# Approach to automated processes for satellite operations

Gianmarco Broilo[1], Jan-Christoph Scharringhausen[2]

[1] EDRS Subsystem Engineer, Telespazio Germany GmbH, Europaplatz 5, 64293 Darmstadt, gianmarco.broilo@telespazio.de
[2]EDRS Lead Flight Director, Deutsches Zentrum für Luft- und Raumfahrt, Münchener Str. 20, 82234 Wessling, Germany
jan-christoph.scharringhausen@dlr.de

**Keywords:** European Data Relay System (EDRS) – Spaceraft Operations – Automated processes – Procedure based operations – Flight Operation Procedure

**Abstract**

The European Data Relay System (EDRS) provides a high speed data link between ground stations and satellites in Low Earth Orbit (LEO). The communication between LEO satellites is achieved via optical inter-satellite links, the data is then relayed to ground using a Ka-Band antenna. During nominal operations, up to 200 links per day are foreseen to be commanded by the ground system at DLR's German Space Operations Center (GSOC). This large number of complex activities connected to each link is beyond the capabilities of a classical operational concept where only manual operations are performed by a single satellite operator. Therefore, it is imperative to introduce an automated system with limited human interactions. The support from the operator will be required only for contingencies or special operations either on the ground or on the space segment. Currently, spacecraft operations are based on Flight Operation Procedures (FOPs) which include all required telecommands (TC) and telemetry (TM) parameters to be verified, as well as additional information, such as expected values, comments, expected deviations. In the classical operational concept, the TC release system and the telemetry verification is performed within the single mission control system (MCS) that is SCOS-2000. SCOS-2000 is the European standard MCS software infrastructure developed by the European Space Agency (ESA). Here, telecommands are uploaded on a stack and sent manually by an operator who will also be in charge to monitor the telemetry parameters and look out for possible anomalies. Thanks to the heritage of the automation system implemented successfully to the EDRS-A mission, an automatic control system to operate the EDRS-C satellite was also developed. This state of the art system commands the satellite in its nominal operations entirely without human interaction. It uses input FOPs as the classical operational approach but it executes the whole procedure at once. This concept provides a complete procedure based interface to the spacecraft controller combining commanding and verifying capabilities of entire procedures on a single entity, thus drastically reducing human error in complex procedure executions. This novel philosophy of spacecraft operations will increase the situational awareness of the satellite operator and at the same time provide more automation throughout the mission. It has proven to be highly reliable and efficient, considering the past four years of 24/7 operations and more than 1 million minutes of inter-satellite communication time. It includes automated, manual, as well as semi-automated operation concept focusing on decision breakpoints and automation in between. The goal of this paper is to present this successful approach and highlight the advantages and risk mitigation associated with an automated system for spacecraft operations.

## Acronyms

**DPCC** Devolved Payload Control Center

**EDRS** European Data Relay System

**FOPs** Flight Operation Procedures

**GEO** Geostationary Orbit

**GSOC** German Space Operations Center

**LCT** Laser Communication Terminal

**LEO** Low Earth Orbit

**LMS** Link Management System

**MCS** Monitoring Control System

**MOC** Mission Operation Center

**SCC** Satellite Control Center

## 1. Introduction

The European Data Relay System (EDRS) objective is to provide a high speed data link between different ground stations and satellites in Low Earth Orbit (LEO). The project was created from the need to increase the communication times between a satellite orbiting in LEO and its respective ground stations. A typical communication time between a LEO satellite and a ground station is in the order of 10 minutes, which is all the time available for data downlink and other activities [1]. This is where the EDRS project is employed, providing a data relay using optical inter satellite links. The first EDRS payload, EDRS-A, was mounted onboard the EB9B Eutelsat geostationary satellite in 2016 and located on a 9 degrees East Geostationary Orbit (GEO). EDRS-A is equipped with a Laser Communication Terminal (LCT), as well as a Ka-band inter satellite antenna. The LCT is capable of data rates of up to 1800 Mbit/s, while the data rate decreases to 600 Mbit/s in pararell to a Ka-band link. The maximum distance at which optical links can be achieved in space is up to 45000 km. This system is designed to cope with up to 400 links per day, 200 on the optical terminal and 200 on the Ka-band antenna repeater. The second payload, EDRS-C is on a dedicated satellite based on the SmallGEO platform developed by OHB-Systems AG. It was launched on

the 6th of August 2019 onboard an Ariane 5 Flight VA249 from Guiana Space Port and located on its final GEO orbit at 31 degrees East, with an altitude of approximately 36000 km. The EDRS-C satellite is equipped with a hosted payload and an LCT like his predecessor, which is capable of data rates up to 1800 Mbit/s and a total data volume of up to 40 terabytes per day. It also offers data rates of 600 Mbp/s with the possibility to have data encryption [4]. This system is designed to cope with up to 200 links per day [12]. Such a high command load in combination with a required on board autonomy of eight hours is beyond the capabilities of a manual operational concept. To achieve reliability and efficiency, the introduction of automated systems and processes is required. For the older EDRS-A, an automatic command and control system has been developed at the Devolved Payload Control Center (DPCC), part of DLR's German Space Operations Center (GSOC). This innovative autonomous system is able to command the payload during nominal routine phases without any human interaction. Moreover, it dispatches telecommands to the spacecraft. A Link Management System (LMS) triggers ad-hoc flight procedures to be executed, while an Automator egnine executes and monitors these procedures. In the flight procedures the steps are defined relative to a reference time. The Automator sends the telecommands all at once with an execution time up to eight hours in the future while the telemetry checks are queued and executed at the scheduled time [11].
This work's aim is to present the automation of the routine operations of the EDRS mission, with a focus on the latest EDRS-C satellite. Firstly, we will present the European Data Relay System, encompassing both its ground and space segments, and focusing on the latest Satellite Control Center architecture of the EDRS-C satellite. In the second part, we will delve deeper into the operational concept of procedure-based operations specifically developed at GSOC for the EDRS mission, and the principles of the Flight Operation Procedures (FOPs). Starting from the tools tailored for the initial EDRS-A payload, we will analyze the two most important tools employed for the automation: the link management system and ProToS, which is the automation front-end of the MCS. Finally, we will discuss how the EDRS mission has been integrated into the already existing Multi Mission environment at GSOC, together with the measures adopted to maximise the operator's situational awareness in this complex scenario.

## 2. The European Data Relay System

The European Data Relay System was introduced to reduce delays in communication time between a satellite orbiting in Low Earth Orbit and the ground station, and also to increase the amount of data that can be transmitted over a period of time [11]. The EDRS project is a public private partnership (PPP) between Airbus Defense and Space as prime contractor and the German Aerospace Center DLR. DLR is responsible for the ground network and the implementation of the DPCC for EDRS-A and the Satellite Control Center (SCC) for EDRS-C [1], both located at GSOC. Due to the commercial nature of the program, many high performance requirements have been set and need to be met by the SCC: continuous payload utilization with a target service availability of at least 99.6% over 60 days, expected lifetime over 15 years and a reaction time within 1 hour in case of contingencies. These set of requirements cannot be handled in a manual or semi-automatic operational concept [1]. The EDRS payload is operated with the aid of a fully automated operation engine which interacts with GSOCs core Monitoring Control System (MCS) called GECCOS, an in-house derivative of SCOS-2000 v3.1 developed by ESA. The automation engine on EDRS-A and EDRS-C has been developed to monitor the entire process of telecommands instantition, uplink and execution, as well as reaction monitoring of telemetry. The management of the onboard mission timeline is scheduled completely autonomously, triggered by onboard events or external requests. Telemetry feedback and performance indicators of the service are provided to the Mission Operation Center (MOC).

## 3. Ground Segment Architecture

EDRS involves a number of parties from different national organizations and industry partners. The MOC is operated by Airbus Defense and Space and is located in Ottobrunn, Germany. Its duties are the coordination of user requests, service availability and link planning, which are the basis for the scheduling and commanding of the inter-satellite link requests [1]. For EDRS-C, the ground segment was designed via a layered system architecture which is centered around the core MCS, with the first layer consisting of the commanding front-end and automation engine while the second layer comprises the LMS and interface to the MOC [1]. The mission employs two ground stations for uplink and downlink in the Ka-band frequency, one located in Weilheim, Germany which acts as the prime feeder link ground station and a backup ground station located in Redu, Belgium. EDRS customers uses the MOC as interface for link requests, which are then forwarded to the Satellite Control Center (SCC) for execution [9]. The EDRS-C specific ground segment is presented in Figure 1.

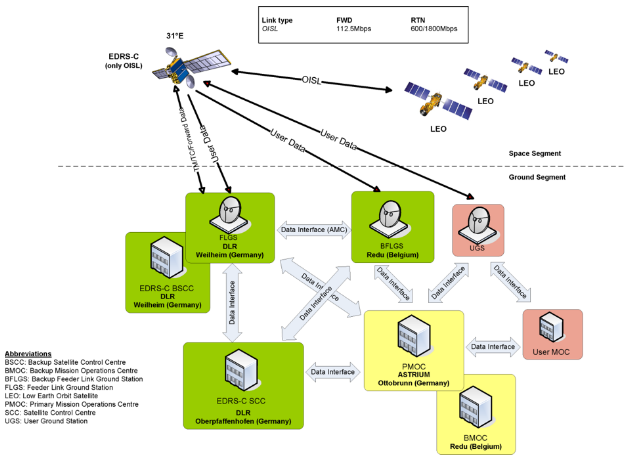



Figure 1. Overview of the EDRS-C ground and space segments. The different data flows with users and control centers are displayed
[10]

The main drivers which were taken into account for the design of the SCC ground segment follow from the EDRS-A concept [1]:

1. A single system design for the payload control center.

2. Fully autonomous routine operations to manage payload utilization, with no need for continuous operator supervision.

3. Seamless integration to the already existing multi-mission environment and reuse of available software and hardware components.

4. Updates and maintenance without service interruption for the entire satellite lifetime of 15 years.

## 4. Flight Operations Procedures at GSOC

Before explaining the difference between a manual commanding approach and a procedure based approach to satellite operations, we analyze the concept of FOPs. At GSOC, there are two types of operational procedures: one type dedicated to all the tasks to be performed on the ground segment, the so called Ground Operations Procedures (GOP) and Flight Operations Procedures (FOP), dedicated to activities to be performed to the space segment. A FOP is a uniquely predefined and validated set of instruction to perform a task organized in steps. The procedure contains all the required information to safely perform the task. It includes: constraints, comments and notes, pre and post conditions, expected state, exact timings for telemetry checks and sequence of commands to be sent in a specific order and checks to be performed on the telemetry. It also contains variables that may have to be instantiated for a correct execution. A sample FOP can be seen in the next Figure 2.

| Step | Time | TC | Activity/Remarks | TM/CmdFlags | Result/Display |
|---|---|---|---|---|---|
| Start of Procedure | | | | | |
| | | | Start of sequence | | |
| | | | *Sequence Parameters* | | |
| | | SSC | ‹*raw*› Source Sequence Counter | | |
| | | APID | ‹*raw*› APid of telecommand | | |
| 1 | | | Delete TTAG | | Next step: ► 2 |
| | | | Delete TT-TCs by Number | | |
| | | | *Command Parameters* | | |
| | | APID | ‹*var*› APID | | |
| | | SEQCNTR | ‹*var*› Sequence Counter | | |
| | | NTC | ‹*raw*› Number of Telecommands | | |
| | | SSCHDID | ‹*eng*› Sub-schedule ID | | |
| 2 | | | TTAG Deletion verfication | | Next step: ► END |
| | | | Report all TT-TCs | | |
| | | | *Command Parameter* | | |
| | | SSCHDID | ‹*eng*› Sub-schedule ID | | |
| | | | End of sequence | | |
| End of Procedure | | | | | |

Figure 2. Sample Flight Operation Procedure (FOP). Clear red demarcations are present to define the start and end of the entire procedure. Blue sections indicate variables to be instantiated while yellow sections indicate the execution of telecommands. White sections are used for telemetry checks

More complicated FOPs have several number of sub steps within a single step and logical decisions, for example: "if-then-else" or "while" statements. In addition to that, they can also include call to other procedures to be performed before continuing with the next step, depending on the operational scenario. Since FOPs are used both by the human operator and the automation engine, they have to be readable by both entities. As an example, special care is needed when selecting the timeout for a telemetry check. The timeout is calculated based on the time needed for the telecommand to be executed (for example a switch on / off time) and the propagation delay caused by the signal to reach the ground station and the telemetry update frequency which can range between seconds and minutes. This information is well known by a trained operator but not by the automation engine. For this reason, a specific time window must be defined for each telemetry check within a procedure to let the automation engine know how long to wait within steps before declaring the step as failed or before proceeding with the next one.

## 5. Concept of manual operations for satellite operations at GSOC

In a typical spacecraft mission in which no automation of processes is achieved, the commanding and monitoring operations are performed solely by the operator. The operator, or spacon (spacecraft controller), interacts with all the components of the ground system, as well as operating the spacecraft by following predefined sets of validated procedures which contains a list of the needed telecommands and telemetry checks to be performed. In addition to that, the operator is the first respondent in case of anomalies on both segments, which are usually presented as notifications and audible alarms. The spacon has the task to correctly identify the problem and give feedback to the engineers to initiate further investigations and recovery actions. In this classical manual operational approach, the operator himself loads each telecommand one by one, or multiple at once, into an integrated manual stack on the MCS. A manual stack is just a graphical user interface of the SCOS-2000 tool used to dispatch telecommands. GSOC uses a enhanced and mission dedicated version of SCOS-2000 which is called GECCOS. The execution of telecommands is performed step by step by the operator or engineer following the required procedure. The telemetry validation that is performed during the execution, is completely detached from the commanding system and is displayed on a different interface. At GSOC, this monitoring interface is called Satmon (from Satellite Monitoring). Here, telemetry parameters are displayed as alphanumeric values and graphical plots. This requires the operator to manually look for specific parameters and sometime to determine the trend of them (for example, a decreasing current when a unit is switched off). To ease the work on the operator and to provide a faster and more efficient way to perform operations, it was decided to move towards a procedure based commanding approach that is based on FOPs and GOPs. With the use of FOPs, the operator is able to load the entire set of telecommands associated to a specific flight procedure directly on the manual stack of GECCOS in the correct order as the written procedure. In addition to that, a great implementation on Satmon was the development of dedicated procedure pages. Procedure pages have been developed to follow all the steps and telemetry checks of a single FOP, showing their expected and current value. Moreover, it also provides the same step titles as the FOP, in order to not create confusion. A simple green or red dot confirms the expected state of a telemetry parameter. Telemetry parameters in the procedure page can be frozen to confirm the output state at the time when the check was performed. Finally, the procedure page can be exported as an image for offline analysis. A screenshot of a typical Satmon procedure page can be seen in Figure 3.

Having these two tools detached from one another, requires the operator to simultaneously keep track on the progress of the FOP on two separate instances, one dedicated to the telecommand stack and one for the telemetry verification. Moreover, some FOPs are connected with each other, requiring the operator to move within different procedures at the same time. The collection of these factors, make the manual commanding slower and more error prone even with the addition of dedicated procedure pages, especially when taking into account complex procedures to be performed under time pressure several times during a day. Since the EDRS mission requires the execution of many link requests per day, using complicated parameters to instantiate, in addition to nominal daily activities and routine tasks, an operational scenario where the operator works to-

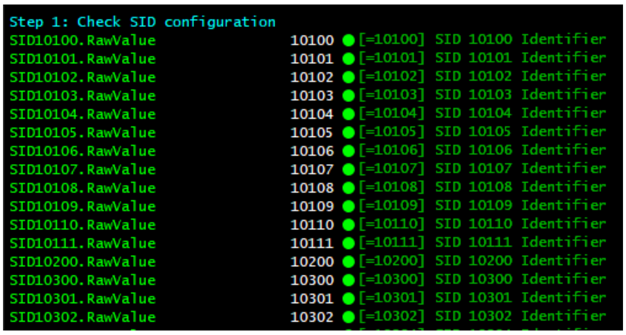



Figure 3. Example of a procedure page displayed in Satmon. The procedure view provides a quick and effective way to determine the state of a specific parameter and if it is within the required value

gether with an automation engine in charge of all routine activities was chosen as the most effective solution. Link requests are scheduled, uplinked and executed as a predefined set of FOPs. Different FOPs are needed to correctly configure the EDRS-C payload to execute the required optical inter satellite link in a safe manner. The automation engine of EDRS-C takes care of all routine and nominal requests to be performed by autonomously selecting the correct FOPs, polling the required parameters from other applications of the ground system to instantiate variables within the procedures, add an execution time to each telecommand and uplink them to the satellite, all without any interaction by the operator. The telecommands within the FOPs are sent all at once and not step by step, allowing them to be stored in the onboard buffer of the satellite dedicated to payload operations and to be executed at a desired time. In this way, nominal satellite operations remain unaffected by a temporary loss of connection with the ground station. This is important as the Ka-band frequency used for uplink and downlink is susceptible to bad weather events [13]. This approach reduces the workload on the operator which is then only in charge to monitor the automation processes and to immediately react in case of a contingency situation. Any non routine or more complicated procedure where an engineering support is required will still be performed by manually commanding the spacecraft.

## 6. Automation of EDRS-C Routine Operations

The SCC has the full responsibility over the satellite platform and payload, and has implemented all its operational products, i.e. ground and flight procedures enabling a continuous service availability. Routine operations fall into two categories: nominal activities related to the link service, based on requests from MOC, and payload configuration and maintenance activities, which must be scheduled in coordination with any external request. Since EDRS-C is capable of almost 200 link requests per day, a fully autonomous scheduling tool called Link Management System (LMS) and the automation front-end ProToS (Procedure Tool Suite) work together and are responsible for the scheduling and execution of all types of automated requests. In addition to that, ProToS controls routine operations on the payload, for example: link requests and internal requests and most of the platform routine maintenance tasks: the complete execution and monitoring of Station Keeping Maneuvers (SKMs) and eclipse monitoring. The LMS is entirely designed and developed by DLR GSOC exclusively for the EDRS mission. All other non-nominal and anomalous situations are manually addressed by the operator and the subsystem engineer.

### 6.1 Link Management System

Since EDRS-A is very similar to EDRS-C regarding link workflow and interfaces, the EDRS-A LMS code base was reused for EDRS-C [9]. The similarity of the exact same LCT payload allowed for a smooth implementation of newer features and bug fixes, benefiting both missions at the same time since the LCT was first developed in 2014 [5]. The LMS is responsible for the scheduling and management of configuration and deletion requests received from MOC and for the scheduling of internal routine requests which can be recurring (On Orbit Propagator update) or triggered by the MCS (time sync of the LCT clock). More specifically, the LMS accepts payload configuration requests, forward data requests and deletion requests from the EDRS MOC, schedules them on the mission timeline as well as archiving on the SCC archive. The LMS is always running in the background and a graphical user interface called LMS GUI is available for easy inspection or maintenance. In case of anomalous behavior, the LMS scheduling process can be stopped or paused. The LMS GUI can be seen in Figure 4.

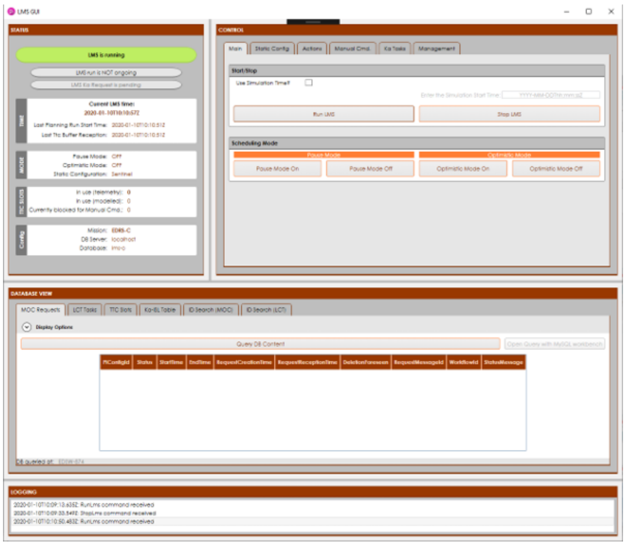


Figure 4. LMS GUI as visible by the operator [8]

The LMS is able to process different inputs and outputs to many interfaces of the SCC. For example, it generates requests to the flight dynamic server to collect the inputs for the variable instantiation and provides its status report to a centralized network monitoring system called NEMO that will be analyzed at a later stage.

Morevoer, the LMS is in charge of monitoring the onboard telecommand memory of the spacecraft and the uplink time budgets. The status of the filling of the on board memory is requested every 20 minutes via an automated telecommand dispatched from the MCS. At any stage of an external request process, report messages are generated and delivered to the MOC in order to keep track of the status of its requests. The interface between MOC and SCC consists of five types of fully automated and one semi-automated payload requests:

1. Payload Link Configuration: link request containing information on the link type data rate and timings. It also contains information about the targeted LEO orbit.

2. Payload Routine Configuration: a request for execution of a predefined FOPs. The request shall contain all necessary parameter values beforehand.

3. Forward Tasking Data: a link request in forward direction, with data to upload to the LEO satellite.

4. Payload Configuration Deletion: a deletion request to remove a scheduled payload link or request from the mission timeline.

5. Forward Tasking Data Deletion: a request to clear the content of the onboard data buffer ready for forward data.

6. Payload Basic Request Configuration: similar to routine request, but to be executed manually by a payload expert and operator. This type of request requires coordination between MOC and SCC.

In addition to these there are several operational workflows that are implemented and executed automatically within the SCC, defined as routine requests [10]:

1. Update of the On Orbit Propagator

2. On-board time correction

3. LCT alignment matrix correction

### 6.2 ProToS: the core of the automation engine

To achieve this level of autonomous operations, the automation engine has a direct access to the FOPs repository, TMTC definitions of the MIB (Mission Information Base) and interfaces with the MCS system. For the EDRS-C mission, this is done with ProToS. ProToS is a software realized as a client/server application with an interface to the MCS. It is written in Java and fully developed at GSOC as part of the GSOC-2020 Research and Development agenda [6]. An overview of the different instances and connections between client, server and MCS can be seen in figure 5. The interface between the server and MCS is achieved using the middleware CORBA (Common Object Request Broker Architecture). This is a standard designed to ease the communication between systems operating on different operating systems.

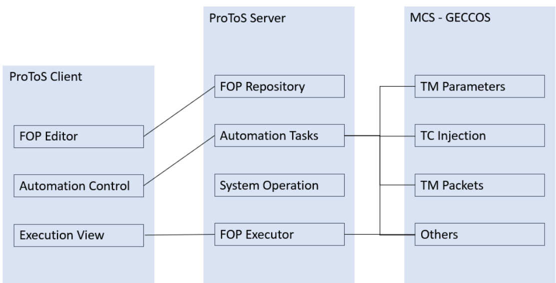
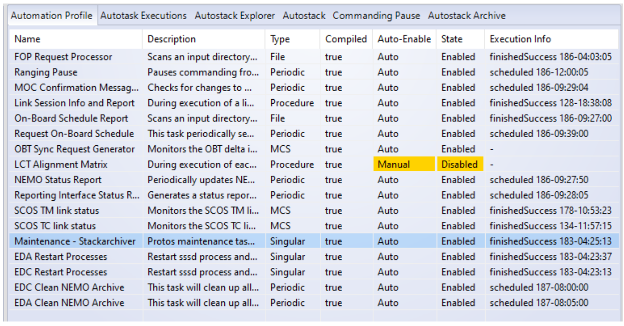



Figure 5. Overview of the different ProToS instances [6]

The task of ProToS is to support the operations team in the creation and execution of FOPs as well as to provide an automation framework for complex operational scenarios. Its user friendly interface allows for different activities to be performed on FOPs: generation, instantiation, execution and automation. Once a payload request or an internal routine request is processed successfully by the LMS, an automation request in the form of a new XML request file is dispatched to the automation engine on the ProToS server instance. Similar automation tasks are grouped in automation profiles on the ProToS server. Each task has a well defined trigger mechanism to initiate the execution. Currently, the active trigger mechanism are: file system event-triggered (procedure request processor), recurring task (status report), MCS event (OBT sync request) and procedure event-triggered (link session report generation). The complete list of the available automation tasks can be monitored from a dedicated page on ProToS GUI: the automation view. Here the operator can monitor the task type (file, periodic, procedure, MCS and singular), task state (enabled or disabled), execution info (schedule time and execution state) and a short description of the task itself. A screenshot of the automation view can be seen in the next Figure 6.




| Name | Description | Type | Compiled | Auto-Enable | State | Execution Info |
|---|---|---|---|---|---|---|
| FOP Request Processor | Scans an input directory... | File | true | Auto | Enabled | finishedSuccess 186-04:03:05 |
| Ranging Pause | Pauses commanding fro... | Periodic | true | Auto | Enabled | scheduled 186-12:00:05 |
| MOC Confirmation Messag... | Checks for changes to ... | Periodic | true | Auto | Enabled | scheduled 186-09:29:04 |
| Link Session Info and Report | During execution of a li... | Procedure | true | Auto | Enabled | finishedSuccess 128-18:38:08 |
| On-Board Schedule Report | Scans an input directory... | File | true | Auto | Enabled | finishedSuccess 186-09:27:00 |
| Request On-Board Schedule | This task periodically se... | Periodic | true | Auto | Enabled | scheduled 186-09:39:00 |
| OBT Sync Request Generator | Monitors the OBT delta i... | MCS | true | Auto | Enabled | - |
| LCT Alignment Matrix | During execution of eac... | Procedure | true | Manual | Disabled | - |
| NEMO Status Report | Periodically updates NE... | Periodic | true | Auto | Enabled | scheduled 186-09:27:50 |
| Reporting Interface Status R... | Generates a status repor... | Periodic | true | Auto | Enabled | scheduled 186-09:28:05 |
| SCOS TM link status | Monitors the SCOS TM li... | MCS | true | Auto | Enabled | finishedSuccess 178-10:53:23 |
| SCOS TC link status | Monitors the SCOS TC li... | MCS | true | Auto | Enabled | finishedSuccess 134-11:57:15 |
| Maintenance - Stackarchiver | Protos maintenance tas... | Singular | true | Auto | Enabled | finishedSuccess 183-04:25:13 |
| EDA Restart Processes | Restart sssd process and... | Singular | true | Auto | Enabled | finishedSuccess 183-04:23:37 |
| EDC Restart Processes | Restart sssd process and... | Singular | true | Auto | Enabled | finishedSuccess 183-04:23:13 |
| EDC Clean NEMO Archive | This task will clean up all... | Periodic | true | Auto | Enabled | scheduled 187-08:00:00 |
| EDA Clean NEMO Archive | This task will clean up all... | Periodic | true | Auto | Enabled | scheduled 187-08:05:00 |

Figure 6. Overview of the automation task on the automation view page in ProToS client. Here is possible to have an overview of the task and when it was last executed, its type, and if the task is enabled or disabled.

The most used trigger in EDRS-C is the procedure or FOP request processor. The procedure request processor is triggered by an XML request file from the LMS: the automation engine will call a specific FOP or set of FOPs from its repository and all telecommands are instantiated with the correct parameters provided by the LMS [1]. Each FOP is not only a set of TCs but it also contains telemetry statements which are checked before the execution of a particular telecommand. The instantiated FOPs are then sent to GECCOS on an internal execution stack to be dispatched to the ground station and then uplinked to the spacecraft. The internal stack is separated from the manual stack used by the operators and the two can work in parallel. The operator is also able to monitor in real-time the commands which are being dispatched by the automation. In case of an automatic request, the telemetry statements are scheduled such that the payload state is automatically verified against a set of pre-conditions. The state flow calculator running on the ProToS server is able to process the command execution, verification and telemetry checks to autonomously evaluate the execution state of a given task and the associated request. This is achieved thanks to the access of the ProToS Server instance to the MCS live TM/TC data. The state flow calculator is also able to decide which step is to be executed next within a procedure depending on the outcome of the previous step. More complex step processing of the form of "if-then-else" or "while" loops can also be performed. In addition to these features, the state flow calculator can also call "procedures within a procedure". The result of each evaluation by the automation engine is automatically logged and reported to the Mission Operations Center via XML confirmation messages in quasi real-time.

### 6.3 Cooperation between the automation engine and the operator during real operations

One of the most important features for an autonomous planning system is that it is running without interruptions with the capability to handle correctly different types of inputs, internal or external [9]. In case of problems with the scheduling system itself or if some invalid data is detected, the system is able to automatically detect it and to alert the operator via a reporting interface. For the EDRS mission, this interface is the NEtwork-MOnitoring (NEMO) application. The main overview page can be seen in Figure 7.

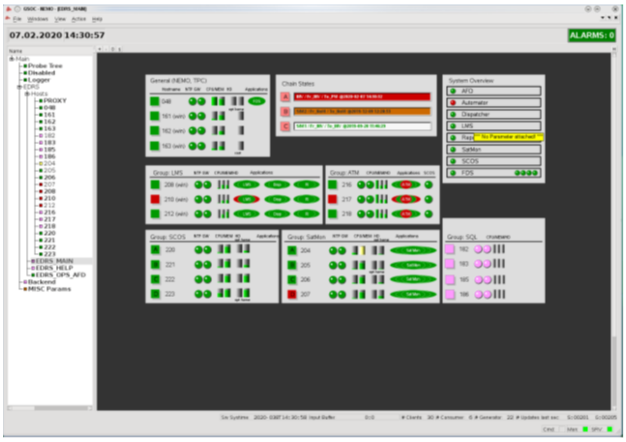


Figure 7. NEMO GUI as visible by the operator [10]

NEMO is responsible of monitoring the entire ground network within the SCC. Whenever an error is detected, a notification with the error name, affected process and short description is sent to the operator via the NEMO Graphical User Interface (GUI), Figure 8 along with an audible alarm.

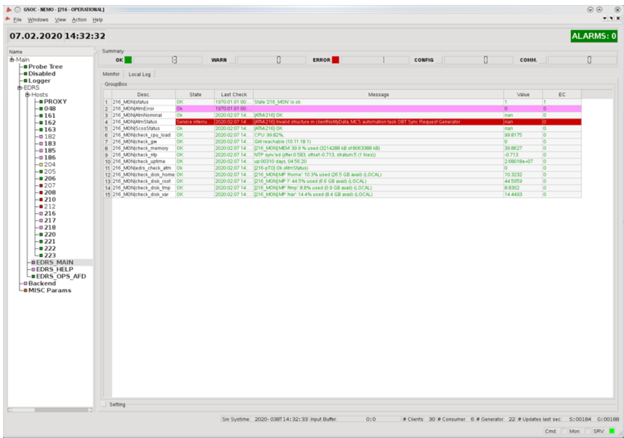


Figure 8. Error message received on the NEMO GUI [10]

Since the operator rarely interacts with NEMO, as it is mostly a monitoring tool, the notification must be as concise as possible [9], avoiding any ambiguous statement that could lead to a misinterpretation by the operator and eventually lead to a slower reaction time. The notification consists of the affected component and a short description of the problem, along with the triggering time of the alarm. The alarm notification can be received multiple times as long as the error is persistent. Moreover, the audible sound is active until the operator manually acknowledges it. These further notifications and manual intervention are very helpful in case of multiple anomalies happening at the same time, preventing the operator to overlook one of the alarms. NEMO has been in used already for many missions at GSOC and its capability has been widely proven [7].

For automated requests which consist of one or multiple FOPs execution, the operator has the possibility to monitor step by step in realtime the execution process of each request directly in ProToS autotask view page. From the ProToS GUI the operator can open individual requests and check the complete list of TCs and telemetry checks, each with a countdown in order to keep track of the overall progress. In addition to that, a green check is marked at each step that has been executed successfully. For telemetry checks, the sampled telemetry value at the time of the check is added. The next figure shows an example execution view for an automated request, Figure 9. In case of

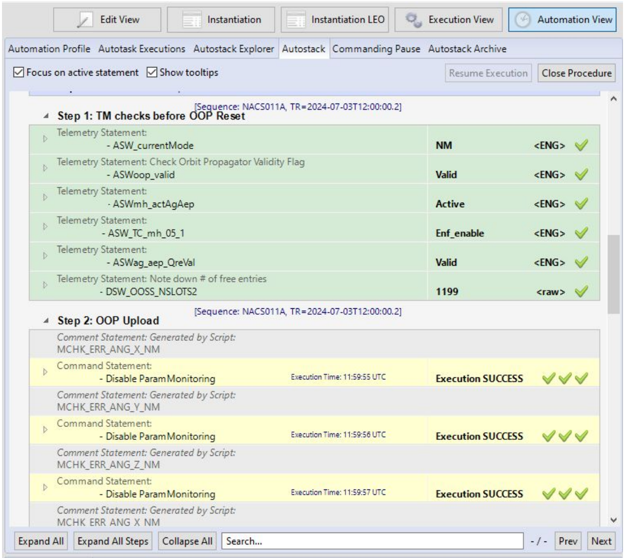



Figure 9. Example of a request processing overview in ProToS. Here, telemetry checks are marked with a green check if the check was successful. Telecommands have three green checks: dispatched, uplinked and executed successfully.

execution fail of a single TC or telemetry check, a red cross is marked at the affected step and the execution of the whole request is stopped automatically and declared failed or it may allow to continue, depending on how it was configured and the operational scenario. The operator can also decide to override the statement made by ProToS if needed, In case of a failed execution, an alarm will then be triggered both in ProToS and in NEMO.

In case of contingencies, it is possible to stop or pause the automatic scheduling processes in order not to interfere with the manual operations dedicated to the recovery of the system or the satellite itself. When the LMS is paused for example, it is still accepting requests, however, it does not schedule any activity or export FOP requests to be processed further by ProToS. Moreover, ProToS can also be paused, avoiding the automatic forwarding of TCs to the MCS and ultimately, to the spacecraft. A major advantage with this functionality is that the reporting capability to the MOC is still active and the system remains responsive. In this case the MOC is able to monitor the status of the LMS and be directly notified in case of outages. A co-operation between manual and automatic processes is therefore required during real operations to resolve any potential anomaly, both on the ground and the space segment. The LMS needs to be able to work with manual activities. If we take as an example the situation in which a telecommand execution fails after the LMS has already exported the FOP request, the operator is able to manually repeat the command execution, thus recovering from the error. The automated system is therefore able to allow to manually trigger certain processes without impacting its capabilities.

### 6.4 Situational Awareness of EDRS-C in the GSOC multi-mission environment

The EDRS-A and EDRS-C satellites are operated as part of the DLR GSOC Multi Mission environment. This environment not only contains the EDRS project but multiple missions, most of which are low Earth orbit satellites. Currently, the missions operated at GSOC are: GRACE FOLLOW ON with two satellites, TerraSAR-X and TanDEM-X and finally EnMAP. In this scenario, the presence of automated processes are very important to ease the workload on the operator, allowing him to take control of a task only if needed and to rely on automation for most of the daily tasks. To provide a seamless integration between multiple projects, many interfaces and applications are common between all missions [1]. Recalling the classical manual operations approach, the operator is in charge on all operational processes and provides constant monitoring of the systems. This allows the operator to always know what is the current status of the ground and space segment and if any process is on going. The addition of an automation engine which is always running in the background and dedicated to most of the daily operations could affect the knowledge of active processes by the operator, thus influencing his situational awareness.

The situational awareness is defined as the knowledge of what is happening around us, this allows to clearly assess what is happening and to develop an effective action [3]. Being aware of the current status of the system and being able to identify threats is extremely important in a space mission. A reduced situational awareness caused by automatic processes must be properly addressed to successfully operate EDRS-C in a multi mission environment. We will now analyze three main levels of awareness and how the tools that we have defined in the previous chapters and other features implemented in the MCS at GSOC are used to help the operator have a full overview of the active processes at all times.

Three main principles of situational awareness have been defined [2] as:

1. *Perception of the elements in the environment*. A centralized overview of the whole ground system and automation engine status is achieved through the NEMO interface and ProToS GUI for automatic tasks. in NEMO, each ground system process will have a defined color based on its status: green if running nominal and red otherwise. This simple color code will allow the operator to immediately detect a process with an anomalous behavior, thus saving a lot in reaction time. Moreover, audible alarms and notifications are visible in the NEMO GUI. The ProToS user interface can also be used to monitor the running automation tasks and all the requests that are being processed at the moment and the ones that are scheduled in the next 8 hours. Any failed telecommand or telemetry check within a request can be seen by simply opening the affected request from the dedicated "Autotask" page. Anomalies related to the space segment are displayed in GECCOS via notifications and audible alarms. Alarms are generated and displayed with different severity levels and colors in order to maximise the awareness by the operator, which will be able to understand just by quickly glancing at the screen if the anomaly requires immediate attention. Like NEMO, GECCOS alarms also generate an audible alarm. For example the color code for spacecraft on board events displayed on the "On board event history" application present in GECCOS changes with increasing severity: no color = notification, yellow = warning, orange = error, red = alarm. Moreover, since the platform is PUS (Packet Utilization Standard) compliant, the on board event packets are numbered with increasing severity: packet (5,1) = notification, (5,2) = warning, (5,3) = error, (5,4) = alarm. Usually a (5,1) packets provide the operator with a notification which does not require any intervention. (5,3) and (5,4) packets typically indicate the mode change of equipment or triggering of autonomous recovery actions, therefore they require immediate investigation by the operations team. This color choice allows the operator to quickly assess the danger without too much effort and act swiftly. A summary of the color code for the on-board events and associated PUS packets can be found in figure 10.

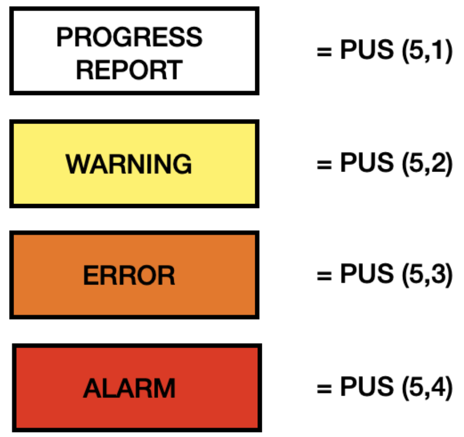



Figure 10. Color code associated to each on-board event with increasing severity

2. *Comprehension of the current situation*. This point is important for the operator to correctly assess the situation and be able to differentiate between a nominal and non-nominal scenario. The aid of audible alarms and notifications from NEMO, ProToS and GECCOS help the operator to define if the anomaly is on the space or ground segment. The NEMO GUI guides the operator to a specific affected component or process while GECCOS alarms provide a short description of the problem and the affected subsystem, as well as the severity of the alarm. ProToS can be used to determine which telemetry check of an automatic request has failed and what is the current state of the satellite. These information are complemented by ground and flight operations procedures that guide the operator

through a correct assessment of the anomaly and to initiate a recovery process. Moreover, the procedures point out if any external entity shall be involved in the process and how to correctly escalate the anomaly to subsystem engineers and flight directors. In addition to these measures, EDRS-C employs a quick lookup tool which is dedicated to the operator and provides an even faster risk assessment and validated recovery strategy with all the relevant FOPs to be used. This lookup tool has the name of SALT (Spacon Action Lookup Tool) and consists of an excel sheet with a quick explanation for on-board events or out of limits actions which is constantly kept updated. The main interface of SALT requires just one input from the operator. The input can be the Event code for an on-board event or the name of the parameter in case of an out of limit. As output, SALT provides a short description of the problem, possible correlated alarms and actions both for the operator and the subsystem engineer. In addition, it provides the documentation used to identify the anomaly. The quick and easy functionalities of SALT make it very useful for both the operator and the engineers as a first mean of anomaly identification and recovery process. The main page of SALT is shown in figure 11.

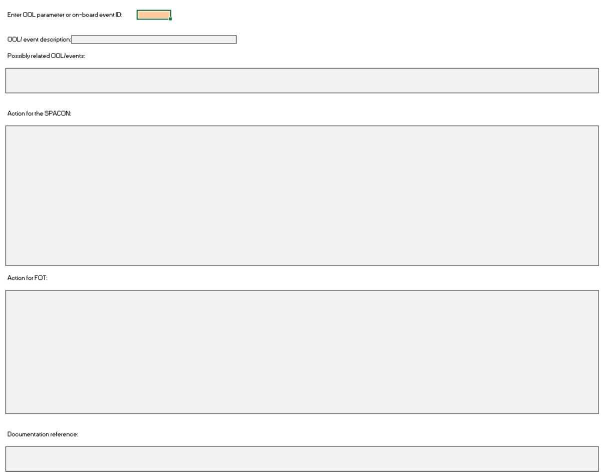



Figure 11. Main page of the SALT lookup tool where input / outputs are shown

3. *Projection of the future status*. In case of contingencies it is very important for the operator to know what would be the impact on the nominal satellite operations and how long it could take to fully recover the service. This level of awareness is achieved by adding an estimated duration time to each FOP, as well as employing a mission timeline that the operator can use to monitor all planned activities in the future. The mission timeline helps the operator to coordinate recovery actions with the least impact on the service. The timeline contains information on planned link sessions, operational activities and constraints, as well as the current ground station configuration. The layout of the mission timeline is displayed in figure 12.

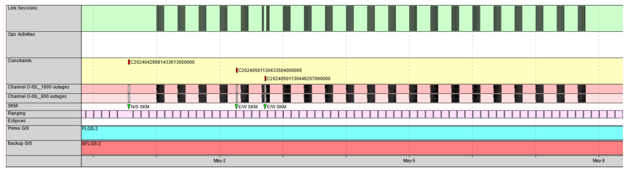


Figure 12. Mission timeline for EDRS-C. Link sessions are displayed in the top timeline, each link session has information on the start and end time

## 7. Conclusion

The DLR GSOC Multi Mission environment is very dynamic, with many satellite missions which are operated continuously. The introduction of the EDRS GEO satellite missions to this environment is challenging and it requires the introduction of novel operational concepts. Firstly, a different operational approach must be considered, shifting from a command based concept to a procedure based commanding which is complemented by a procedure based monitoring using Satmon. To cope with the operational requirements, the introduction of an always-running automation engine and scheduling service which is able to execute most of the nominal operations is vital for the success of the EDRS mission and its implementation in the Multi Mission control room. The automation engine shall not only take care of the nominal operations but it also has to correctly react in case of anomalies and to cooperate with the operator in the recovery process and in any routine maintenance activity. The added complexity of this mission and the many new features that it brings are implemented to the Multi Mission control room in a way that enhances the situational awareness of the operator at all times during the nominal operations phase. This is done by using similar architecture and ground system tools to other missions, as well as providing the operator with effective measures to asses in a short period of time the situation and correctly react to any anomalous situation. As more missions will be launched and operated by GSOC in the future, the automation of processes will be implemented in many aspects of satellite operations, thus bringing many advantages and increasing reliability. However, a well-structured interaction between automation and operators plays a pivotal role; therefore, it must be taken into account during the implementation of such automatic tasks for a successful mission.

## References


Beck, T., Schmidhuber, M., Scharringhausen, J.-C., 2016. Automation of Complex Operational Scenarios - Providing 24/7 Inter-Satellite Links with EDRS. *14th International Conference on Space Operation*.

Endsley, M. R., 2013. Situation Awareness, The Oxford handbook of cognitive engineering. *Oxford Univ. Press*, 88-108.

Endsley, M. R., 2015. Situation Awareness Misconceptions and Misunderstandings. *Journal of Cognitive Engineering and Decision Making*, 9, 4-32.

Esa, n.d. European Data Relay Satellite System (EDRS) overview. `https://connectivity.esa.int/european-data-relay-satellite-system-edrs-overview/`.

Göttfert, T., Wörle, M. T., Prüfer, S., Lenzen, C., 2018. Operating and Evolving the EDRS Payload and Link Management System. *SpaceOps Conference*.

Hamacher, P., Beck, T., 2018. ProToS: Automation of Flight Control Procedures for the European Data Relay System. *SpaceOps 2016 Conference*.

Hauke, A., Ohmüller, T., Häring, U., n.d. An innovative monitoring- and control-system at gsoc and weilheim ground station. *Ground System Architectures Workshop, GSAW*.

Prüfer, S., 2020. LMS Operator Training.

Prüfer, S., Göttfert, T., Wörle, M., 2019. Automated Planning versus Manual Operations in the context of the Link Management System for EDRS – SpaceDataHighway.

Scharringhausen, J.-C., 2020. EDRS-C Training Session 103: Ground Segment Design.

Scharringhausen, J.-C., Beck, T., 2016. Automated Procedure Based Operations for the European Data Relay System. *AC-17-B6.1.2*.

Scharringhausen, J.-C., Kolbeck, A., Beck, T., 2016. A Robot on the Operator's Chair – The Fine Line Between Automated Routine Operations and Situational Awareness. *SpaceOps 2016 Conference*.

Yates, J., 2023. The impact of weather on Ka-band frequencies. https://room.eu.com/article/the-impact-of-weather-on-ka-band-frequencies/.