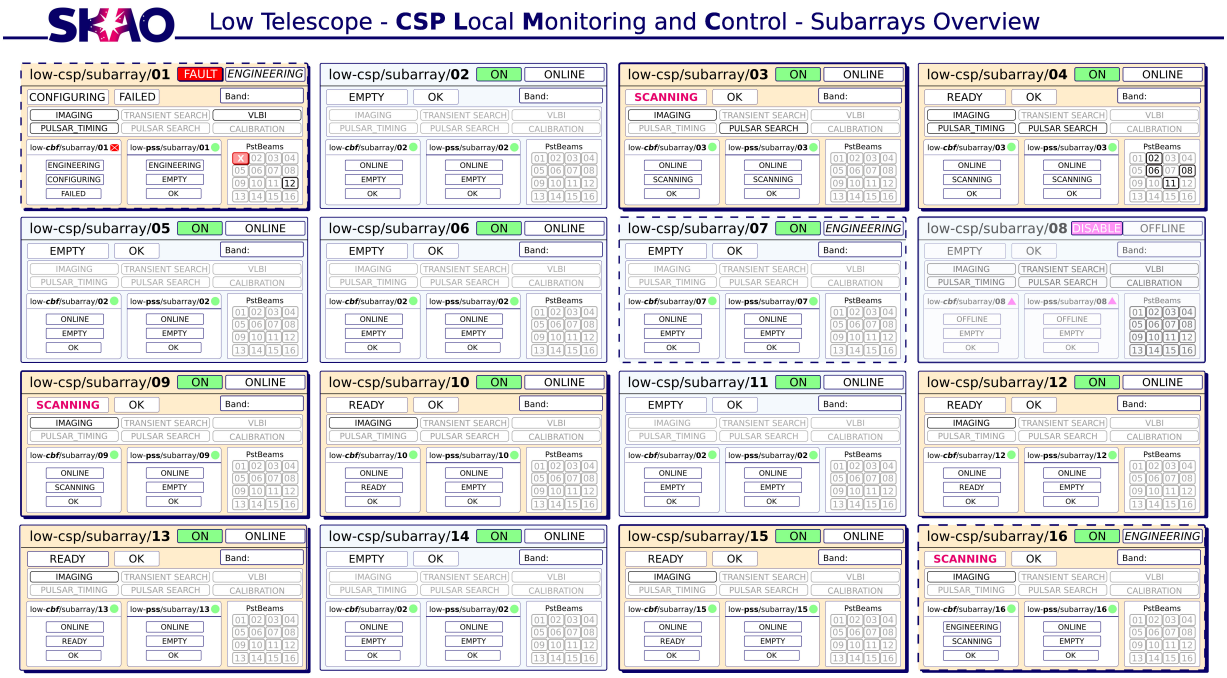

Figure 1: Mock-up of the synoptic view dashboard for CSP.LMC.

embossed effect to enhance visibility, including for color-blind users.

Another key piece of information, to be visible at a glance, is the attribute *AdminMode*, which indicates the administrative status of the subarray. The value ONLINE means the subarray is correctly connected to its subsystems, while OFFLINE indicates it is not. A particular case is ENGINEERING, which denotes that the subarray is operational but reserved for maintenance or engineering activities.

In the mock-up CSP GUI, these states are visually distinguished by border styles and color treatments: continuous borders indicate ONLINE, dashed borders indicate ENGINEERING, while OFFLINE subarrays are displayed in faded colors to highlight their unavailability until connection to subsystems is activated.

All these layout changes can be activated through conditional assertions, which control the visibility of different layers, as supported by Taranta's synoptic view (see Section 2).

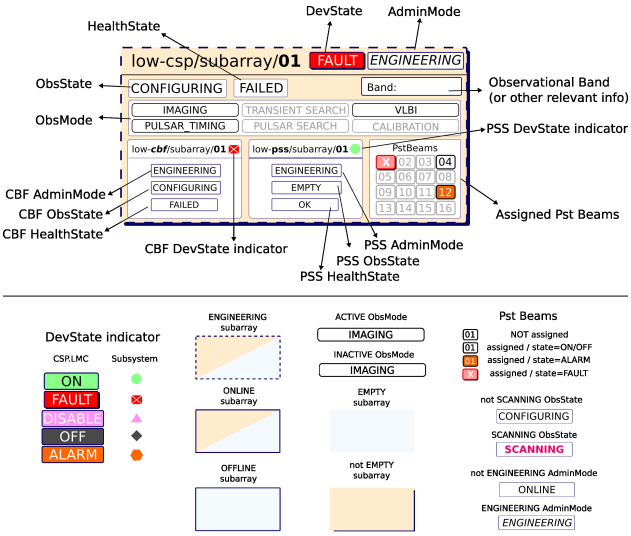

Figure 2: Example of a box reporting information about one subarray, with explanation and legend.

Subarray View

Each of the 16 subarray boxes reports all high-level information related to the corresponding subarray. Figure 2 provides an example of a single subarray view, accompanied by a legend to aid interpretation.

At the top of each box, the unique identifier of the CSP.LMC device is displayed, namely the Tango Resource Locator (TRL). For the CSP.LMC of the SKA-Low telescope, this takes the form "low-csp/subarray/<sub_id>", where sub_id ranges from 01 to 16. At the same level, the device's *DevState*—a native TANGO attribute that represents its internal state—is also reported. This TANGO attribute describes the operational state of the device and is color-coded to facilitate comparison across subarrays in the collective view. Since color coding alone can be difficult to interpret for color-impaired users, the FAULT state is additionally emphasized through an box with shadow, making it more easily identifiable at a glance. Alongside the DevState, the AdminMode is displayed, with the ENGINEERING state highlighted in italic font.

Beneath this top-level information, additional states and modes are reported, including the *ObsState* (previously described), the *HealthState* (an important indicator of the

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-THMG012

reliability of the software component), the *Observational Band*, and a summary of the *ObsMode* values associated with the component. The ObsMode attribute (which is different from the ObsState attribute) defines the observational mode assigned to the subarray during configuration. It can include up to six possible values — IMAGING, TRANSIENT_SEARCH, PULSAR_SEARCH, PULSAR_TIMING, VLBI, and CALIBRATION — in any combination. To improve readability and facilitate comparisons across subarrays, all possible values are displayed in a fixed 2×3 grid. Modes not currently assigned appear faded, while active modes are highlighted, making the selected configuration immediately visible. This layout also enables users to quickly identify ObsMode values by position, simplifying cross-comparison among multiple subarrays.

Below the ObsMode section lies the area dedicated to CSP subsystems other than CSP.LMC, which are directly controlled by the corresponding LMC device. Starting from the left, two identical boxes are provided for the CBF and PSS subarrays. At the top of each box, the TRL of the respective TANGO subsystem device is reported ("low-cbf/subarray/<sub_id>" and "low-pss/subarray/<sub_id>", respectively), together with a small colored symbol representing the device's DevState. The legend in Fig. 2 explains the association between states and symbols. As in other parts of the dashboard, shape variations are used in addition to color coding to improve accessibility for color-blind users. Within the body of each box, the device's main attributes are displayed, namely AdminMode, ObsState, and HealthState.

To the right of the PSS subarray section lies the PstBeams section. The PST subsystem differs architecturally from the others in that it does not include an abstract software entity representing the subarray. Instead, PST resources—i.e., the beams—are dynamically allocated to the CSP.LMC subarray, ranging from 0 up to 16 instances. To indicate which beams are assigned to a specific CSP.LMC instance, the same approach used for ObsMode is adopted: a 4×4 grid displays all 16 pst_id numbers, with assigned beams highlighted and the others shown in a faded style.

Each beam is itself a TANGO device with its own states and modes. However, at this level only failed or alarmed beams are emphasized, using codes explained in the Fig. 2 legend. Again, accessibility for color-blind users is taken into account: failed beams are not only color-coded but also marked with an "X," distinguishing them clearly from alarmed ones.

Subsystem View

The concept of the CSP synoptic view is based on the understanding that the information from the entire instrument cannot be effectively represented within a single user view. Even if such a representation were possible, it would often include redundant details, unnecessarily increasing information density and thereby reducing the readability and usability of the GUI. Nevertheless, access to this detailed information becomes crucial when issues arise, such as failures or unexpected behaviors. Both during the testing and commissioning phases, as well as in production, the ability to drill down into the details of the data-reduction subsystems (CBF, PSS, and PST) is essential for diagnosing and preventing bugs and anomalies.

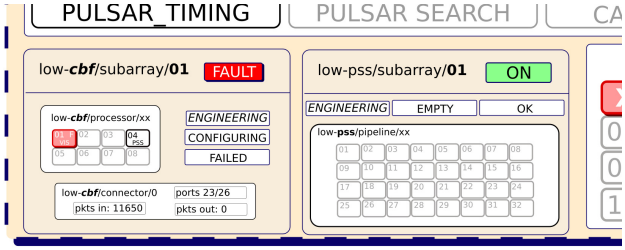

Figure 3: A possible zoom in the CBF and PSS sections, revealing other details hidden in the general view.

In this context, the zoom levels of SVGs represent a key feature for drilling down into the system. When additional information is required, users can zoom into the corresponding subsystem box, revealing a new layer with more detailed information about the subsystem. An illustrative example is provided in Fig. 3 for the CBF and PSS subarray. The same could be done also for PST beams. If necessary, the zoom can also be applied at deeper levels, enabling progressively more detailed exploration.

For defining the zoomed levels of the subsystem dashboards, it is essential to involve the subsystem developers in the GUI design process, as they are the most accountable for selecting the information to be displayed.

DASHBOARD USAGE AND VALIDATION

As anticipated, such a synoptic dashboard can be useful at multiple levels, ranging from the development process to potential support for operators during actual observations. In recent years, CSP.LMC developers have reported significant benefits from using self-developed TARANTA GUIs to better understand code behavior and to identify anomalies not detected through standard CI/CD testing. Building on this experience, Taranta dashboards have since become a consistent part of the exploratory testing process.

Exploratory testing is a software testing approach in which testers simultaneously learn, design, and execute tests, relying on investigation and adaptability rather than predefined scripts. In other words, developers sit together and perform live tests, after agreeing on specific behaviors to be checked. In this process, the availability of accurate and reliable GUIs is crucial, as they provide the means to observe system states, validate expected behaviors, and quickly identify anomalies. At the same time, exploratory testing has also proven to be a valuable source of feedback for validating the GUI itself, often providing clear indications on how the dashboards can be improved.

Although the CSP synoptic dashboard is still in its early stages, we propose the following process for its usage and validation:

ISBN: 978-3-95450-255-4 ISSN: 2226-0358 doi: 10.18429/JACoW-ICALEPCS2025-THMG012

1. The dashboard is employed in a series of exploratory testing sessions led by CSP.LMC developers, with the participation of Taranta developers, subsystem developers, and SKA UI experts.
2. Feedback is collected and the dashboard refined by adding missing information and improving readability.
3. The dashboards are then provided to the AIV teams, who conduct their own test sessions with the support of CSP developers.
4. Additional feedback is collected and incorporated into further dashboard improvements.

The main characteristic of this process is its iterative nature, which aligns well with the Agile framework adopted for SKA software development. Taranta is particularly well suited for such an approach, as its no-code and user-friendly environment enables back-end developers to create and refine dashboards independently.

This process resembles the system release workflow and it closely aligns with Lean UX, an approach that combines product and interface design into iterative cycles [9]. It reinforces the idea that GUIs should be regarded as an integral, rather than secondary, component of the released product.

CONCLUSIONS

The development of the CSP synoptic dashboard demonstrates how graphical user interfaces may become core components of the SKA software ecosystem, well beyond their traditional role as auxiliary visualization tools. By leveraging Taranta's synoptic view and SVG-based zoom levels, the dashboards can provide developers, AIV teams, and eventually operators with a flexible and intuitive means of monitoring and interacting with the CSP subsystems.

The experience gained so far shows that GUIs are not only valuable for daily operations but also play a critical role during testing, commissioning, and throughout the development cycle itself. Their integration into exploratory testing workflows has proven effective in uncovering anomalies, validating system behavior, and improving communication between developers and subsystem teams. Moreover, the dashboards themselves benefit from this process, evolving iteratively through user feedback to better meet technical and usability requirements.

The proposed iterative validation cycle—where dashboards are designed, tested, refined, and re-tested—aligns naturally with the Agile methodology adopted within the SKA project. Taranta's no-code and user-friendly environment facilitates this approach, allowing developers to take ownership of their GUIs and adapt them quickly to emerging needs.

Ultimately, this work highlights the importance of treating GUIs as first-class elements of the SKA software infrastructure. Their careful design, iterative refinement, and integration into development, validation, and operational workflows will be essential not only for ensuring reliable CSP operations and efficient collaboration among international teams, but also for supporting the ongoing scale-up of the SKA system toward its full capacity.

REFERENCES

[1] SKA Observatory, http://www.skao.int

[2] V. Alberti *et al.*, "Leverage collaborative design techniques to develop a unified interface for the SKA central signal processor", in *Proc. SPIE 13101, Software and Cyberinfrastructure for Astronomy VIII*, Yokohama,Japan, Jun. 2024, paper 130141. doi:10.1117/12.3021688

[3] V. Alberti *et al.*, "A lean UX approach for developing effective monitoring and control user interfaces: a case study for the SKA CSP.LMC Subsystem", in *Proc. ICALEPCS'23*, Cape Town, South Africa, Oct. 2023, paper FR2BCO02, pp. 1650-1656, doi:10.18429/JACoW-ICALEPCS2023-FR2BCO02

[4] G. Marotta *et al.*, "Software design for CSP.LMC in SKA", in *Proc. SPIE 12189, Software and Cyberinfrastructure for Astronomy VII*, Montreal, Canada, Jun. 2022, paper 121891A, doi:10.1117/12.2630140

[5] Tango Controls, https://www.tango-controls.org/

[6] Taranta Docs, https://taranta.readthedocs.io/en/latest/

[7] Inkscape Website, https://inkscape.org/

[8] M. Canzari *et al.*, "Enhancing feature development and user satisfaction in SKA: a beta testing approach within the Cream team", in *Proc. SPIE 13101, Software and Cyberinfrastructure for Astronomy VIII*, Yokohama, Japan, Jun. 2024, paper 131014J, doi:10.1117/12.3022097

[9] J. Gothelf and J. Seiden, *Lean UX: Designing Great Products with Agile Teams*. Sebastopol, CA, USA: O'Reilly Media, 2016.