ESOC New Generation M&C User Interfaces

P. Steele[1]
ESOC, Darmstadt, Hessen, Germany

F. Flentge[2]
ESOC, Darmstadt, Hessen, Germany

J. Schütz[3]
ESOC, Darmstadt, Hessen, Germany

M. Pecchioli[4]
ESOC, Darmstadt, Hessen, Germany

Performing efficient operations requires a consistent interface to the varied complex back-end tasks performed by a control system. A User Interface (UI) must be fast, clear and efficient for routine user tasks, allowing the user to concentrate on difficult tasks. Different operator communities have different UI needs: interplanetary missions (infrequent but long passes, non-real-time), Earth Observation missions (short but frequent passes), continuous coverage missions and ground station controllers (real-time operations, focus on automation) etc. This may result in conflicting solutions and design. SCOS has provided the user interface for ESA's spacecraft operators with great success for 20 years. The current implementation however relies on a specific COTS product and on a specific platform. As development and maintenance moved between different contractors and used varying technologies, small inconsistencies were introduced and maintenance effort increased over the years.

ESA's EGOS User Desktop (EUD) provides a framework for M&C User Interfaces for all types of ground segment systems, including ground station backend systems and mission control systems. This not only allows exploiting synergies in the UI development but also facilitates a common look and feel across all ground segment systems. To adapt to the varying needs and different software interfaces a flexible approach based on Eclipse has been chosen and an Agile development methodology followed. Two operational pilots of the new EUD User Interface are underway: Ground Station Monitoring & Control and the parameter monitoring applications of the spacecraft control systems.

In this paper we introduce the general challenges of creating UIs for M&C control operations, EUD and explain the generic approach using the mimic displays as an example. We then focus on the specific UI for the Mission Control System. We conclude with a short summary and an outlook on future developments.

I. Introduction

A consistent User Interface (UI) begins at the design stage. ESOC's EGOS User Desktop Project (EUD) defined guidelines and interfaces for the suite of applications in use for monitoring and control by a variety of

[1] Technical Officer, HSO-GI.
[2] Technical Officer, HSO-GI.
[3] Technical Officer, HSO-GI.
[4] Head of Section, HSO-GI.

spacecraft and groundstation operators at ESOC. An analysis of the functionalities of the existing display methods was performed and then compared in order to identify the functionality to be provided by EUD based User Interfaces. The users of each system are heavily involved in the design stage and review mockup examples of the proposed implementation at different stages. This *Agile* approach allows at each sprint to consolidate/refine the requirements, update the design, provide and test the implementation. Agile developments are currently ongoing for the following suites:

- ESA's Spacecraft Control Operations System (**EUD4S2K**)
- Graphical representation of systems (**EUDMIMICS**)
- Use of EUD to display and control simulator data (**EUD4SIMSAT**)
- Pilot operational deployment with GAIA spacecraft operators (**EUD4MCSOPS**)
- New generation Ground Station Monitoring and Control (**GSMC**)

The scope of these EUD projects is to provide consistent implementations for:

- General look and feel of application and its displays
- Look and feel for specific displays
- General behaviour of displays
- Common parameter monitoring displays

II. EUD Development and Deployment at ESOC

Five parallel developments are currently underway, providing ESOC users with new display capabilities as well as the ability to customize and share their own displays and control systems from an improved set of widgets. The development projects are all managed via an Agile process. This process enables a much closer collaboration between the users and the developers. Specifically the developer will propose a design and set of functionalities, perhaps inherited from the previous (non-EUD) versions of the applications. This proposal is then turned into a mockup presentation highlighting the differences and the users have the opportunity to redesign the GUI. More importantly, the developers exchange ideas with the users to improve the design and to inform the users why a design choice is necessary or more efficient. Operations that are frequently performed are given precedence over seldom used operations resulting in optimization of the display space. This leads (gradually) to a better appreciated application and a reduction in software change requests after deployment. The duration of the process from proposal of design to implementation can be as short as six weeks and is referred to as a sprint. Sprints enable faster feedback in both developer and user directions and allow the project to stay 'on track' with reduced time lost to investigate misunderstandings or poor requirements. The associated overhead of increased feedback during the sprint has already proven beneficial in the long term for these developments.

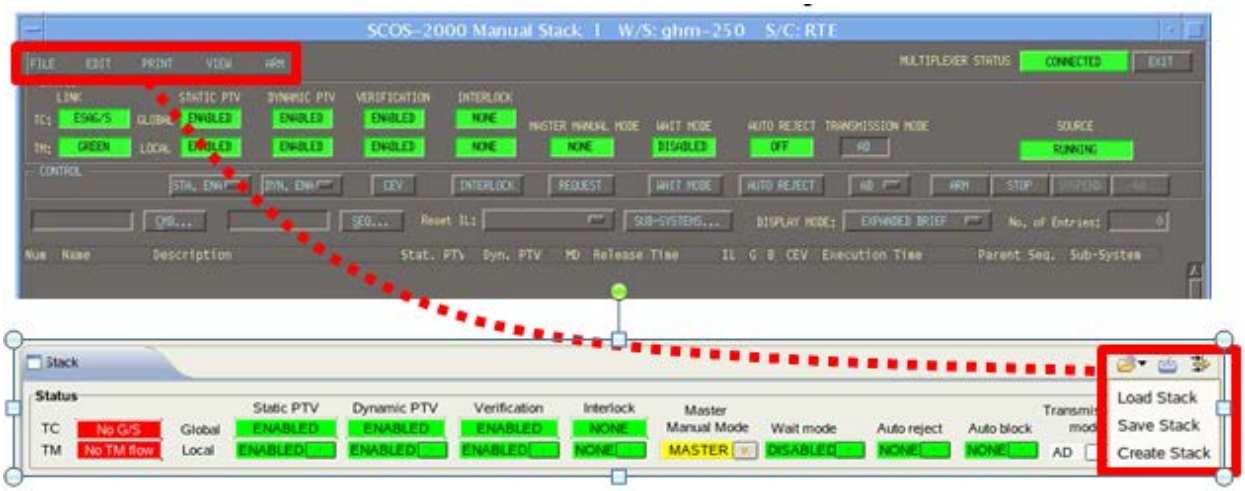

Figure 1 Example Mockup Display Proposal (original at the top)

Various companies worked on the SCOS front end user interface and the back end model processing both in development and maintenance over fifteen years. The suite of applications have been changed, redesigned, fixed, re-implemented and updated throughout this time. The result is numerous cases of inconsistencies between filter mechanisms, dialogue boxes, presentation formatting, autocompletion, wildcard usage etc. In some cases the original design decision was made on request from a specific mission type e.g. LEO missions, but would not be the current preferred decision of e.g. interplanetary missions or ground station engineers. The UI must combine the needs of the various users without reducing the efficiency of the interface to the back-end models. A "One size fits all" solution is the most effective system to develop, deliver and maintain but this must be compared with the requirements of the different users and developers/maintainers.

The EGOS (ESA Ground Operation System) User Desktop EUD 1 was made available in November 2008. It provides a development and run-time framework for User Interface and contains basic displays: Status, Message, parameters monitoring Alphanumeric Displays (AND), parameters monitoring Matrix Displays, parameter plotting Graphical Displays (GRD), parameters monitoring Scrolling Displays (SCD), Login, Task, NavTree, Roles & Privileges- and Account-Display. The EUD was first adopted to develop the UI of new ESOC ground systems SW such as DABYS and also to refurbish existing interfaces such as the SCOS Management Interfaces.

EUD 2 was/is a "Consolidated" version integrating more sophisticated parameter monitoring displays (EGOS Views) and providing enhanced display management functionality. EUD 2 was released in 2010.

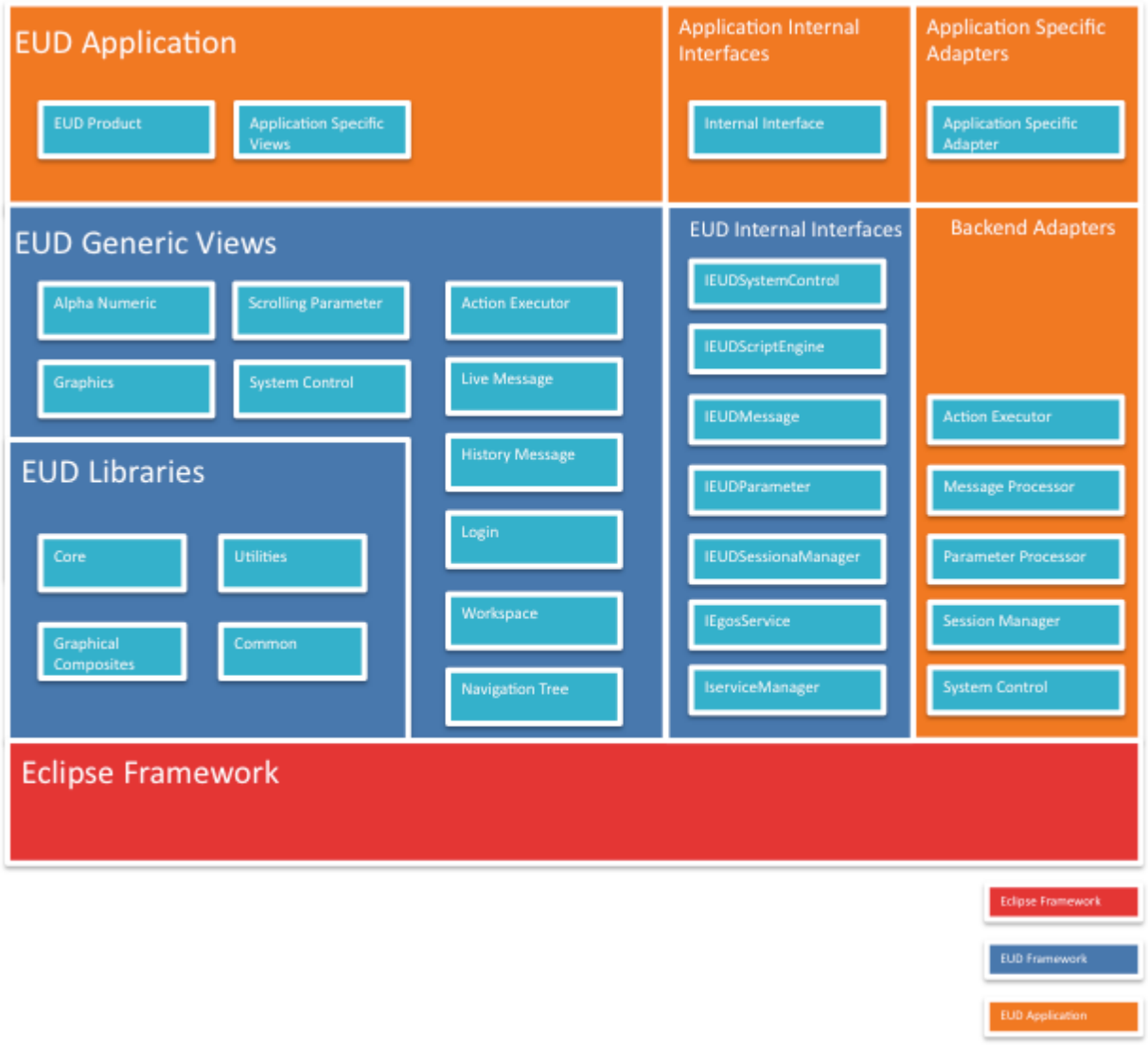

Figure 2 The EUD Layers

The EUD-2.0 can be described as three main layers:
Eclipse Framework: Eclipse Rich Client Platform (RCP) provides the base framework for plugin architecture, graphical elements and binding mechanisms. It also provides mechanisms to store preferences, define extension points, keyboard shortcut mappings, model and view graphical elements, etc.
EUD Framework: The EUD framework is built recurring to the Eclipse framework, namely its plugins based approach and SWT/JFace graphical user interfaces widgets. It defines a set of libraries and provides also generic views. Due to its generic nature, the EUD framework is not tied to any particular backend system (e.g. GSMC, SCOS or SIMSAT), instead the EUD framework defines a number of internal interfaces to access generic services that have to be implemented and adapted to each backend system.
EUD Application: The EUD application is built on top of the Eclipse and the EUD framework. It is made of: 1) Backend Adapters that implement the EUD Internal Interfaces and adapt them to a particular backend (e.g. SCOS), 2) Views that are specific to the application and not provided by the EUD framework and 3) the product specification. EUD makes use of the Adapter pattern as it is the responsibility of a particular backend adapter to translate calls to its interface into calls to the original backend system and vice-versa.
The adapter is also responsible for transforming data into appropriate forms, e.g. the different parameter definitions used in the various ground segment systems have to be translated into the IEUDParameter interface in order to be displayed in the EUD views.

In Eclipse RCP, a perspective is a visual container for a set of views and content editors. The views exist wholly within the perspective and are not shared, but any opened content editors are shared across perspectives. If two or more perspectives have the same view opened, they share the same instance of the view although its layout may differ in the perspectives. For perspectives in different Workbench windows, neither editors or views are shared. A perspective is like a page within a book. It exists within a window along with any number of other perspectives and, like a page within a book, only one perspective is visible at any time.

In the frame of EUD4S2K (the ESOC Spacecraft Operations GUI suite), a client application is a piece of software with i) a presentation layer (the graphical user interface + presentation logic) developed on top of EUD 2 and the Eclipse Rich Client Platform ii) a piece of business logic completely decoupled from the GUI, wrapped within the so called service and iii) the service: a piece of code that encapsulates part (sometimes all) the business logic required by the front-end. Client applications may use several services. In addition, some services are general purpose and shared among all applications. This is for example the case for the File Archive (FARC) and the parameter Data Archive (DARC). Service reusability is one of the key points of the architecture.

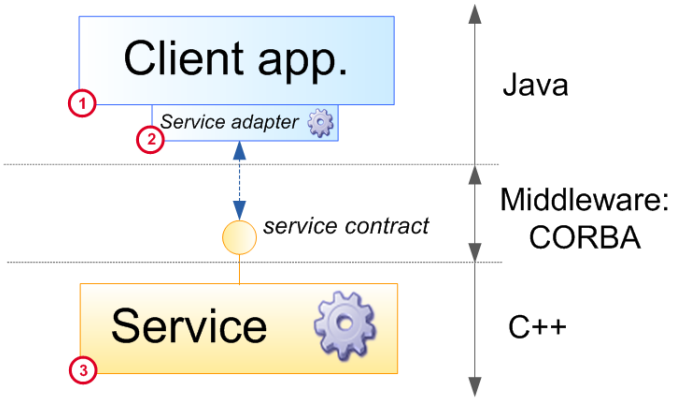

Figure 3 Application concept

Adopting the concept of "service adapter" results in technology agnostic client applications, leading to a system which is robust to changes in the implementation of the service.

In the visualisation part, the functionality provided by each old Ilog Views-based application is migrated to a new Eclipse view. The main implication of this approach is that concrete UI functionality (e.g. a display) is not responsible for the creation of the application skeleton, but just for "contributing" a view (and any other element) to the EUD platform. The same view can contribute to multiple applications, this is a matter of deployment configurations which is under control of the SCOS based Mission Control System developers. The Ilog Views-based applications are being redesigned in pairs in each sprint. Currently Sprint 15 is being performed.

For EUD4SIMSAT, a prototype set of displays are under development. These will be used in parallel to the traditional displays and undergo heavy testing in the operational simulations areas. The prototype incorporates Java interface classes generated from the IDL descriptions. These are directly copied over from the Simsat Kernel. This is in contrast to the classic Simsat MMI which uses a JAR archive provided by the Kernel. The opportunity to redesign the GUI was taken but strong emphasis put on continuity of the GUI concepts to allow experienced users to still benefit.

Throughout the projects using EUD, visual design is applied consistently throughout icons, components, status, basic layout, ergonomic concepts and can be controlled through the configuration of the look & feel covering e.g. fonts and colors. From a practical implementation this is reflected in time selection dialogues, colour and font selection, print selection, filtering (long and short) mechanisms and results, collapsible display/options areas, wildcards, autocompletion, display column re-ordering, tooltips upon hovering over an icon or select areas, hotlinks to user manuals and progress indicators. In the area of graphical displays the users are exposed to tens/hundreds of different graphical display applications such as Excel, Matlab and financial software therefore some functionality increase was expected when redesigning the GUI. Parameter plotting displays (GRDs) now support zooming, autoscaling and a "Statistic mode". The statistic data are used to display the trend of parameters over long time intervals within very short time. Also, in retrieval mode the same GRD is capable of merging data originating from two different sources, namely the DARC containing processed data in engineering form and the native SCOS telemetry processors, extracting data from the packet archive (PARC) and reprocessing them according to the current spacecraft database. . The DARC provides an excellent retrieval performance, whereby the traditional SCOS based approach provides the flexibility offered by the reprocessing of raw data.

Almost all displays can now be heavily customized by the users but with default setup from ESA Ground Systems Infrastructure deployed. Users can in real-time also reorder display data (e.g. move column H next to column C to correlate the data) when investigating an anomaly.

An Example: EUDMIMICS

While the use of mimic diagrams in Mission Control Systems for ESA missions varies from mission to mission the use of mimics in ground station monitoring and control is essential. In spacecraft operations mimics are often used as an additional display to provide some quick overview but operations always relies on Alphanumeric Displays and mimics do not support commanding. In addition, the creation of new mimic diagrams is often tedious, so that they are not used as much as may be desired.

For ground station control the mimic are really the central display for the operator and are used for commanding as well. These differences are probably due to the more "real-time" nature of the ground station operations and the need to provide a communications service. An example of a ground station mimic is shown in the Figure below. Such a mimic allows an operator to easily follow the path of the signal and intervene in case of problems (e.g., rerouting the signal).

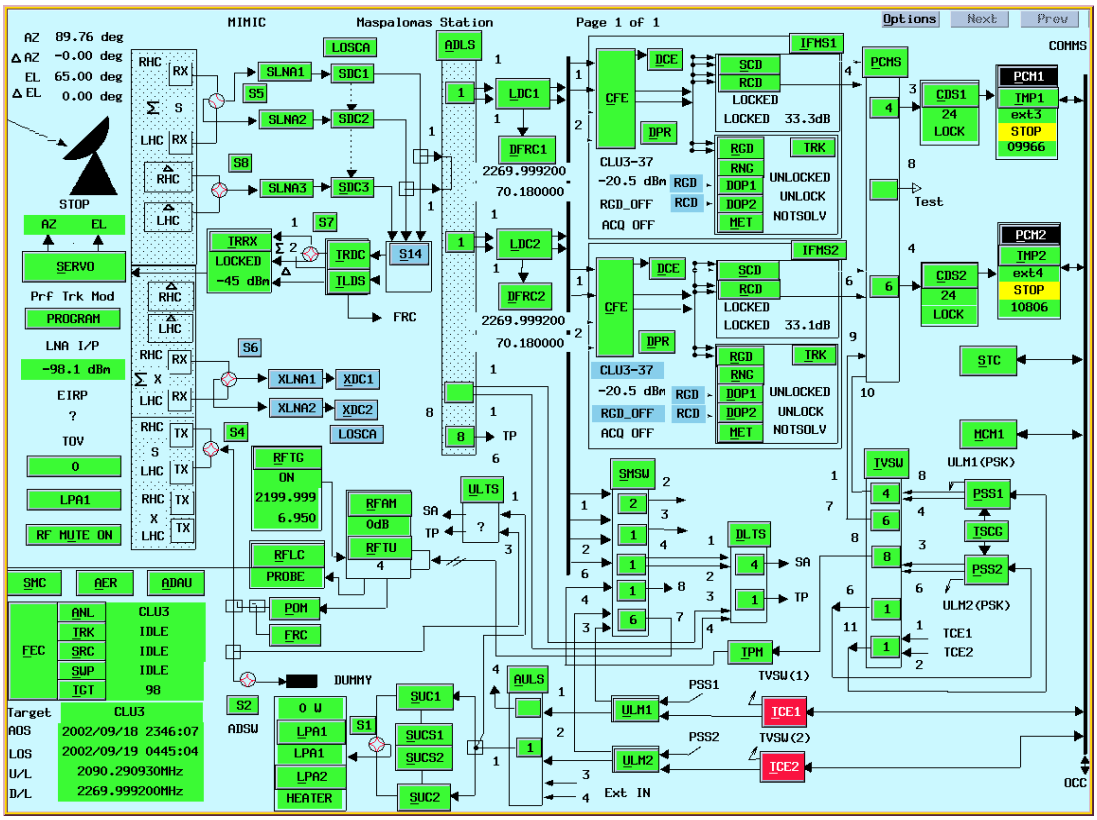

Figure 4 Example Ground Station Mimic before EUDMIMIC

Because of the importance of mimic diagrams for monitoring and control systems, in particular for ground stations, it was always clear that the EGOS User Desktop has to provide mimic capabilities as well. Commercial solutions were not an option due to licensing costs, the administrative overhead of handling licenses, and the need to sub-license ESOC infrastructure products. So, a separate project for the implementation of mimic support in EUD had been set-up and run following a *scrum*. The main requirements for this project were to deliver a Mimic Designer and a Mimic Viewer that should facilitate creation of mimics for ground station and spacecraft operations. Therefore, elements used in different mimics have been analysed and it soon became clear that it is sufficient to support a set of basic widgets (geometric figures, switches, connectors, labels, buttons, images) in addition to a mechanism enabling users to create their own widgets combining these basic widgets. Besides these custom widget mechanisms there were other user requests like the support for sub-mimics (allowing to display more details if needed) and dynamic path highlighting (highlighting the complete data flow if an end-to-end link exists and data is flowing).

To control the dynamic behavior of widgets so-called action procedures can be written in Java Script for each widget. There are two basic types of action procedures. Monitoring action procedures allow to change the appearance of a widget. They are triggered by parameter updates. For example, a label can have a monitoring action procedure to display the value of a certain parameter as its caption. Control action procedures are triggered by the user (commanding a switch, clicking a widget, etc.). These action procedures can be used to control the backend system by sending commands and the user will only be able to trigger them if he/she has sufficient privileges.

As EUD is based on Eclipse it was a natural choice to base the implementation of the mimics on the Eclipse Modeling Framework (EMF) and the Eclipse Graphical Editing Framework (GEF). In addition, the Eclipse Graphical Modeling Framework (GMF) was used in order to allow a model-based development approach. The use of standard Eclipse technologies should also ensure the maintainability of the software for the coming years.

The choice of these technologies proved to be of value as it allowed to produce working prototypes very early in the project and get initial user feedback. Some features that were not in the initial requirements even came for free with these technologies (for example, adding comments to diagrams under design). On the downside, everything which is not natively supported by these tools becomes difficult. So, it turned out that the widget that created the most problems was the line widget as we tried to implement this using connectors. That led to a couple of problems

concerning the placement and the ordering of widgets. Nevertheless, the use of EMF and GEF saved a lot of effort and allowed the implementation of powerful mimic capabilities for the EUD. An example mimic covering part of the mimic in Figure 4 is shown in Figure 5 below.

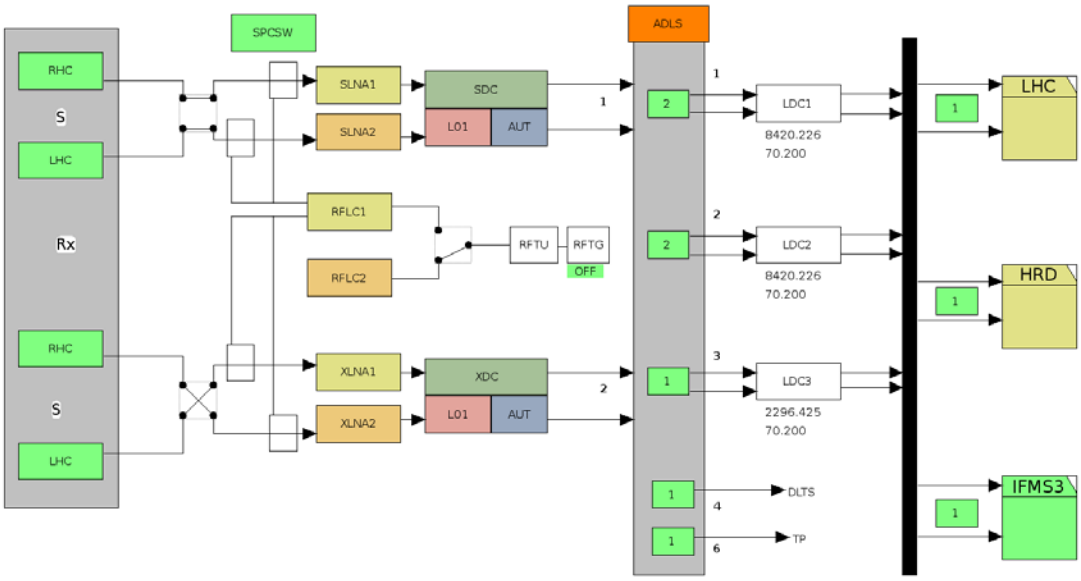

Figure 5 Example ground station mimic with EUDMIMIC.

III. Conclusion and future developments

The development and use of the EUD framework has enabled monitoring and control to become more standardised amongst the large variety of users at ESOC. Better display functionality has been incorporated based strongly on user feedback and collaborative design. Diverse sets of users are participating in design/mockup reviews and in pilot EUD deployments for operations. User interfaces are constantly progressing, especially with the influence of touchscreen mobile devices (phones, tablets), so having an interface that is merely current is no longer enough. The EUD and Agile design process allows fast, efficient and consistent changes to be made. Continuing with these successful pilot projects, other ESOC applications and suites are incorporating EUD design and implementation into their UIs. These include the Generic File Transfer System (GFTS) and the Mission Operations Automation System (MATIS). Close collaboration between users and developers in the design, implementation and validation phases of operational infrastructure software should benefit all participants.