REWINDING THE CLOCK: PLAYBACK CONCEPTS IN EGS-CC

Tom Moya Schiller
moya@asofterspace.com
A Softer Space

Abstract

Most space missions do not have the luxury of constant visibility of their spacecraft. Housekeeping data is therefore often stored on-board and downlinked at a later time, when a link to a ground station has been established. On the other hand, live data might be sent immediately when a link is being opened, such that a current snapshot of the spacecraft's status is available to the operators at the earliest possible moment. Monitoring and control systems for such missions therefore have to be able to accept this mix of deferred and live data, which can quickly become a non-trivial task when telemetry from the spacecraft is not just displayed, but also causes automatic behavior to arise.

The new EGS-CC (European Ground Systems – Common Core) has been designed from the beginning with this requirement in mind. Not only has it been specified how exactly such deferred data is received and immediately processed by the EGS-CC-based monitoring and control system, but also how it should be integrated into the history of existing data at a later time. In particular, the concept of a "playback session" has been introduced, which iterates over small increments of time, interleaving the live data that was actually received at that time with the deferred data that was generated but not yet received then.

As the implementation of EGS-CC has now progressed to a point at which such designs can actually be seen working in the resulting system, the actual implementation will be outlined. Special attention will be given to existing and future approaches for testing these capabilities and ensuring their adherence to the PUS 15 standard, to increase confidence in the validity of the designed – and implemented – methods. In addition, related concepts such as the full reprocessing of data and the generation time ordering of data already during the live operation of the EGS-CC based system will be shown.

Introduction

EGS-CC is the new basis on which monitoring and control systems will be developed at ESOC, with the ESOC-specific additions being referred to at EGOS-CC. The first such system that is currently under development is the J-MCS – the JUICE Mission Control System for the Jupiter Icy Moons Explorer. [10]

The idea for a common infrastructure on which monitoring and control systems for several missions can be based is not new. Already the MSSS, which was conceived in the mid 1970s [4] and used from 1984 until 1996 [6] was based on this idea. This was followed by SCOS 1 in 1984, SCOS-II in 1992 [3], and SCOS-2000 in 2002. These systems are also used outside of ESOC, such as the SCOS-2000-based GECCOS at DLR being used for TerraSAR-X. [7]

EGS-CC will be used for various kinds of missions as well. A concern of many satellite missions is that they do not have constant visibility, meaning that there is no constant link between the ground and the space segment. In case of missions such as GOCE [5], data is therefore stored on board and downlinked upon request using PUS 15. [1]

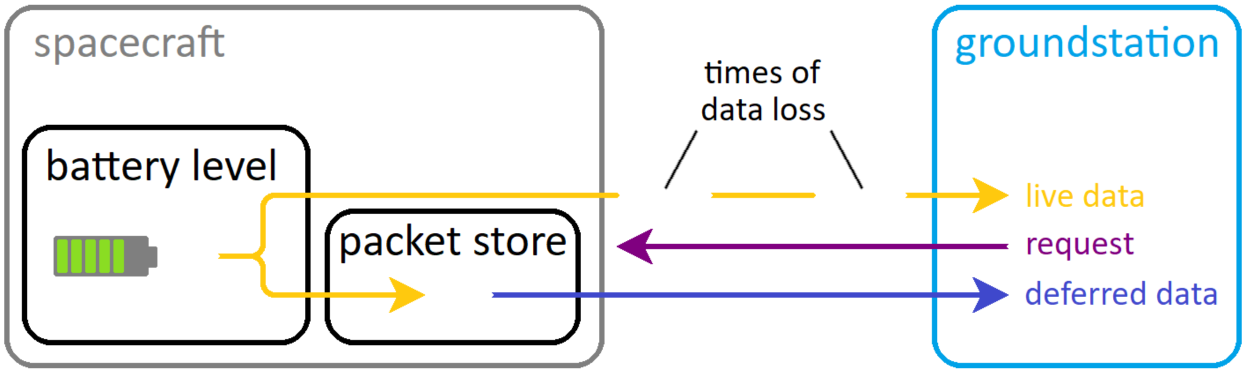


Fig. 1 Some live data is lost between space and ground segments, therefore live data is stored and later downlinked as deferred data.

When this deferred data arrives on ground, the MCS has to handle it somehow such that it can be properly integrated with the previously received live data and shown to the operator in a consolidated form. The process of playing through the already handled live data and the newly received deferred data is referred to as "playback."

Requirements

The main requirement on the MCS is to show the deferred data such that an operator can view the complete history of a parameter, rather than just the parts of the history for which live data was available. It is not enough to just show this data – e.g. in a plot view – but in addition it also needs to be fully processed. The reason for this is that the entire behavior of the MCS might be affected by the incoming data, and upon reception of the deferred data it might become clear that actually a different behavior would have been appropriate at an earlier time.

A scenario in which this handling of deferred data can be understood is shown in Figure 2. Here, the live data that was received did not trigger any alarms, as all received data was within the defined limits. However, when the deferred data is processed it becomes clear that there was actually a time during with the monitored parameter was not inside the predefined limits, and an alarm should have been raised.

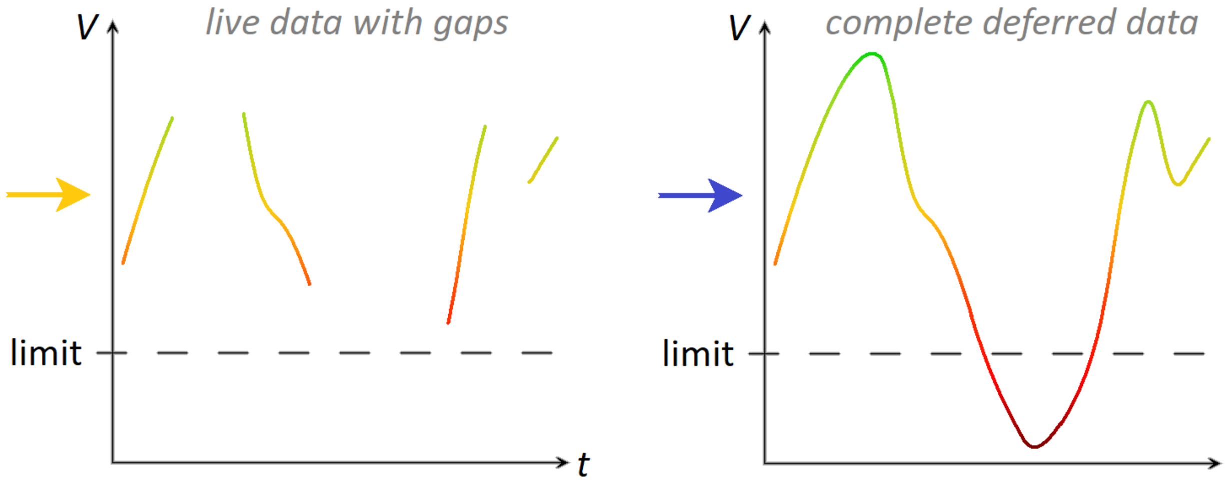


Fig. 2 Live data has no values out of limits, deferred data does

Furthermore, not all deferred data just represents parameter values. Another category of data that can be downlinked from the spacecraft are PUS S1 reports which inform the operator about the success or failure of executing previously sent telecommands. These verifications also need to be shown to the user in some form, but will not usually be shown as part of parameter plots.

At ESOC it is also required that the playback should by default be executed continuously alongside the regular live processing. Such ongoing playback happens until a certain packet is received from the spacecraft, which signifies that all the data before that packet is complete for a certain time interval, such that no data arriving later is still falling into the same time window. It should also be possible to deliberately play back a certain time window using a certain set of data, but it should not be necessary to always manually start playback processes.

Finally, it is required that the system should also allow the reprocessing of data. This is conceptionally different from the main playback processing but is technically very similar. While playback concerns the integration of older deferred data, reprocessing concerns another run through existing data, potentially even just live data without any deferred data at all. The reason why it might be desired to reprocess data is that the MCS itself may have changed since the data was originally processed, such that new processing can reveal new information. One such scenario would be changed tailoring data in which new limits or new synthetic parameters have been defined, such that a reprocessing of all historical data can then use these new definitions.

Concepts

To understand the implementation of the playback functionality in EGS-CC, we first of all have to understand the concept of sessions in EGS-CC – and the differences between the ones that have so far been defined.

A session in EGS-CC is abstract concept that represents a collection of components and their shared context. Such a session is not necessarily represented one to one in the actual technical layout of the system – e.g. there could be components belonging to different sessions running on the same machine, and there could be components belonging to the same session running on different machines. Instead, this concept is achieved by components deliberately joining into the shared context that is offered by a particular session due to their containing application joining into that session. An application can start directly as part of a session – such as a backend application started to host components of a particular session – or can join into a session later on – such as a user interface application joining into a session upon user input.

Each such session represents an isolated instance of EGS-CC. Unless deliberately configured otherwise, the components of one session are fully isolated from the components of other sessions, and there is no limit imposed on how many sessions can be executed within an EGS-CC based system. As each session has its own configuration and tailoring, each can be seen as a full monitoring and control system on its own, potentially serving completely different use-cases and missions. In short, each session of an EGS-CC based system is equivalent to a full SCOS-2000 based system.

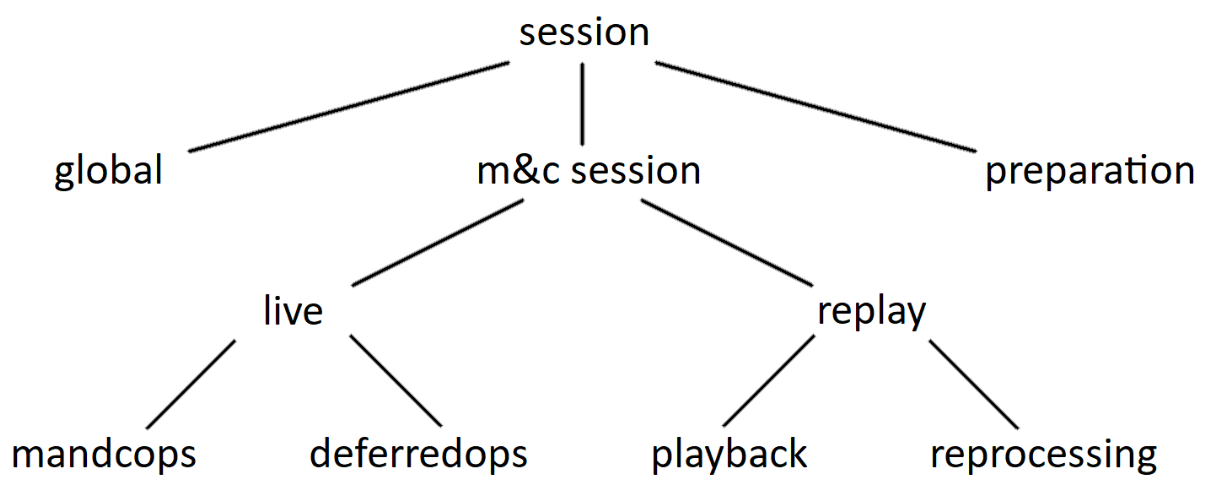


Fig. 3 Overview of the different kinds of sessions in EGS-CC.

The actual kinds of sessions that have been defined so far are:

- The *global* session, which corresponds to the outermost context from which all other sessions are launched
- The *preparation* session, which is a session dedicated to the preparation of tailoring data for actual monitoring and control sessions
- The *mandcops* session, which is a live session for use within a system that has constant visibility with a spacecraft or simulator
- The *deferredops* session, which is a live session just like the mandcops one, but is configured to expect deferred data to arrive at a later time
- The *playback* session, which is a dedicated session to replay the deferred data that was received from the spacecraft at an earlier time
- The *reprocessing* session, which replays all data for use with an updated system e.g. with updated tailoring containing new synthetic parameters

We refer to both mandcops and deferredops as “live” sessions as they handle live monitoring data and control the spacecraft live. On the other hand, as both playback and reprocessing just replay existing data and do not control anything in live mode, we refer to them as “replay” sessions.

As mentioned before, the EGS-CC is just the basis for any real MCS. Therefore, to build an actual MCS on top, even new additional sessions can be defined, or some of the predefined ones can be left out.

Implementation

Having introduced the concept of live and replay sessions, we can now investigate how the data actually behaves in an EGS-CC based MCS.

Live data goes into the live session and is shown in the UIF connected to this session. This includes both the current state of a parameter, as well as all of its history – however, all just based on the live data, which may have gaps.

To initiate the download of deferred data from the spacecraft, an activity is invoked in the live session. This can be done based on user input in the UIF, but also based on various kinds of internal or external sources, such as events defined within EGS-CC or scripts created in EKSE. Upon invocation of the activity, a telecommand is sent to the spacecraft, which is instructed to downlink the deferred data from its onboard packet store.

The deferred data arriving from the spacecraft then also enters the live session, as only the live session is connected to the actual groundstation, simulated spacecraft or other external source. In the live session the playback data is for the most part just archived and not further processed. A dedicated playback session takes the deferred data from the archive and processes it to construct a complete historical view. Upon configuration the playback session can also take the live data in addition to the deferred data and construct the parameter history based on both of them.

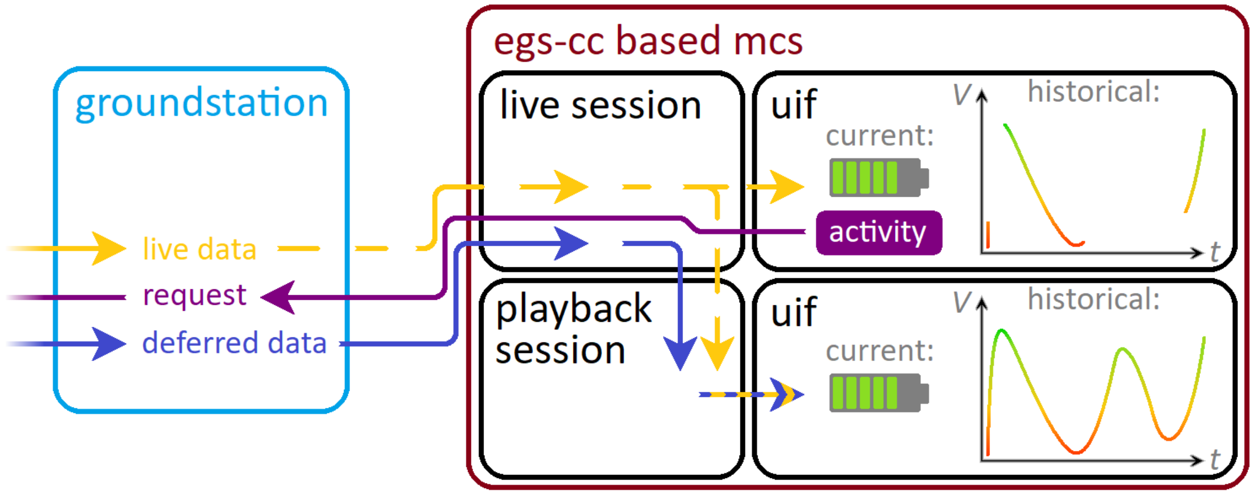


Fig. 4 Basic understanding of live and deferred data flow in EGS-CC.

As shown before, each session in EGS-CC is basically a complete MCS on its own, such that the live and playback sessions represent two mainly independent systems that partially work with the same data. The previous MCS generation at ESOC was handling this differently, as SCOS-2000 had a TM Replayer inside the live system, instead of creating a completely independent system for the replay of TM packets. [2]

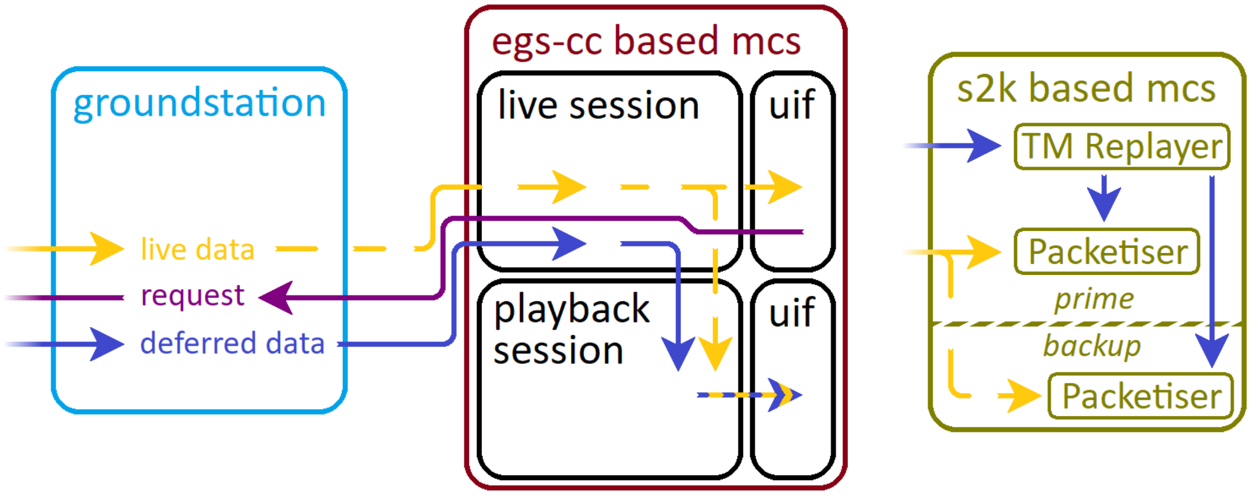


Fig. 5 Live and deferred data flow in EGS-CC compared to SCOS-2000.

The abstractly shown flow of live and deferred data in an EGS-CC based MCS will now be illuminated in more detail. However, a lot of components and interactions are still being left out such that the main concepts can be more clearly shown.

Considering the live data, it is first handled by the Source Data Access component (SDA), which stores it in the Archive (ARC.) From SDA the data goes to the Parameter Extraction component (PEX) which extracts the actual parameter values from the packets and injects them into the Monitoring and Control Model (MCM.) The MCM contains the current live state of the parameter, which can be shown in a UIF connected to this live session.

When a user opens a UDD in the UIF, such as a plot, it starts showing live data. To also see the history of a parameter, archived data can be requested, which however still only shows the live data as it was received – including potential gaps in times without an active TM link.

Deferred data also first enters the system through the live session, as mentioned before. This again happens via SDA, which puts the data into the archive.

The playback session is currently comprised of the same components as the live session. Several components, such as AUT, will in the future be left out of the playback session as they are not actually needed there. Components that are also deployed in the playback session involve ARC, SDA, PEX, and MCM, with the role of SDA being even more important in playback than it was in the live session. In fact, SDA is the main orchestrator of the playback process. [9]

SDA reads the deployment plan, which contains specific information in the playback session which is not present in the live session, therefore differentiating between them. The information that is read out are the `CoordinatedDeliveryConfiguration` and the `DeliveryGroup`. In a future release, this information might actually move over to the system configuration.

PEX also reads the deployment plan, in particular the `DeliveryGroup`, and registers itself with SDA as playback listener. Other listeners can register themselves too, if more are present – e.g. in the future MSM might become such a playback listener. There also exists an EKSE script that can be used as listener, which just outputs the calls that are made to it on the command line for testing purposes.

SDA now iterates through time in short time steps, telling all the listeners about the current step.

The listeners request the actual data within the given time step from SDA, which requests it from ARC. The two archive components in live and playback to not duplicate the same data, but instead both use the same underlying Hadoop HBase database from which they retrieve it. The data now flows from ARC via SDA through PEX and MCM all the way to the UIF, where it is displayed. Once the listener is done with a time slice, it notifies SDA, which starts the next time slice as soon as all listeners have reported back that they are done.

For each time step, the SDA → PEX → SDA → ARC → SDA → PEX loop is repeated, until the historical display is complete.

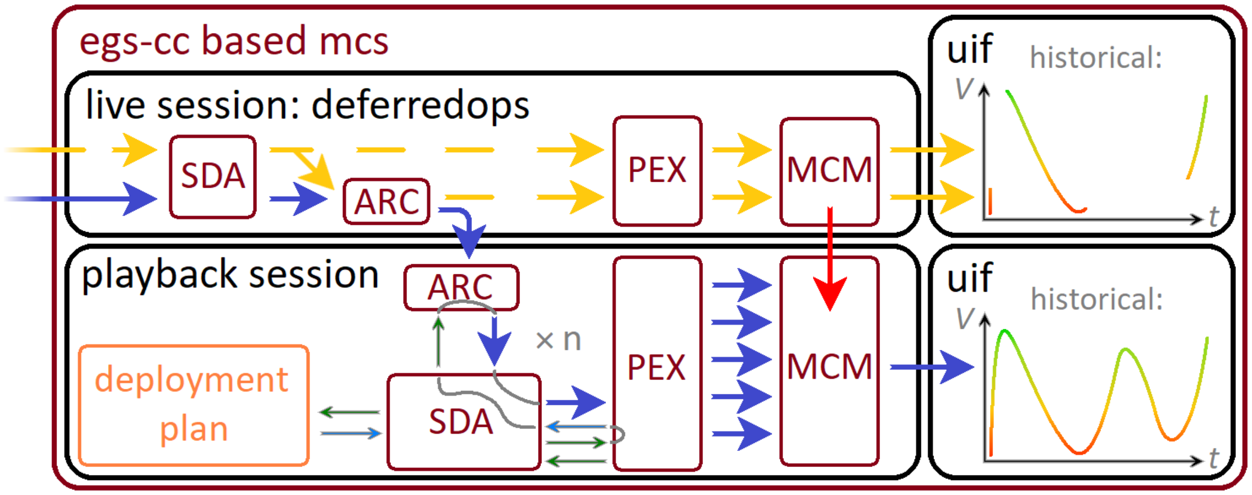


Fig. 6 Detailed view of deferred parameter value data flow in EGS-CC.

It might seem unexpected that SDA informs all listeners of the current time slice only for the listeners to immediately return to SDA and ask it for the data within the time slice. However, this does make sense as listeners might conceivably do other things in the future, e.g. request data from external sources for that same time slice. Other approaches have been considered, such as letting SDA directly deliver the data within the increment, but these would lead to problems with distributed systems in which not all listeners exist within the same application as the SDA that is orchestrating the session.

Furthermore, it has been realized that the approach the listeners are using for fetching data from SDA in this case is actually quite impactful. During early testing a listener was using a subscription-based approach to fetch the data, which is introducing a lot of overhead into the whole process. Instead, data needs to be fetched using simple retrieval calls from SDA in this case.

Having shown how deferred parameter values are propagated through the live and playback sessions to finally be made available to the user, we will now inspect how deferred S1 verifications are handled.

As would be expected, live S1 verifications are directly handled in the live session. They flow through SDA and MSM, setting the correct stages in the MCM from where they are shown in the user interface.

Counter-intuitively, this approach is also true for the deferred S1 verifications – in particular, they are also sent from the live SDA to the live MSM and injected into the live MCM, however clearly identified as playback stages. The main impact this has from a user's point of view is that playback S1 reports can already be inspected while connected to a live session, while playback parameter values can only be seen when connected to a dedicated playback session.

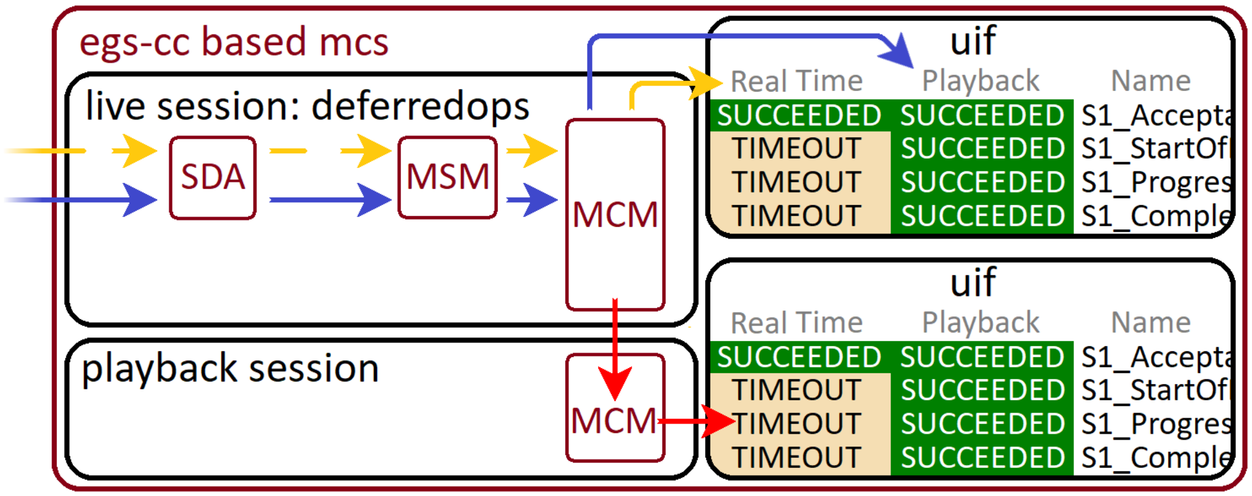


Fig. 7 Detailed view of deferred S1 report data flow in EGS-CC.

In the playback UIF, the verification reports are shown by bringing their state over from the live MCM into the playback MCM through the "MCM Input archive," which furthermore also handles activity invocations and other data. [8]

Overall, the discussed implementation approach leads to the following flow of live and deferred data through the EGS-CC based MCS:

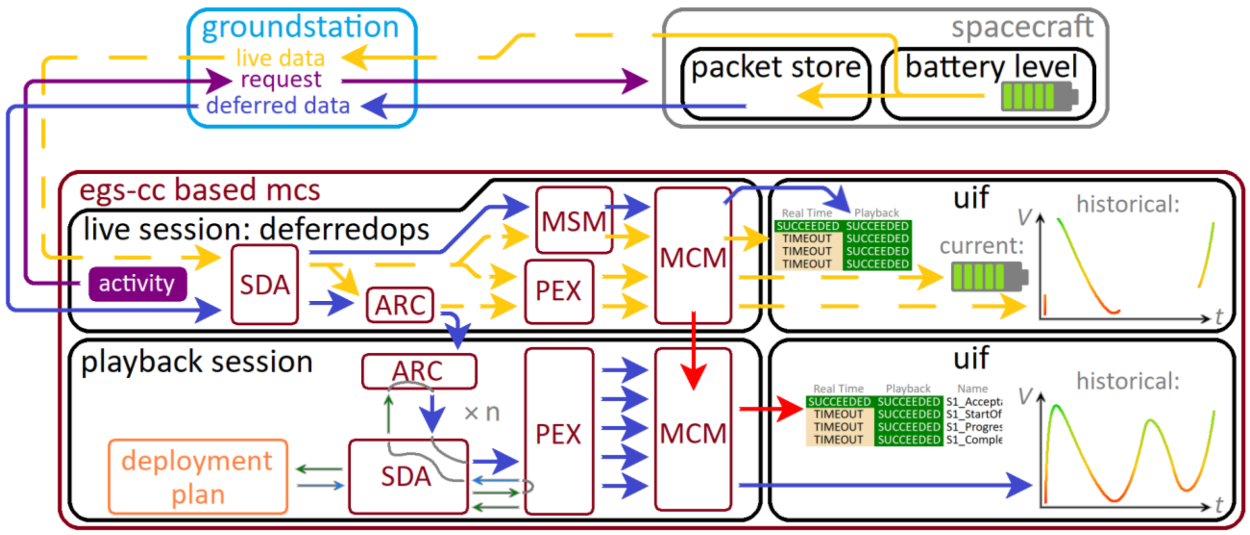


Fig. 8 Complete overview of live and deferred data flow in EGS-CC.

Test Approach

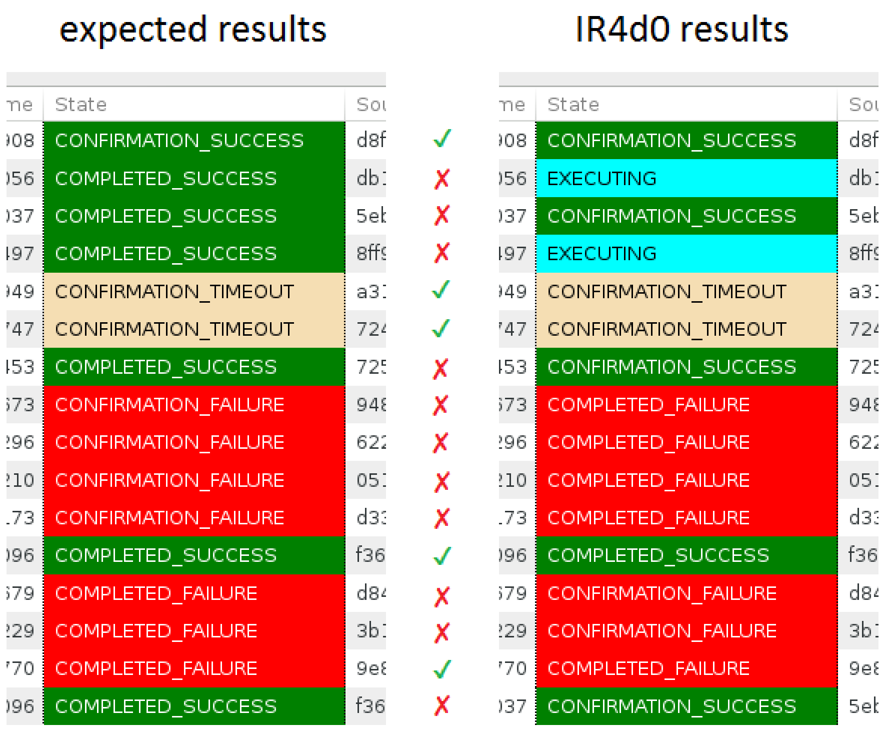
To figure out the current status of the playback implementation in EGS-CC, several tests have been created as part of the Scenario Based Validation of EGS-CC. [11]

The first test only uses the deferredops session and verifies that several different telecommand verification scenarios behave as expected. In particular, this means that a telecommand is sent 16 times, and each time a different kind of verification is received – sometimes no verification is received at all, sometimes only live verifications are received, sometimes only deferred verifications, sometimes both, sometimes the verifications verify successful execution while sometimes they verify a failure on board, etc.

For each of the 16 test cases details on which behavior would be expected have been included, such that it can be checked whether the EGS-CC based system under test behaves exactly as expected, slightly differently, or completely wrong. Based on these results, internal SPRs have been raised which are being worked on such that future releases of EGS-CC will behave more correctly.

To facilitate such a clearly defined test, a test script has been provided which sends in a reproducible manner exactly the desired telecommands and the desired S1 verifications. This ensures that only problems in EGS-CC can lead to deviations from the expected results, instead of a potentially quite complex simulator also possibly introducing unexpected behavior. The test script exists both as EKSE script and as regular EGS-CC Groovy script which can be started from an activity.

Overall, in the current EGS-CC release IR4d0 five out of the sixteen defined test cases are working exactly as expected, with the rest showing minor or major deviations from the expected results:



me	State	Sou		me	State	Sou
08	CONFIRMATION_SUCCESS	d8f	✓	08	CONFIRMATION_SUCCESS	d8f
56	COMPLETED_SUCCESS	db	✗	56	EXECUTING	db
37	COMPLETED_SUCCESS	5e	✗	37	CONFIRMATION_SUCCESS	5e
97	COMPLETED_SUCCESS	8ff	✗	97	EXECUTING	8ff
49	CONFIRMATION_TIMEOUT	a3	✓	49	CONFIRMATION_TIMEOUT	a3
47	CONFIRMATION_TIMEOUT	72	✓	47	CONFIRMATION_TIMEOUT	72
53	COMPLETED_SUCCESS	72	✗	53	CONFIRMATION_SUCCESS	72
73	CONFIRMATION_FAILURE	94	✗	73	COMPLETED_FAILURE	94
96	CONFIRMATION_FAILURE	62	✗	96	COMPLETED_FAILURE	62
10	CONFIRMATION_FAILURE	05	✗	10	COMPLETED_FAILURE	05
73	CONFIRMATION_FAILURE	d3	✗	73	COMPLETED_FAILURE	d3
96	COMPLETED_SUCCESS	f36	✓	96	COMPLETED_SUCCESS	f36
79	COMPLETED_FAILURE	d8	✗	79	CONFIRMATION_FAILURE	d8
29	COMPLETED_FAILURE	3b	✗	29	CONFIRMATION_FAILURE	3b
70	COMPLETED_FAILURE	9e	✓	70	COMPLETED_FAILURE	9e
96	COMPLETED_SUCCESS	f36	✗	37	CONFIRMATION_SUCCESS	5e

Fig. 9 Current test results for playback S1 verifications in EGS-CC.

A second test has also been defined, which serves to ensure that the approach of playing back deferred data within a playback session works as expected. This test again uses an EKSE / EGS-CC Groovy script for its execution.

At first, this script injects parameter values as if they were live telemetry packets arriving from a spacecraft for which constant visibility is not given, with large blocks of data for windows of visibility, and large blocks of data missing in times of an interrupted TM link.

At a later point, the deferred data that was stored on board the simulated spacecraft is sent into the system, but not directly injected into the live TM chain. Finally, the user executing the test starts a playback session and can then compare a UDD plot containing the used parameter of the live and playback sessions, seeing that the missing data in the live session has been filled via the playback process in the playback session:

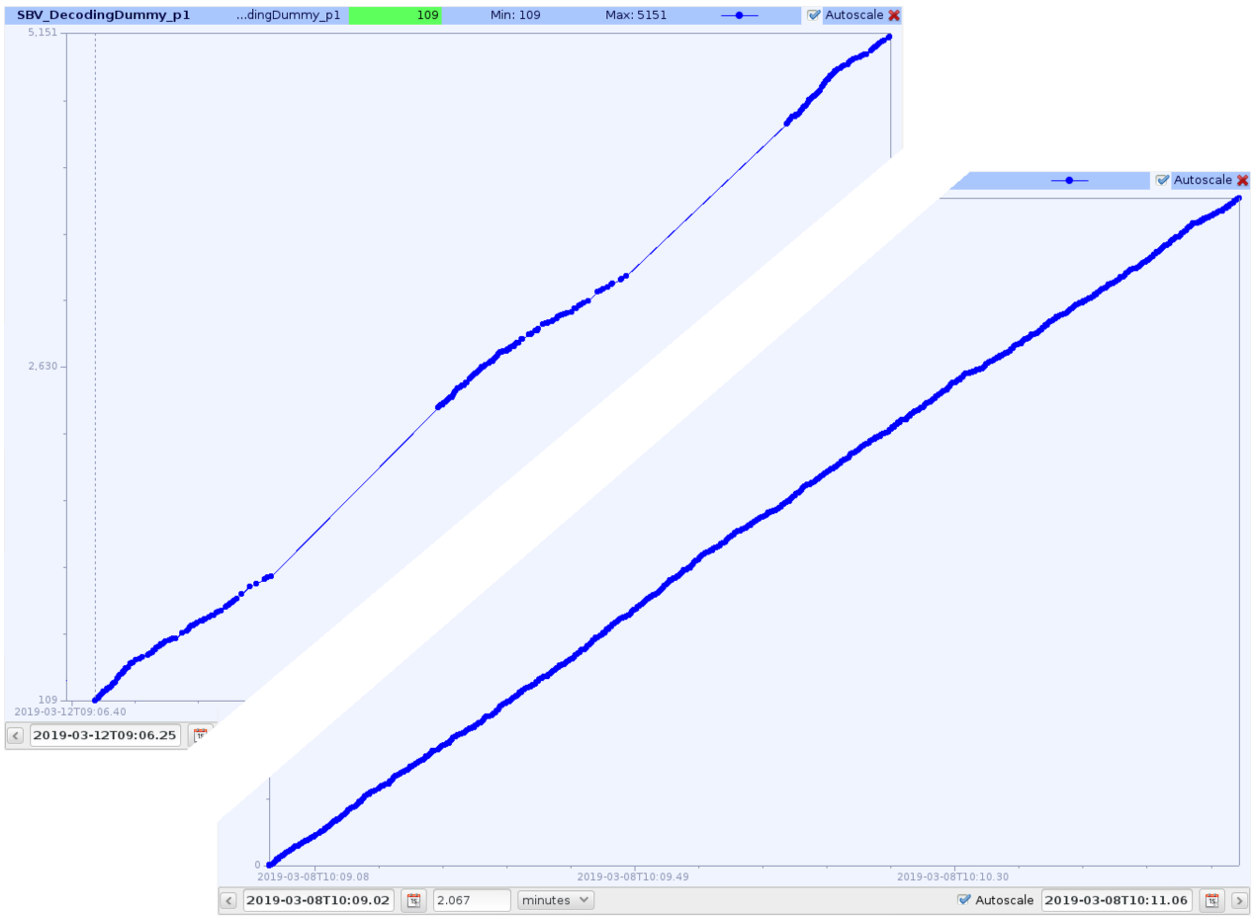


Fig. 10 Current test results for deferred parameter value playback in EGS-CC, showing a live UDD on the left and a playback UDD on the right.

To make problems with the system under test more easily recognizable, the injected parameter furthermore is not just injected in large blocks by the test script, but is actually given values that are increasing with the generation time of the parameter value. Therefore, the generated line should always have a positive slope – if any values appear with a lower value than previous values, this shows immediately that something went wrong. Furthermore, parameters are not injected exactly in their generation time order, but are actually first perturbed on a local scale, such that they are slightly our of order. This also helps more quickly recognizing system misbehavior, as it allowed e.g. finding a problem with the actual display of the data inside the UIF.

Overall, as of IR4d0 this test is running well for one-off playback processes, with the continuous playback as required by ESOC still being under development.

Challenges

Just as with the rest of EGS-CC, the main challenge for the implementation is the coordination of the behavior of the different components involved in performing the playback operations. Changes to the architecture have caused different components to be responsible for different tasks – such as the playback orchestrator role which moved from RUM over to SDA.

In a similar vein, when problems are detected during testing it is not necessarily straightforward to identify the responsible component. It therefore happens that a problem at system level is clearly known, but fixing it requires a lot of initial effort in actually discovering the component that contains the underlying problem. Even more difficult can be decisions about which component should be tasked to make a change to solve a problem, if several components each could be tasked to make such a change.

Conclusion

The EGS-CC is still being implemented and has not even seen its first official release as actually finished software yet. Having clearly defined tests for the playback process, which is a fairly advanced concept, at such an early stage means that even larger architectural changes can still be made before it is too late based on the test results. On the other hand, it means that the tested system is a "moving target", which might still change so much that tests will have to be adapted several times. Overall it seems to have been a good decision to already start working on these advanced concepts now, instead of waiting for a final product first.

The creation of different sessions for handling live and deferred data might increase the learning curve for new users, especially due to some deferred data being handled in live anyway. On the other hand, it clearly demonstrates the flexibility and adaptability of EGS-CC, for which many more sessions might be defined in the future which have not even been conceived of yet.

Overall, the current playback capability of EGS-CC is sufficient for basic use cases. Further improvements are currently being made, such that at the Release 1 of EGS-CC most requirements the various future missions could have should be supported out of the box.

Appendix
Acronym List

ARC:	Archive (kernel component of EGS-CC)
AUT:	Automation (kernel component of EGS-CC)
EGOS-CC:	ESA Ground Operation System – Common Core (M&C system on top of EGS-CC)
EGS-CC:	European Ground System – Common Core
EKSE:	EGS-CC Karaf Shell Extension
MCM:	Monitoring and Control Model (kernel component of EGS-CC)
MCS:	Montoring and Control System
MSM:	M&C Service Models (reference implementation component of EGS-CC)
MSSS:	Multi-Satellite Support System (predecessor of SCOS 1)
PEX:	Parameter Extraction (reference implementation component of EGS-CC)
PUS:	Packet Utilization Standard
S2k:	*see* SCOS-2000
SCOS 1:	Spacecraft Control Operations System 1 (predecessor of SCOS-II)
SCOS-II:	Spacecraft Control Operations System II (predecessor of SCOS-2000)
SCOS-2000:	Satellite Control and Operation System 2000 (predecessor of EGS-CC)
SDA:	Source Data Access (kernel component of EGS-CC)
S1:	PUS Service 1 (telecommand verification)
SPR:	Software Problem Report
TC:	Telecommand
TM:	Telemetry
UDD:	User-Defined Display
UIF:	User Interface (reference implementation component of EGS-CC)

References

[1] ECSS, "Space Engineering: Telemetry and Telecommand Packet Utilization Standard", ECSS-E-ST-70-41C, April 2016

[2] Merri, M., Pecchioli, M., Vázquez, R., Prieto, J., "The CRYOSAT/GOCE Mission Control System", *SpaceOps 2002 Conference*, SpaceOps Conferences, (AIAA 2002-T3-29) doi: 10.2514/6.2002-T3-29

[3] Jones, M., Head, N.C., Keyte, K., Howard, P., Lynenskjold, S., "SCOS II - ESAS New Generation of Mission Control System", *Third International Symposium on Space Mission Operations and Ground Data Systems*, NASA - Goddard Space Flight Center, Part 1, 1994, p. p 641-654

[4] Ercolani, A., Delhaise, F., Guerrucci, D., Jones, M., Kerr, G. W., Merri, M., Pignede, M., Ponz, D., Reggestad, V. , "Use of reusable software for cost effective development of Mission Control Systems", *SpaceOps 2004 Conference*, SpaceOps Conferences, (AIAA 2004-288-137)
doi: 10.2514/6.2004-288-137

[5] Steiger, C., Piñeiro, J., Emanuelli, P. P., "Operating GOCE, the European Space Agency's Low-flying Gravity Mission", *SpaceOps 2010 Conference*, SpaceOps Conferences, (AIAA 2010-2125)
doi: 10.2514/6.2010-2125

[6] Nestor, P., "SCOS-2000 – ESA's Spacecraft Control for the 21st Century", *GSAW 2003*, Ground System Architectures Workshop

[7] Stangl, C., Lotko, B., Geyer, M. P., Oswald, M., Braun, A., "GECCOS - the new Monitoring and Control System at DLR-GSOC for Space Operations, based on SCOS-2000", *SpaceOps 2014 Conference*, SpaceOps Conferences, (AIAA 2014-1602)
doi: 10.2514/6.2014-1602

[8] EGS-CC System Engineering Team, "Software Specification and Design Document EGS-CC", EGSCC-SYS-SSDD-1000, Issue 3.2, 4th April 2018

[9] EGS-CC Phase C/D Team, "Deferred Data Distribution and Processing", *EGS-CC Phase C/D Working Note*, EGSCC-PCD-WN-1111, Issue 1.4A, 22nd Oct 2018

[10] Berton, J. C., Gómez, E., Widegård, K., Smith, C., Teixeira, L., "The ESOC End-to-End Ground Segment Reference Facility", *SpaceOps 2018 Conference*, SpaceOps Conferences, (AIAA 2018-2513)
doi: 10.2514/6.2018-2513

[11] EGS-CC System Engineering Team, "Scenario Based Validation", EGSCC-SYS-SVS-1003, Issue 2.1, 13th March 2019

There are two kinds of fools:
one says, 'This is old, therefore it is good';
the other says, 'This is new, therefore it is better.'
~ William Ralph Inge