CODE GENERATION BASED ON IFML FOR THE USER INTERFACES OF THE SQUARE KILOMETRE ARRAY (SKA)

M. Brambilla, S. Pavanetto, M. Gasparini, Politecnico di Milano, Milano, Italy
A. Marassi, R. Cirami, INAF-Astronomical Observatory of Trieste, Trieste, Italy

INTRODUCTION

SKA Dish
The Square Kilometre Array (SKA) project is responsible for developing the SKA Observatory, the world’s largest radio telescope ever built: eventually two arrays of radio antennas - SKA-1-Mid and SKA-1-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 SKA-1Mid 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 User Interfaces
In this work, we have considered the Engineering interfaces used by DSH sub-elements engineers for test, diagnostic, maintenance and, in particular, LMC engineering interface, already identified and specified in previous work. An example is reported in Figure 2. LMC will provide GUIs to be used for testing and DSH control in stand-alone mode for testing, commissioning and maintenance, offering basic functionalities of DSH control & monitoring, set-up, control and testing, health monitoring, alarm management, lifecycle support, direct access monitoring in case of TM failure.

Qt/Taurus code generator based on IFML
The Tango framework and its UI tools, selected for SKA in 2015, support the types of basic control interfaces currently used at both radio telescopes and within high energy physics experiments. We aim at the development of a Qt/Taurus code generator prototype based on the IFML (Interaction Flow Modeling Language) standard and respective modeling tools, that are extended for supporting the platform-specific code generation, thus enabling the use of low-code development in SKA GUI design, with increased efficiency, reliability and coherency of the produced UI.

METHODS

Usability and Accessibility
We aim at maximizing usability and accessibility of SKA-LMC user interfaces.

USAGE-CENTERED DESIGN
We start by applying usage-centered design (UCD) approaches for interactive software applications, based on feedback offered by users, iterative design, prototyping and evaluation based on usability.

All the DISH LMC GUIs Usage Centered Design activities have been carried out as part of the tasks performed in the so-called SKA pre-construction phase by the SKA DSH consortium (SKADC). INAF – Catania Astrophysical Observatory, as member of the SKA.DISH consortium, had the responsibility of DISH LMC design, prototyping, testing and validation.

In this work we apply conceptual modeling of user interaction, focusing on expressing the content, user interaction, and control behavior of the UI through visual diagrams that represent the navigation paths of the user. Interactions have been modeled using the standard Interaction Flow Modeling Language (IFML) [1]. IFML is instrumental to provide a conceptual view of the user interfaces (see excerpt in Figure 4), which can lead to automatic verification and quick prototyping.

We exploit IFML as a conceptual modeling language, and IFMLEdit.org as an editing tool and implementation platform for specifying a full model-driven development process with automated code generation for Qt/Taurus. To this end, we apply the practices of model-driven software development (MDD), which entails automation of some of the steps of the development process from a high level conceptual representation of the desired software features, down to deriving a running application out of it, possibly through a set of intermediate steps to enable customization (Fig 3).

Figure 3 The model-driven development process.

When following an MDD approach, the running application can be obtained through one or more model-to-model (M2M) and model-to-text (M2T) transformations that subsequently produce designs, implementation and test cases of the software. Artefacts are (semi-)automatically generated using transformations taking as input the models obtained in the previous phases.

Figure 4 The model-driven development process.

CONCLUSIONS

Control of large-scale scientific infrastructures like SKA requires coherent and effective user interfaces that can be specified only through on usage-centered development practices. Us implemented through model-driven development and automation of the code generation process can obtain highly configurable and yet standardized interfaces as requested. In this work we demonstrated the feasibility of the approach and we reported on the implementation of a prototype of code generator. Future work will include the extension of the generator and field-testing of the generated interfaces.

REFERENCES


ICALEPCS 2019, October 5-11, New York - NY