ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA188

THE SKA DISH LOCAL MONITORING AND CONTROL SYSTEM USER INTERFACE

A. Marassi†, INAF-Astronomical Observatory of Trieste, Trieste, Italy
M. Brambilla, Politecnico di Milano, Milano, Italy
A. Ingallinera, S. Riggi, C. Trigilio, INAF-Catania Astrophysical Observatory, Catania, Italy
G. Nicotra, INAF IRA, Bologna, Italy
G. Smit, SKA South Africa, Cape Town, South Africa

Abstract

The Square Kilometre Array (SKA) project is responsible for developing the SKA Observatory, the world's largest radiotelescope ever built: eventually two arrays of radio antennas - SKA1-Mid and SKA1-Low - will be installed in the South Africa's Karoo region and Western Australia's Murchison Shire, each covering a different range of radio frequencies. In particular SKA1-Mid array will comprise 133 15m diameter dish antennas observing in the 350 MHz-14 GHz range, each locally managed by a Local Monitoring and Control (LMC) system and remotely orchestrated by the SKA Telescope Manager (TM) system.

Dish LMC will provide a Graphical User Interface (GUI) to be used for monitoring and Dish control in standalone mode for testing, TM simulation, integration, commissioning and maintenance.

This paper gives a status update of the LMC GUI design involving users and tasks analysis, system prototyping, interface evaluation, provides details on the GUI prototypes being developed and technological choices and discuss key challenges in the LMC UI architecture, as well as our approaches to addressing them.

SKA DISH

SKA-MID1 Dish array is composed of 15-m Gregorian offset antennas with a feed-down configuration equipped with wide-band single pixel feeds (SPFs) for the bands 1 (0.35-1.05 GHz), 2 (0.95-1.76 GHz) and 5 (4.6-13.8 GHz) of SKA frequency. The array will consist of 133 dishes plus the 64 MeerKAT dishes, arranged in a dense core with quasi-random distribution, and spiral arms going out to create the long baselines that go up to 200km.

Four sub-elements can be identified in the SKA-Mid1 dish element: the Dish Structure (DS), the Single Pixel Feed (SPF), the Receiver (Rx) and the Local Monitoring and Control (see Figure 1).

The Dish structure features the following components: an offset Gregorian reflector system with a feed-down configuration to optimise system noise performance, a fan-type feed indexer at the focal position which allows for changing between the 5 frequency bands by moving the appropriate feed into position, a pedestal providing a RFI shielded cabinet for housing digital electronics and computing equipment hosting other sub-elements' controllers, hardware for antenna movement control and monitoring (Antenna Control Unit or ACU), power distribution to all sub-elements, networking equipment, lightning protection and earthing, cooling ventilation for all the equipment mounted in the RFI shielded compartment itself.

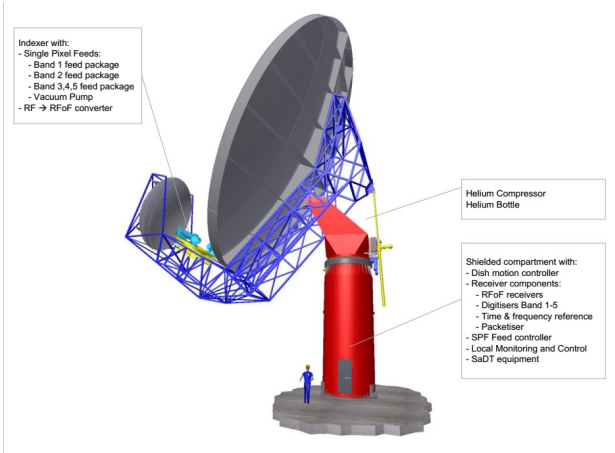

Figure 1: SKA DISH overview.

Single Pixel Feed (SPF) receivers include feed packages for the bands 1 (0.35-1.05 GHz), 2 (0.95-1.76 GHz) and 5 (4.6-13.8 GHz) of SKA frequency, three cryostat assemblies (respectively for band 1, band 2 and band 3,4,5) housing each a Gifford McMahon (GM) cryogenic cooler to cool the LNAs at a set point of approximately 20K, a second amplification stage and a calibration noise source, both temperature stabilised inside the vacuum, a common shared Helium System, a Vacuum System and a SPF controller, i.e. a single controller located in the pedestal which controls and monitors all three feed packages, helium system and vacuum system, and interfaces with the Dish LMC for external control and monitoring.

The Receiver (Rx) includes the following components: RF over fibre transport to the antenna pedestal where the digitisers are located, Digitizers performing some RF conditioning (filtering and level control), digitisation, packetizing and transmission to SKA Central Signal

† marassi@oats.inaf.it

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA188

Processor (CSP), the Master Clock timer which receives time and frequency reference inputs externally and generates timing and frequency references and a Central controller that acts as single point of control and monitoring to the LMC sub-element.

Dish Local Monitor and Control System (LMC) is the subsystem for each dish antenna that deals with the management, monitoring and control of the operation as orchestrated by the Telescope Manager (TM). It consists of a commercial off the shelf controller that serves as a single point of entry for all control and monitoring messages to the outside. Besides configuring the static configurations of the various sub-elements, it also relays the real-time pointing control and applies local pointing corrections. For the monitoring, it aggregates and filters monitoring data as set up from the external (central) controller. The LMC allows for a drill-down capability for maintainers to access detailed diagnostic information of sub-elements on request. The LMC implements also a circular buffer of detailed monitoring information that can be downloaded remotely for diagnostics purposes after a system failure.

DISH USER INTERFACES

From the actual available information about operations and development, maintenance, diagnostic, integration needs, two user interface types are assumed from the element side:

1. Engineering interfaces (engineers for test, diagnostic, maintenance of DSH sub-elements)
2. Navigation interface (control room operator for operations purposes)

With the following user roles:

- engineer (responsible for maintenance operations and hardware fixing)
- operator (operates the telescope in the control room)
- scientist on-duty (person in charge of supervising the observation)
- software maintainer (provides everything is needed from a software point of view)

LMC is in charge of the design and implementation of its own engineering interface, which should also enable the access to the other sub-elements UIs, while the navigation interface is TM direct responsibility.

Engineering interfaces will be accessible either directly from LMC (to be connected with keyboard/mouse and a screen) as desktop application or remotely (from the control room or via some other application (e.g. a TM simulator)) using a tunnelling mechanism (e.g. SSH tunnelling) and are supposed to support authorization and authentication for specific interactions like changing some configuration parameters or setting mode for a sub-element.

LMC will provide GUIs to be used for testing and DISH control in stand-alone mode for testing and commissioning and maintenance. A TM simulator to be used via GUI is envisaged too.

A main LMC engineering interface is here assumed, offering the following functionalities:

- basic DSH control & monitoring
- set-up, control and testing the integrated (or not/partially integrated) Dish
- launch Element specific UI/tools for configuration, debugging, testing and diagnostics
- health monitoring
- alarm management
- lifecycle support, maintenance
- provide direct access to monitoring data by external operators (engineers) in case of TM failure
- create custom visualization
- navigation to other DSH sub-elements engineering interfaces via a tunnelling mechanism

In particular, DISH LMC engineering interface will include the control of Dish States and Modes and will also expose an aggregated health monitoring capability. Such GUI will provide to the test engineer/operator the capability of complete configuration, control and visualization of the Dish parameters in the absence of TM during integrated DSH stand-alone site integration and verification.

It will be capable of managing both TM (or simulated TM) and stand-alone Dish operation. TM (or simulated TM) operation will be performed according to the TM-LMC Interface Control Document.

BACKGROUND CONCEPTS

Usability and Accessibility

The ISO 9241 standard Ergonomics of Human-System Interaction (ISO, 2008) defines usability as “the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use”, specifying:

- *Effectiveness*: the accuracy and completeness with which users achieve specified goals.
- *Efficiency*: the resources expended in relation to the accuracy and completeness with which users achieve goals.
- *Satisfaction*: the comfort and acceptability of use

The ISO 9241 standard Ergonomics of Human-System Interaction (ISO, 2008) focuses on important, but rather difficult to measure goals: effectiveness, efficiency, and satisfaction. For a practical evaluation heuristics may be used or direct usability measures, such as:*Time to learn, Speed of performance, Rate of errors by users,Retention over time, Subjective satisfaction*.

Accessibility is the degree to which a product, device, service, or environment is available to as many people as possible. Accessibility can be viewed as the "ability to access" and benefit from some system or entity. The concept often focuses on people with disabilities or special needs (such as the Convention on the Rights of

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA188

Persons with Disabilities) and their right of access, enabling the use of assistive technology.

Usage-Centered Design

A Usage-Centered Design (UCD) approach [1] for interactive software applications is based on the early involvement of users of the application from its conception. In practical terms, it means that, in order to achieve high usability standard, *feedback offered by users* is to be considered in analysis phases, as well as in design and evaluation. The design process has to be *iterative* also because building a usable UI requires all those involved in its construction to understand, and actually conceive, the mental model that users will have of the application. Each iteration is based on design-prototype-evaluate activities, whereas the *evaluation is based on usability* criteria.

In order to be effective (i.e. produce results that are valid and useful) and efficient (and therefore be sustainable), users need to be involved in structured ways, not simply by asking them casual questions and looking for their opinions. Techniques that can be put in place to follow a UCD approach include structured interviews, contextual enquiries, sketching, storyboarding, user testing, writing scenarios and personas, among others[1],[2],[3],[4].

Sketching and *storyboarding* are techniques to materialize design ideas in such a way that any stakeholder, regardless of his/her design experience, is able to decide if a given UI is appropriate or not for some task [2]. A storyboard is essentially a scenario described through sketches and may be implemented also by using interactive sketches.

A complementary tool that can be used in user-centered design together with storyboarding is conceptual modeling of the user interactions. Interaction design focuses on expressing the content, user interaction, and control behavior of the front-end of software applications through visual diagrams that represent the navigation paths of the user. Various languages, approaches and tools exist to support this task. Among them, we select the Interaction Flow Modeling Language (IFML) [5], an international standard proposed by the OMG (www.ifml.org). It integrates with other mainstream software modeling languages like UML and BPMN, and covers the following design perspectives: the *view structure specification*, which consists of the definition of ViewContainers, i.e., screens, windows, panels and their structure; the *view content specification*, which consists of the definition of ViewComponents, i.e., content and data entry elements contained within ViewContainers; the *events and event transitions specification*, which consists of the definition of Events that may affect the state of the user interface; and the *reference to parameters and actions* to be performed by the software logics.

Interaction modeling through IFML is instrumental to provide a clear conceptual view of the user interfaces, as well as to provide a formal representation of it, which can lead to automatic validation and quick prototype generation on the target platform of choice. Indeed, some tools exist that check the validity of IFML models and compute their properties, based on Petri-Nets or LTL formalization. Furthermore, code generators can produce running applications out of IFML models (for instance, see http://www.webratio.com and http://info.ifmledit.org), and can also integrate user behaviour analysis at runtime with the model specification [6].

USER CENTERED DESIGN ACTIVITIES FOR DISH LMC GUIS

Considering the lessons learned by SKA precursors and the inherent complexity of SKA systems and interactions, a user-centered design approach has been adopted.

As already outlined, user interface design is an iterative process that involves close liaisons between users and designers. It is important to have early users' feedback in the software design and development life cycle to elicit new requirements, validate existing requirements, and highlight possible critical interactions. It covers topics ranging from usage-centered design during analysis and design, through to testing and validation in later application life cycle phases.

Currently the three core activities in this process are:

- User and task analysis: understanding what the users will do with the system
- System prototyping: developing a series of prototypes for experimentation
- Interface evaluation: allowing the users to experiment and explore these prototypes

Starting from the set of requirements on LMC GUIs objectives, users and tasks, we began the analysis by identifying users (engineers, software maintainers) together with their roles and activities and by describing the main usage sceneries of the interfaces by means of *use case* diagrams, detailing the tasks to be performed by the users textually and via swimlane interaction diagrams.

Use cases are used to represent activities and goals that users might want to achieve. More specific goals may be derived from more general ones. In use case diagrams use cases may be linked through generalization, inclusion and extension relationships [1]. This coarse-grained representation can be used as a task model (i.e. a representation of how designers expect that users would carry out relatively complex activities).

The *analysis of the users* of the system is another crucial point in the application of the UCD approach, because it helps to address the user characteristics that can impact on the design of the interface. Its aim is to describe what are the activities that the user performs to achieve his/her goals and how they would use the available technology to do it. In general, information collected in a profile describing a user role can be used to derive *design objectives* and to validate a user interface. A profile of the Dish element/sub-element engineer has been obtained performing user analysis by using sketches and

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA188

storyboards as discussion documents for brainstorming, with the aim of eliciting opinions of stakeholders. We decided to implement these sketches with one of the several existing tools (Balsamic Mockups) also able to implement limited interactive features, such as the definition of clickable hot spots that are linked to other sketches to illustrate interactively the intended dynamics of the UI. In this way we explored several design ideas such as the interaction models and the features to implement. Stakeholders have also been presented with a discussion document integrating and refining the initial set of requirements with a new set of tentative scenarios proposals illustrated by means of an interactive mock-up simulating the interface. The discussion with stakeholders resulted in the definition of a set of *user roles* and *tasks* that take into account both the *context* within which the role is played and the characteristics of the performance.

Figure 2:Sketch of a screen of DISH LMC engineering interface showing the alarm management UI.

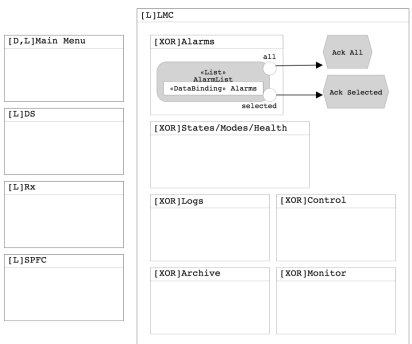

Figure 3: Partial conceptual model of DISH LMC engineering interface.

Figure 2 shows one of the interactive sketches that were produced during the design process of the UI. It shows a panel dedicated to the Dish LMC engineering interface. It allows the user to open also the other Dish sub-elements UIs and to get back to a main menu in which further high level selections are available. The main part of the panel consists of a simple tabs bar implementing a flat menu of options and controls windows to be chosen and opened by the user via a simple click. The open window actually shows the Tango Alarms Managements GUI designed and implemented at Elettra. Figure 3 shows a (partial) conceptual model designed with IFML for the same user interface, where the model highlights the structure of the interfaces in terms of main windows and, for the LMC window, also the organization in panels, which are represented as XOR sub-screens, because they will always be shown once at a time. The opened panel (Alarms) contains the list of alarms and the two options available, i.e., the possibility of acknowledging all alarms at once, or only a selected subset of them.

Figure 4 shows another of the interactive sketches: a tentative DSH element view (operator) aggregated dashboard, where all the most important information regarding the whole DSH element is displayed drawing the attention to element/sub-element states also by the use of colours (green = OK, yellow = warning, red = alarm).

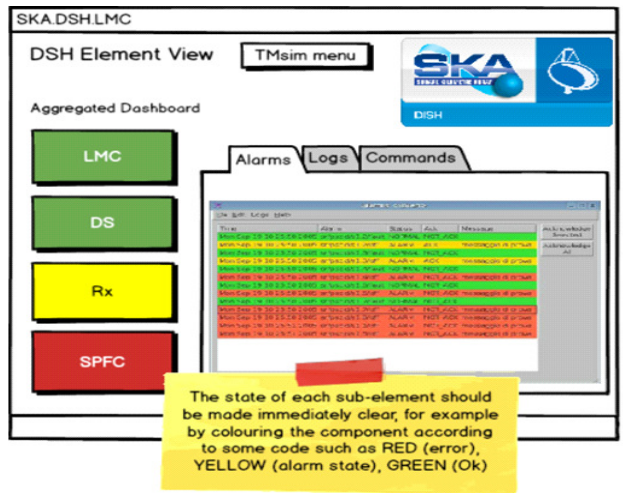

Figure 4: DSH element view (operator) aggregated dashboard.

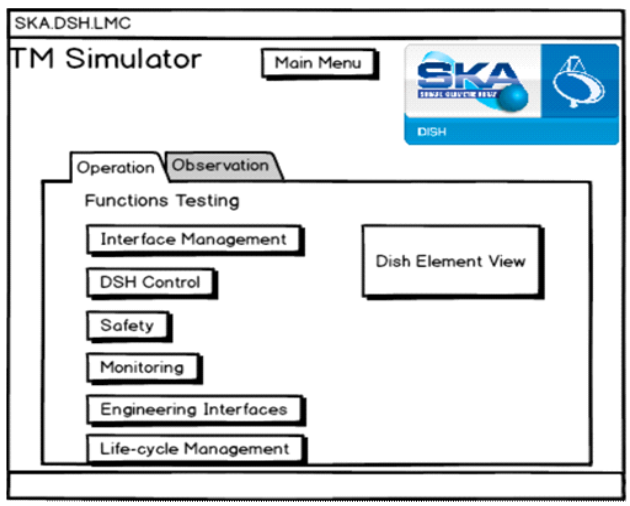

Figure 5: TM simulator window.

The main operator dashboard offer the user a drill-down navigation function to navigate sub-elements and monitor low-level components detail data.

From the same main operator dashboard it is possible to open a TM simulator window (Figure 5) from which TM

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-THPHA188

(or simulated TM) operation can be performed according to the TM-DISH Interface Control document. Similarly to the case reported in Figure 3, IFML models are designed for all the other sketches (not reported here for space reasons).

TECHNOLOGICAL PROTOTYPING

Within the SKA Dish UI workstream, prototyping is also a key tool for the evaluation of technologies. Based on the documented lessons learned by precursors (LOFAR [7], MeerKAT[8], ALMA[9], ASKAP[10]) and specific UI analysis conducted by precursor sites (LOFAR), a set UI tools has been selected to be analyzed against SKA Dish UI requirements. The set includes TANGO tools (e.g. Taurus and Sardana) and general UI frameworks (such as AngularJS, Django, PyQT, PyTango, and TurboGears).

The Tango Control System was selected as the SKA framework in March 2015. The TANGO framework and its UI tools support the types of basic control interfaces that are currently used at both radio telescopes and within high energy physics experiments. They can be divided into Desktop TANGO UI tools and those that provide a web-based interface to a TANGO environment. The options for TANGO desktop development includes ATK based on Java Swing, QTango based on C++ and Qt and Taurus based upon Python and PyQt. These all fulfill the basic SKA.DISH UI requirements and could be used to implement desktop UIs like SKA.DISH UIs.

On the practical aspects of the prototyping, the technology decisions all seem reasonable. In particular Taurus has been used to implement SKA.DISH UIs prototypes with success.

CONCLUSIONS

Considering the lessons learned from precursors, it is clear that proper analysis and design activities are needed. These should be focused on what combination of UIs will best support SKA users (operators, scientists on duty, schedulers, PIs, etc) and based on usage-centered development practices. Such a usage-centered development approach can mitigate product risks (i.e. those concerning with what will be developed and whether it will be the *right* solution) and consists of several key steps:

1. To elicit new requirements in terms of activities that have to be supported (through techniques such as user analysis, precursors analysis, brainstorming and focus group sessions among stakeholders)
2. To study the tasks that SKA users would have to carry out (through task analysis, use case modeling, scenario definition, sketching and storyboarding)
3. To design and validate appropriate UIs (through refinement of sketches, storyboards and low-fidelity prototypes and user testing).

REFERENCES

[1] Constantine, L. and Lockwood, L., [Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design], Addison-Wesley Professional, (1999).

[2] Greenberg S., Carpendale S., Marquardt N. and Buxton B., [Sketching User Experiences: The Workbook], Morgan Kaufmann, (2011).

[3] Rosson M.B. and Carroll J.M., [Usability Engineering: Scenario-Based Development of Human-Computer Interaction], Morgan Kaufmann, (2001).

[4] Preece J., Sharp H. and Rogers Y., [Interaction Design: Beyond Human-Computer Interaction], Wiley, (2015).

[5] Marco Brambilla and Piero Fraternali. Interaction Flow Modeling Language: Model-driven UI engineering of web and mobile apps with IFML. Morgan Kaufmann, 2014.

[6] Carlo Bernaschina, Marco Brambilla, Andrea Mauri, and Eric Umuhoza. A big data analysis framework for model-based web user behavior analytics. In International Conference onWeb Engineering, pages 98–114. Springer, 2017.

[7] Van Haarlem M.P. et al., "LOFAR: The LOw-Frequency ARray", Astronomy & Astrophysics, vol. 556, A2 (2013).

[8] Alberts M., Joubert F., "The MeerKAT Graphical User Interface Technology Stack", Proc. ICALEPCS 2015 User Interfaces and Tools, 1134-1137 (2015).

[9] Pietriga E., Cubaud P., Schwarz J., Primet R., Schilling M., Barkats D., Barrios E., Vila Vilaro B., "Interaction Design Challenges and Solutions for ALMA Operations Monitoring and Control", Proc. SPIE 8451 (2012).

[10] Guzman J.C., Humphreys B., "The Australian SKA Pathfinder (ASKAP) Software Architecture", Proc. SPIE 7740 77401J-1 (2010).