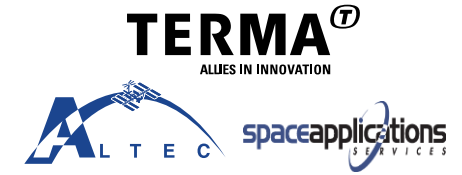


# One Operation Data System Infrastructure for all ESA space assets (OODSI) Study Final Presentation

27-Feb-2023

# Presentation overview

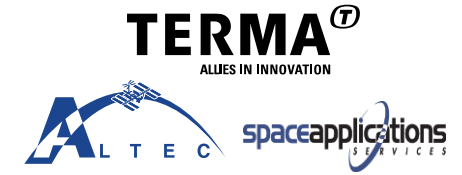
---



1. Study overview
2. Analysis
3. OODSI requirements
4. OODSI architecture design
5. PoC prototype development
6. PoC prototype validation
7. PoC prototype demonstration
8. Results, conclusions and recommendations

# 1. Study overview

# Study team



- Technical Officer: Evridiki Ntagiou
- Technical support: Sebastian Martin, Klara Widegard, Dominik Marszk, MCS-CC/EGS-CC engineers
  
- Study team:
  - Kevin Davies (Terma)
  - Alberto Polverini (Terma)
  - Hamza Faiz (Terma)
  - Diego Bussi (ALTEC)
  - Nicolae Mihalache (Space Applications Services)

- ESA mission types are becoming more varied.
  - Examples: Robotic surface operations (such as rovers); robotic systems (such as the European Robotic Arm); human habitats; active debris removal; in-orbit servicing; human space operations (ISS and Gateway); re-entry missions (such as Space Rider).
- Compared to classical satellite missions, new mission types impact ESOC's ground operations data systems because of:
  - Different operations concepts (for example, different concepts for operations preparation and planning, on-board autonomy, situational awareness, and communications concepts including data relay via other control centres and their assets);
  - Complex, coordinated multi-asset operations involving multiple control centres;
  - Different ground data systems functions needed to support the operations.
- ESOC wants to ensure that its Mission Operations Infrastructure (MOI) will be able to support future mission types.

# Study objectives

1. Analyse and identify requirements and the common functional basis for next generation operational data systems to support distributed, multi-asset operations for future robotic mission types.
2. Devise a harmonised architecture, taking into account existing capabilities of mission operations infrastructure, using EGOS-CC/EGS-CC as the starting point.
3. Validate the result through a representative proof-of-concept demonstrator for a common M&C system for robotic assets.

This study focussed on Mars rover-type missions, in particular the ExoMars RSP (Rosalind Franklin rover) and Mars Sample Return (MSR) Sample Fetch Rover (SFR).

# Study phases and tasks

## Phase 1:

- Task 1: Analysis and specification of requirements for the generic MOI functions of a common M&C basis for future space missions, focussing on robotic rover missions (ExoMars, MSR SFR), and in particular addressing distributed multi-asset operations.
- Task 2: Requirements mapping and gap analysis related to multi-asset operations w.r.t. EGS-CC/EGOS-CC.

## Phase 2:

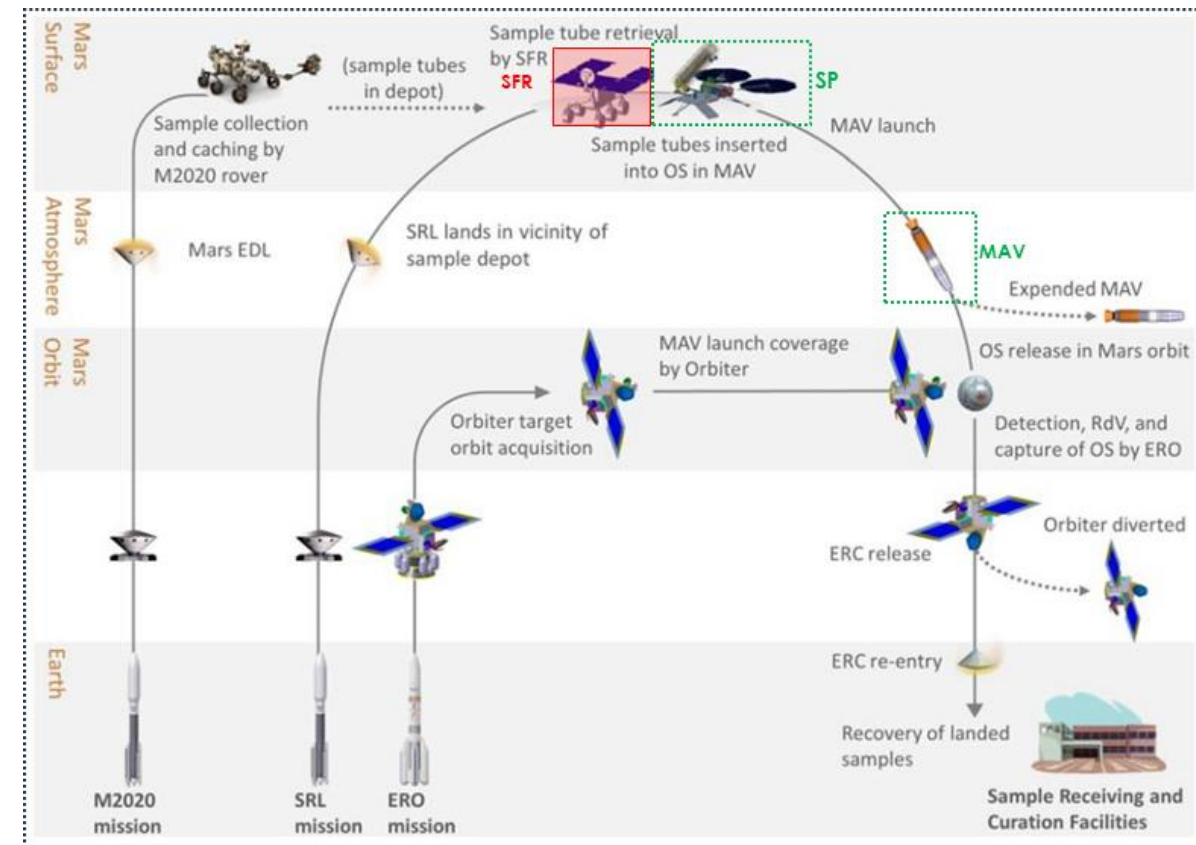
- Task 3: For prototyping preparation, devise an architecture for an infrastructure supporting next generation missions operations based on EGS-CC/EGOS-CC.
- Task 4: Create the Proof-of-Concept prototype based on EGS-CC/EGOS-CC, validated with a representative operations scenario.

## 2. Analysis



# Control system scenarios and functional breakdown analysis (D1)

- Analysis covered:
  - ExoMars RSP (rover and surface platform) mission:
    - ROCC function building blocks
    - ROCC operations concepts
  - MSR (Mars sample return)
    - MSR campaign (NASA Mars-2020, NASA/ESA SRL/SFR, NASA ERO)
    - SFR (Sample Fetch Rover) ground segment elements
  - Other M&C tools
    - 3DROCS, 2DROV, MMI4EXP, MOE



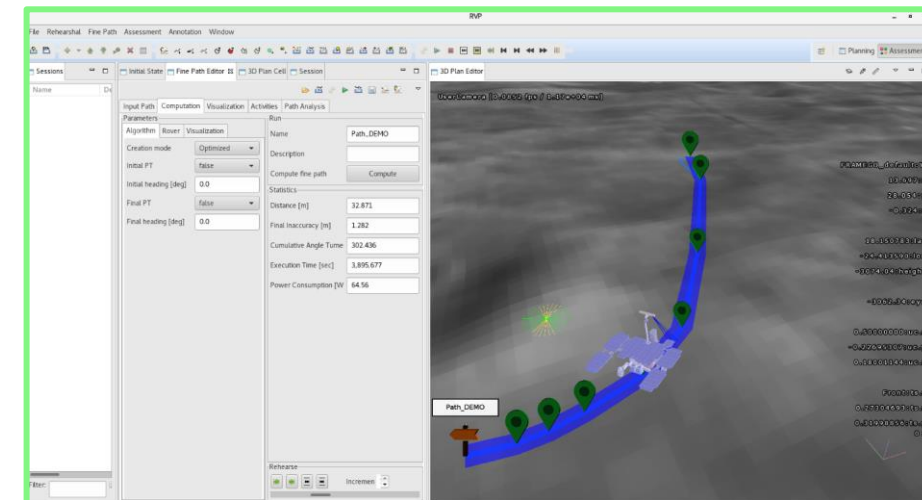
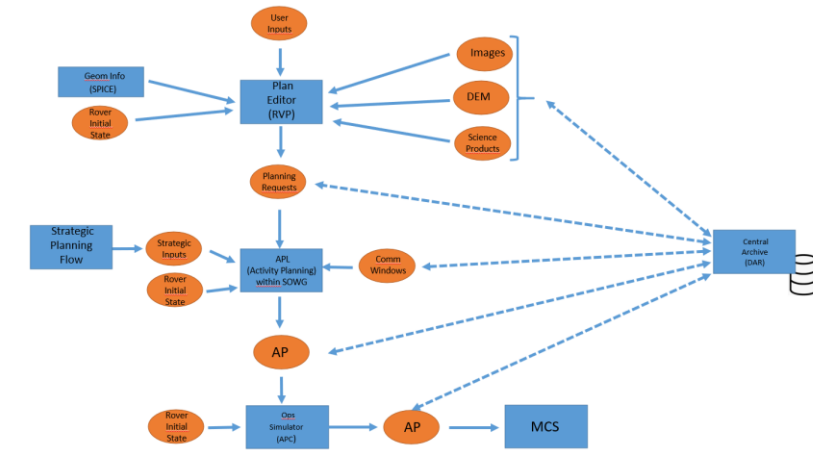
# OODSI robotic M&C system function blocks

- Main building blocks for a robotic mission M&C system:
  - ✓ Distributed Planning and Simulation
  - ✓ Data Acquisition and Processing
  - ✓ Commanding and Uplink
  - ✓ Data Assessment
  - ✓ Data Dissemination
  - ✓ Data Archive

# Collaborative distributed planning and simulation

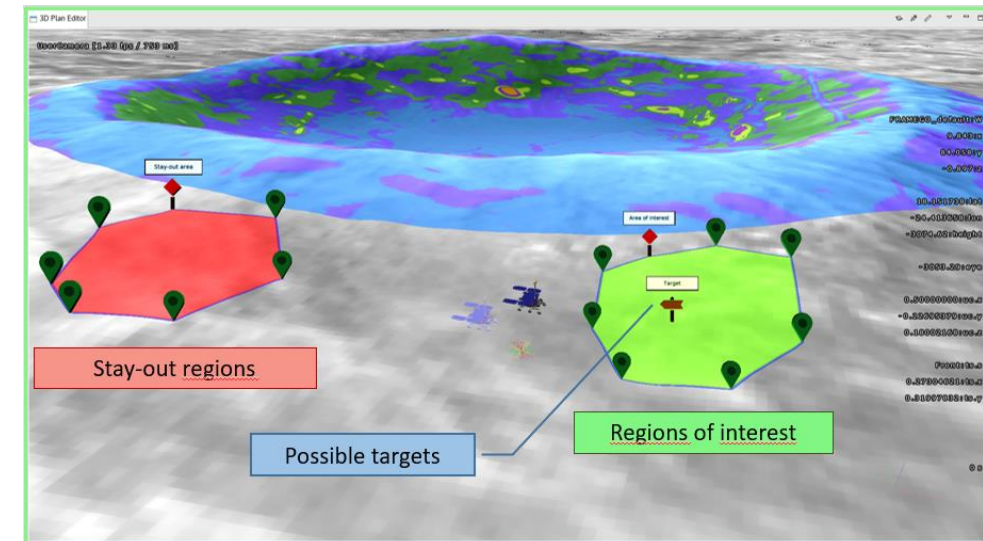
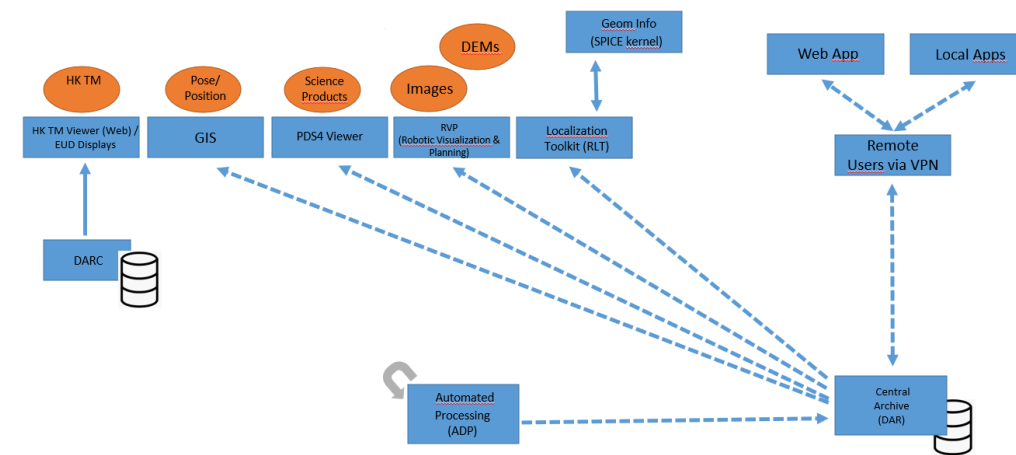
Planning and Simulation is a key element for an MCS designed to support a robotic mission.

- ✓ Multiple teams need to collaborate to construct an Activity Plan for the upcoming days of the mission.
  - ✓ An Activity Plan is a file containing high level instructions for the Rover (i.e. Move, Acquire Image, etc).
- ✓ The system has to be capable of receiving planning requests from other distributed stakeholders, and organise them into a viable plan.
  - ✓ «Viable» means that the plan is within the resource constraints (e.g. time, power, amount of data generated) for those specific days.
- ✓ The Activity Plan is modified iteratively within the planning session until all constraints are respected, taking into account the needed margins.



# Data Assessment

- In addition to standard monitoring capabilities needed by a spacecraft (HK data assessment), a robotic mission needs to provide good situational awareness, i.e. understand where the rover is, what it is around, etc.
- A robotic MCS may need to:
  - ✓ Be capable to import images to reconstruct the environment around the rover.
  - ✓ Use navigation and science products (i.e. project images and elevation models into the scene).
  - ✓ Enhance the original (and orbital) base-map as long as the Rover travels on the surface of the planet with engineering / science products acquired during its journey.
- Furthermore, the Data Assessment may need to:
  - ✓ Perform terrain assessment (slope, rocks size, etc).
  - ✓ Identify targets (for navigation and/or science).
  - ✓ Identify Keep-Out Areas, Region of Interests, etc.
  - ✓ Compute absolute localization.



- In robotics missions, Commanding and Uplink management is mainly an offline, file-based process.
  - Commands are generally built into a binary file to be loaded on the rover for execution.
  - Such a binary file goes through the dedicated communication networks and several nodes, and finally, an Orbiter radiates the content of this file to the Rover.
  - Commanding real time capabilities can be needed for testing and validation (SVT) or for eventual teleoperations.
- This functional block needs also to have the capability to uplink all the auxiliary files needed to support the Activity Plan execution on the rover (Mobility Path, Instruments configuration files, etc), using e.g. CFDP or PUS Service 13.

# “Lessons Learned” from the ExoMars RSP mission

1. Lack of a really “Integrated” M&C system: ROCC is made by several applications, quite different from each other. A unified interface would be easier to learn and more efficient in terms of operations management.
2. There is a total separation from the “Low level commandability” management and the “High level commandability” management made by Activity Plans.
3. Remote users’ capabilities. The remote users experience needs improvement. Not all the applications are conceived as Web Apps, some have to be locally installed on physical machines, making those tools Operating System dependent.
4. Planning process is mainly a manual process with limited automation. Exploitation of machine learning / AI could play a key role in analysing received TM and image/video data and planning operations, especially in a complex environment.
5. Collaborative environment for planning has to be improved.
6. No chat capability provided in any ROCS application. This has been also considered a limitation by scientists during simulations.

## 3. OODSI requirements

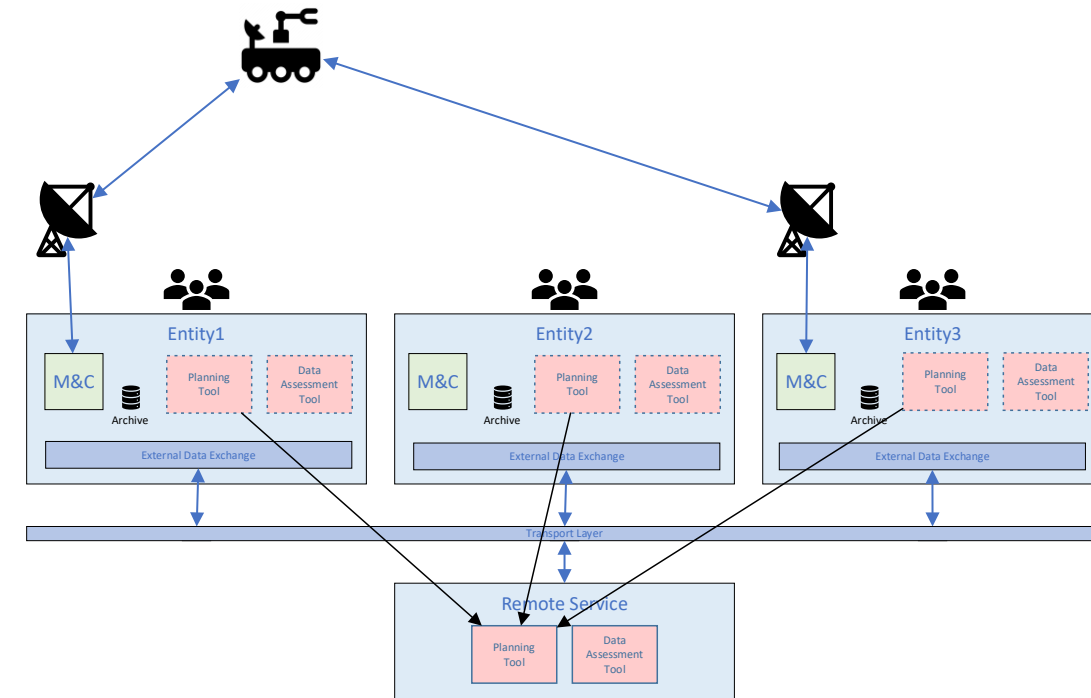
- The OODSI requirements are grouped in the SRS (D2) into function blocks:
  - System level
  - Data acquisition and processing
  - Data assessment
  - Commanding and uplink
  - Communications
  - Data dissemination
  - Planning and simulation
  - Data archive
- OODSI requirements mapping to ESA infrastructure (D3) provides a descriptive assessment of the coverage of the functions of the existing EGS-CC infrastructure.
- Some function blocks are not within the scope of EGS-CC at all (e.g. planning and simulation).



## 4. OODSI architecture design

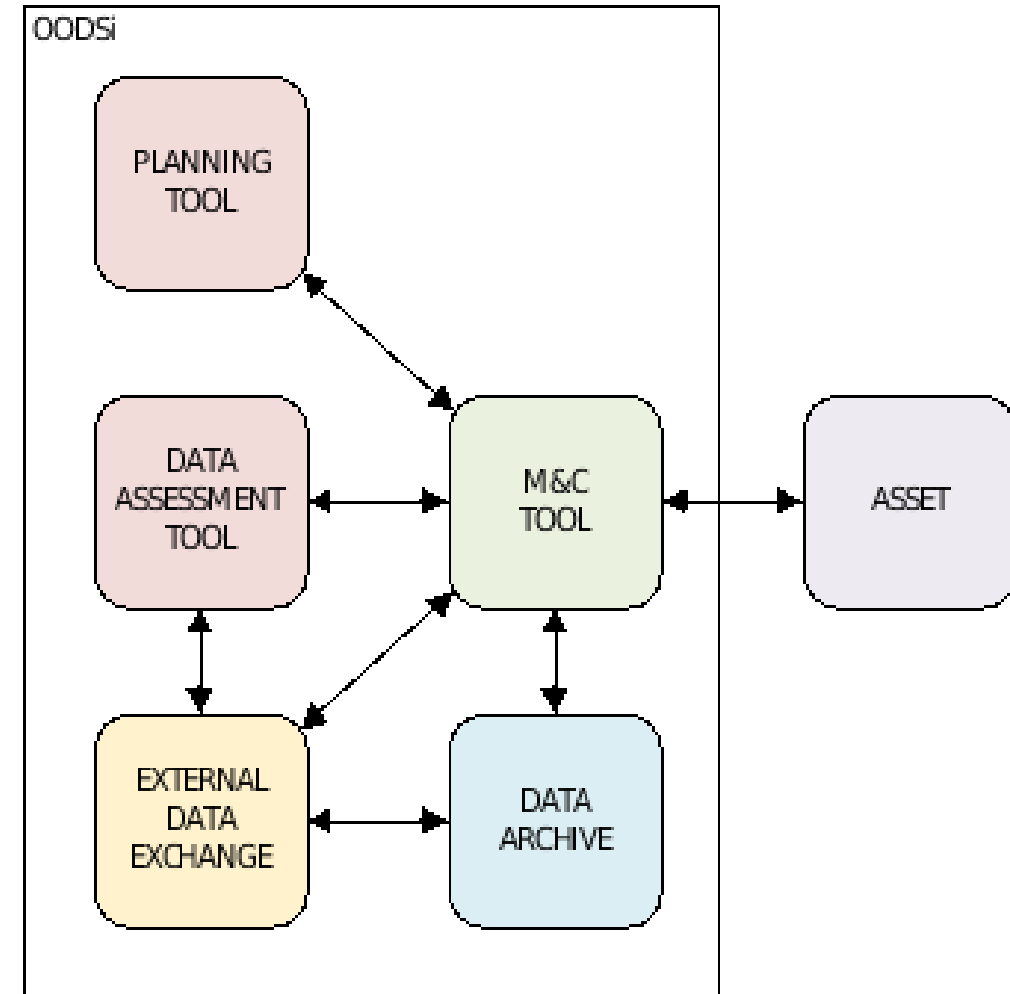
# OODSI system context

- OODSI multi-asset architecture has to support missions with different operations organisations and responsibilities:
  - 1 control centre only with uplink/downlink capability.
  - Multiple control centres with uplink/downlink capability.
  - Centres collaborating with shared tools (planning tool, data assessment tools, etc).
  - Centres collaborating but using their own infrastructure (e.g. for M&C)



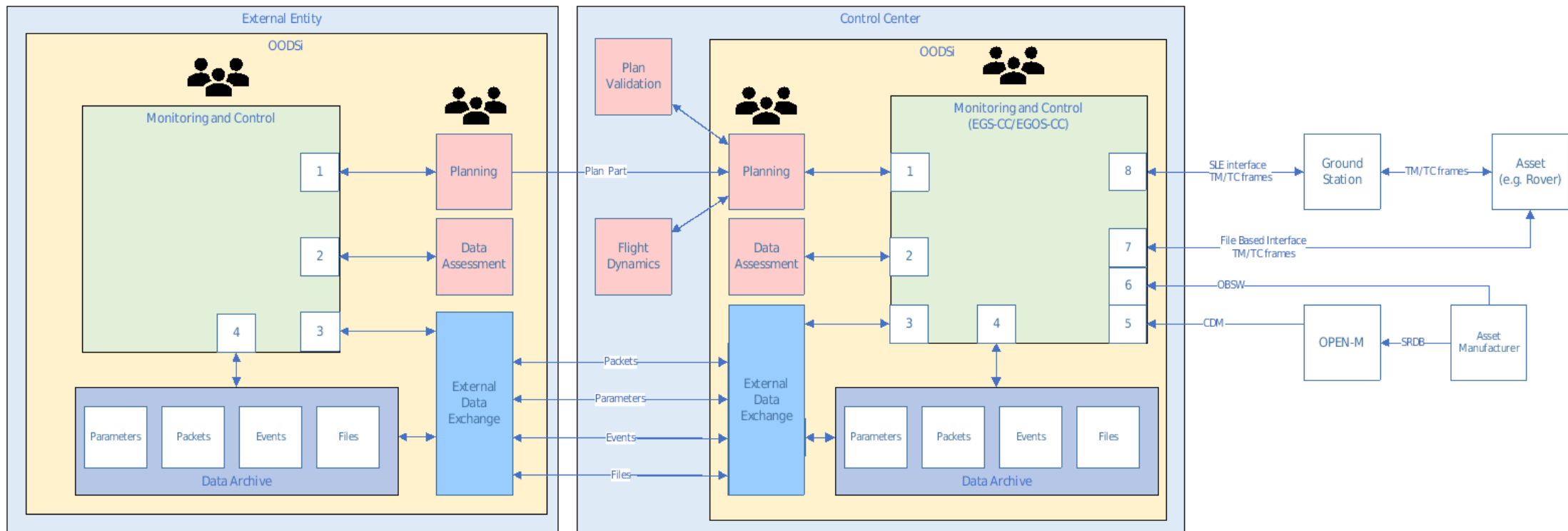
# OODSI top-level components

- Planning Tool
- Data Assessment Tool
- Monitoring And Control Tool
- External Data Exchange
- Data Archive



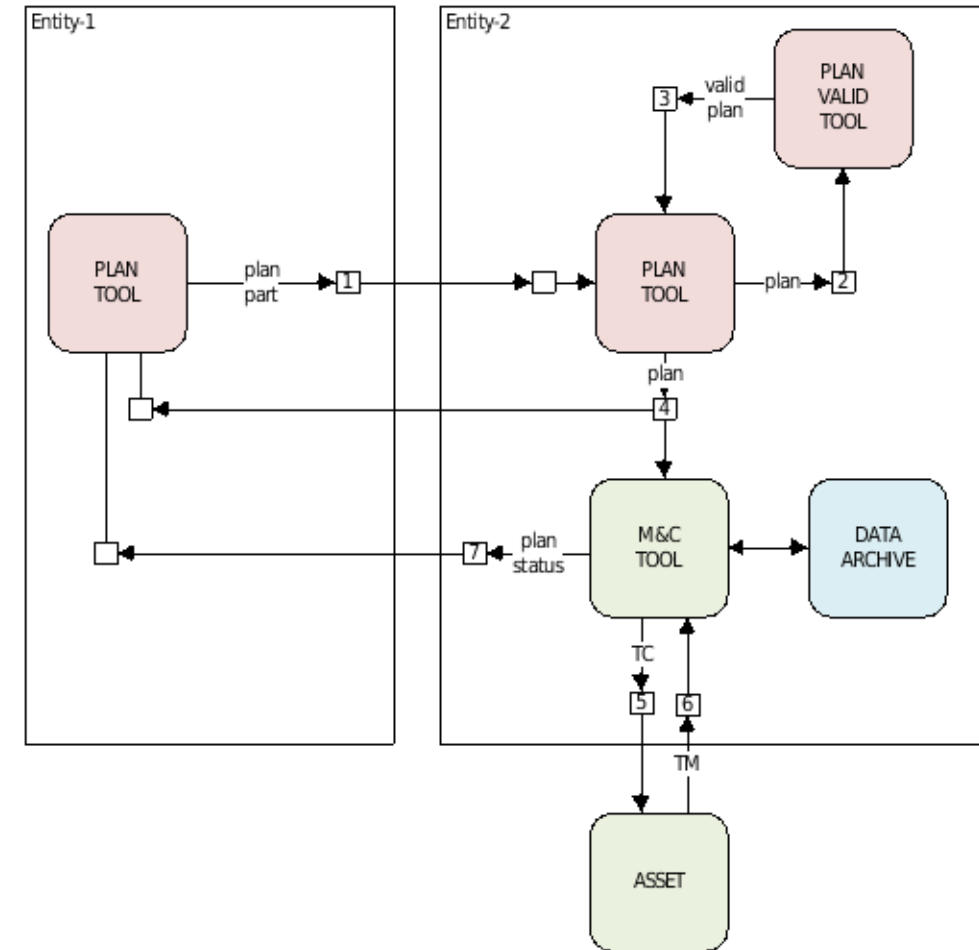
# Example system deployment

- In this example, one control centre has uplink/downlink capability.
- Both control centres have M&C capability for the assets under their responsibility (e.g. platform, instrument, ...)
- Planning and Data Assessment included in the scope of the OODSI.
- Planning and data assessment collaboration through data exchange (files, parameters, packets, events).



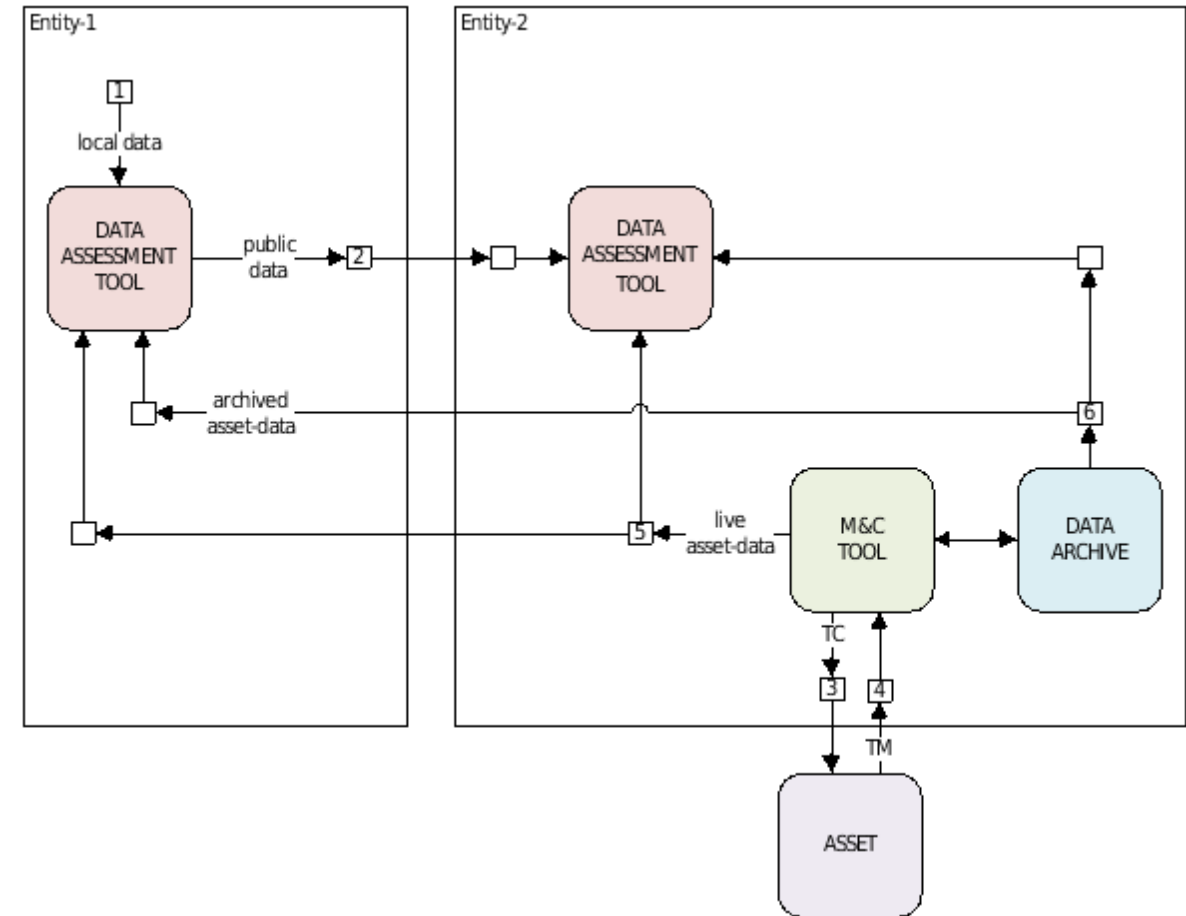
# Distributed planning

- Planning may need to be frequently, e.g. on a daily basis.
- Collaboration in plan preparation may be done in near real-time, with iterative updates checked with the simulator and results assessed immediately.
- OODSI architecture does not define the functions and architecture of the planning tool, only the mechanisms for exchanging data.



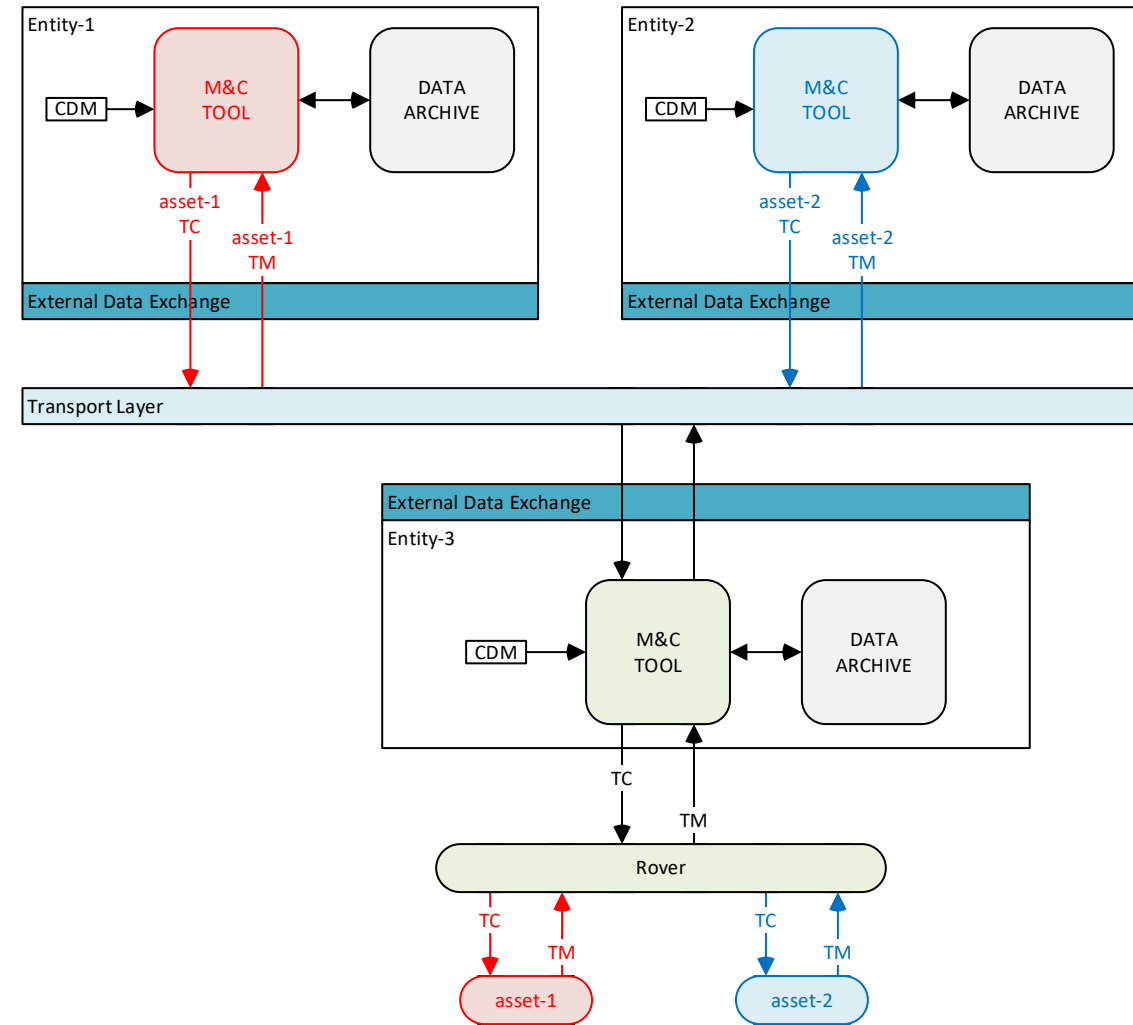
# Distributed data assessment

- Similar to planning, data assessment may be performed continuously after each data dump, e.g. on a daily basis, and collaboratively in near real-time.
- Data assessment may be done independently using local tools at each centre, or using other methods such as remote access tools (e.g. web applications) or desktop sharing tools.
- OODSI architecture does not define the functions and architecture of the data assessment tool, only the mechanisms for exchanging data.



# Distributed commanding

- Distributed commanding can be considered, although such as approach usually goes against operations principles.
- Examples where distributed / shared commanding might be possible include:
  - Teleoperations.
  - Direct commanding of a payload or instrument.
- Distributed commanding involves many potential requirements that have not been explored within the scope of the study, and is therefore is not specifically covered by the OODSI architecture.



OODSI design architecture describes design recommendations for the infrastructure to support future robotic missions.

- Monitoring and Control:
  - Offline file-based interfaces.
  - File-based operations support – file encoding for uplink, file reconstruction from downlink (CFDP & PUS S13)
- Planning:
  - Provision of data exchange for artefacts involved in distributed planning (provision/reception of plan-parts, provision/reception of plan for encoding and transfer to the asset/simulator/validation tool, provision of activity execution reports).
- Data assessment:
  - Provision of data exchange involved in distributed data assessment (monitored parameters from asset or other sources, mission products).
- Data archive:
  - As for classical satellite missions (including access to mission data and artefacts for external entities).

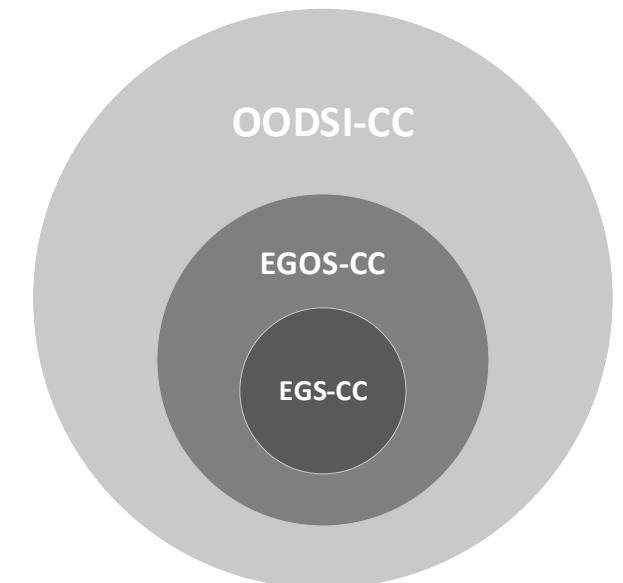


- Information security:
  - CDM Partitioning: Restricting an entity to monitor / control only parts of the system, such as a payload or instrument.
  - Data access security: Integration of (remote) data access (e.g. using MO services) with user authentication and roles & privileges concepts.
- Data exchange between entities:
  - Different transfer mechanism concepts are considered (point-to-point (including web APIs), message bus)
  - Experience shows both concepts are usually needed.
  - Data exchange architectural design recommendations, including MO services:
    - Provide a MO API (based on message bus) for Monitoring/Control. Work with the MO WG to address the limitations of the current MC model.
    - Provide a MO compatible API (based on message bus) for provisioning of packets/frames. The interface can be made MO compliant when the corresponding MO service is fully defined.
    - Provide a MO compatible API (based on message bus) for implementation of the planning service. The interface can be made MO compliant when the MO planning service is fully defined. The API shall allow development of mission specific planning applications but the interface between two applications running in different entities shall be based on MO.
    - Provide a non-MO interface (point-to point) for a) Data retrieval from archive, b. Data catalogue, c) Data description.
    - Provide a (non-MO) object storage provider for sharing/accessing large files (images, etc).

## 5. PoC prototype development

# PoC prototype overview

- OODSI-CC based on MCS-CC v0.3.2.
- Built as OODSI Gitlab project. Runtime deployed as 2 docker containers (mcs-cc, yamcs).
- Preparation done using OPEN-M:
  - ExoMars RSP sessions created and configured (based originally on BepiColombo MCS-CC configuration)
  - Tailoring based on ExoMars rover S2K MIB (TAS v3.15), imported into OPEN-M.
  - CDM exported to MCS-CC.
- Activity plan encoding as PUS S13 TC packets.
- BPF (CLTU) encoding.
- Activity plan status display.
- TM frame file ingestion and processing.
- Mission files reconstruction from PUS S13 TM packets.
- Multiple Control Centre data exchange using MO services:
  - MO services adapters, CMD export to XTCE.

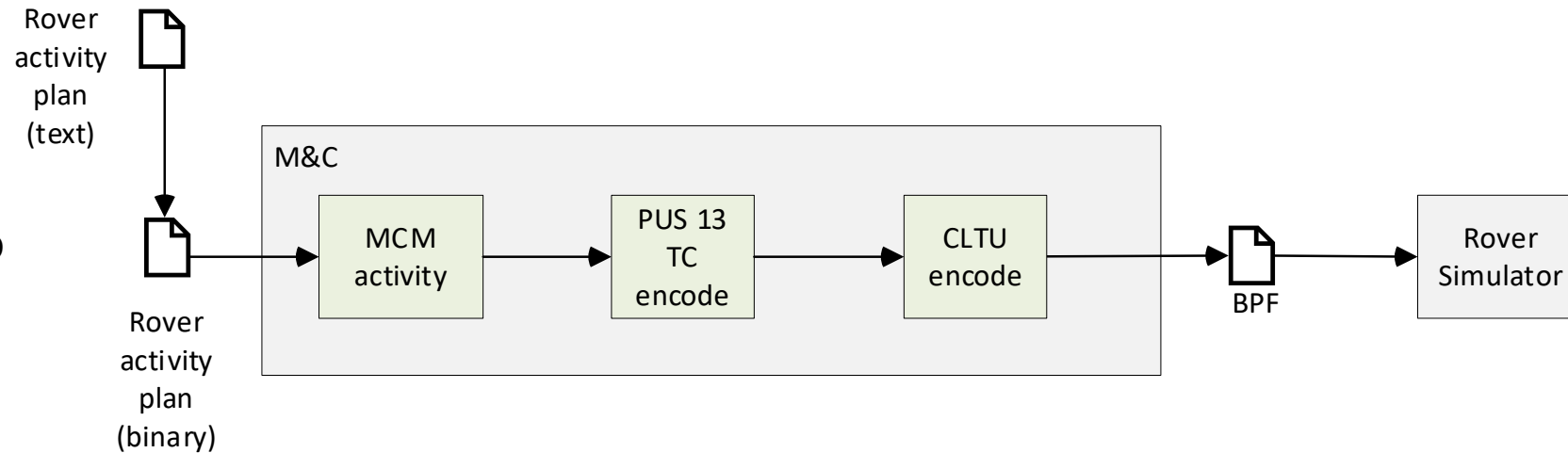


# PoC Prototype Jira backlog

Issue Type	ID	Summary	Status
Story	<u>OODSI-1</u>	Set up dev env & GitLab CI/CD at Terma	Done
Story	<u>OODSI-2</u>	MCS-CC tailoring for ExoMars-RSP (Rover)	Done
Story	<u>OODSI-3</u>	PoC - Offline TC/TM: TMF as input data	Done
Story	<u>OODSI-4</u>	PoC - Offline TC/TM: Activity Plan (binary file) encoding into TC PUS 13 packets	Done
Story	<u>OODSI-5</u>	PoC - Offline TC/TM: BPF as output data	Done
Story	<u>OODSI-6</u>	BPF encoding	Done
Story	<u>OODSI-7</u>	PoC - Offline TC/TM: PUS 13 Files reconstruction	Done
Sub-task	<u>OODSI-8</u>	OODSI-4 Activity Plan (text) -> Activity Plan (binary) conversion	Done
Task	<u>OODSI-13</u>	MPS tool to monitor the Rover Activity Plan execution	Done
Story	<u>OODSI-14</u>	Data exchange - CDM to XTCE export	Done
Story	<u>OODSI-15</u>	Implementation of MO adapter for Yamcs	Done
Story	<u>OODSI-16</u>	C2LOCO MO Adapter integration	Done
Task	<u>OODSI-17</u>	OODSI PoC Prototype User Manual	Done
Task	<u>OODSI-18</u>	OODSI PoC Prototype Software Validation Test Document (SVTD)	Done
Task	<u>OODSI-19</u>	OODSI PoC Prototype Software Release Document	Done
Story	<u>OODSI-20</u>	PoC scenario#2 test case design & execution	Done
Task	<u>OODSI-21</u>	Upgrade to MCS-CC 0.3.2	Done
Story	<u>OODSI-26</u>	Set up GitLab CI/CD at ESOC	Done
Task	<u>OODSI-29</u>	Data exchange between multiple MCS-CC entities using native interfaces	Suspended
Sub-task	<u>OODSI-30</u>	OODSI-1 Install IntelliJ IDEA inside dockerised MCS-CC	Done

# Activity plan and BPF encoding

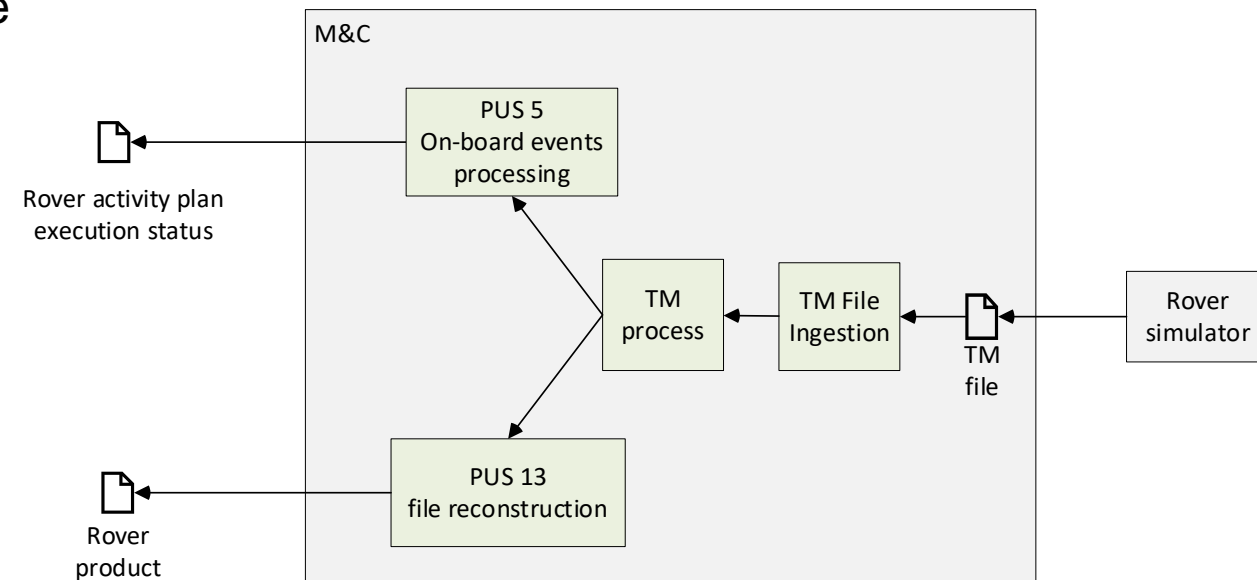
- Activity plan (text) is converted into a binary file which is executed by the rover.
- The plan up uplinked as a file to the rover using PUS Service 13.
- All packets are encoded as CLTUs, which are written to a binary file (BPF) and transferred to the rover.
- In actual operations, BPF uplink/transfer would be via another control centre and spacecraft (e.g. ExoMars TGO).



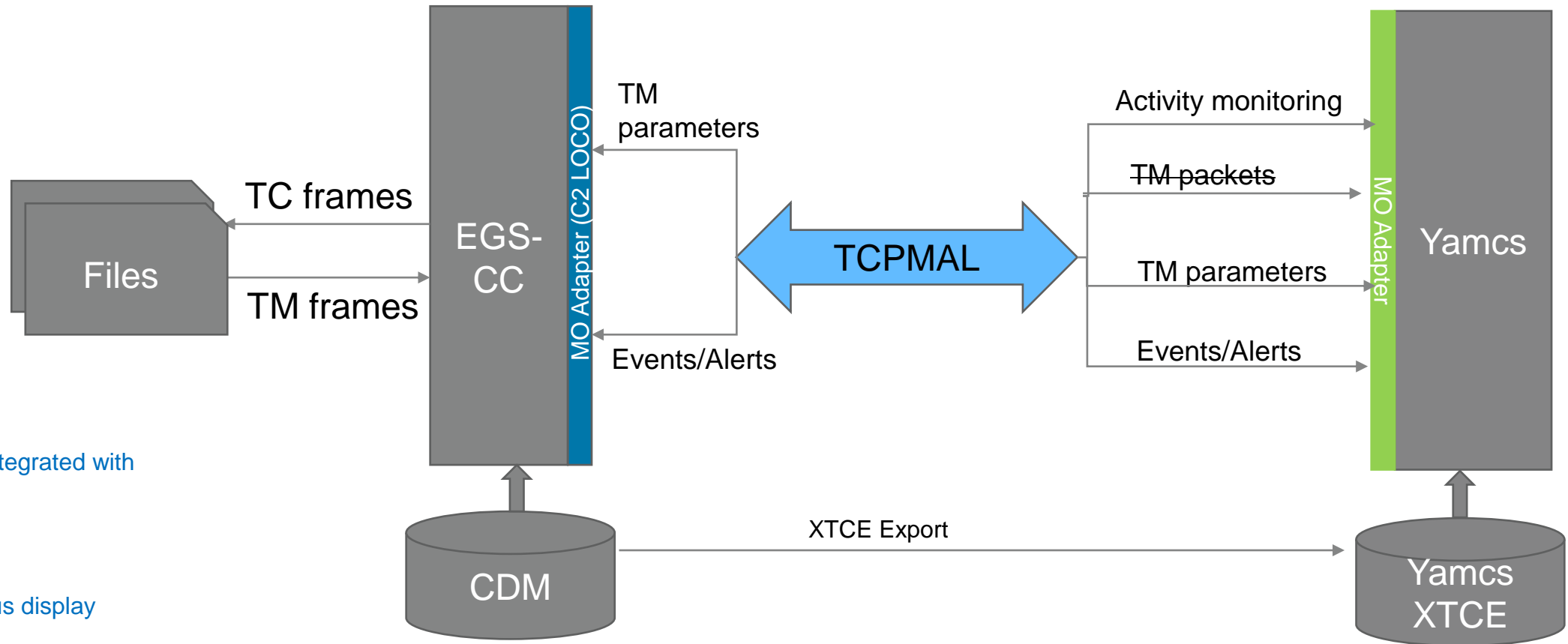
Note: Activities in the activity plan are not the same as activities (e.g. TCs) in the CDM.

# TM file ingestion and processing

- Rover records TM as frames to a file, which is transferred to the rover control centre.
- In actual operations, TM file downlink/transfer would be via another control centre and spacecraft (e.g. ExoMars TGO).
- TM file is ingested, packets and parameters extracted.
- Mission data files (images, etc) encoded in PUS Service 13 packets are reconstructed.
- Event TM provide the status of execution of activity steps.



# Control Centers data exchange overview



- C2LOCO MO adapter integrated with MCS-CC.
- CDM to XTCE export.
- Yamcs MO adapter.
- Yamcs activity plan status display

# Data exchange – using MO services

## Implementation:

- MOA component from C2LOCO integrated into OODSI EGS-CC exmrsp.
- CDM to XTCE convertor, parameter definitions imported into Yamcs.
- Yamcs TM/Parameter/TC/Event MO adapter links created, including Directory and Login.
- Parameter registration and reception.
- Event message reception.
- Activity monitoring.
- Activity (TC) injection not working (MCS-CC exceptions).



## Implementation:

- Using MO monitorValue Pub-Sub service.
- Provides raw value, eng value, status.
- Parameter sample time is not included.
- Need to 'hack' monitorValue service to insert the timestamp into the domain parameter.

## Conclusions:

- Making the MO monitorValue service usable goes against the MO specification.
- The monitorValue service for real-time parameter provision is too basic. Essential information such as parameter generation time is missing.
- The MO adapter monitorValue service can only handle provision of a small number of parameters, due to performance issues.
- MO getValue service provides more information per parameter (generation time could be included), but this is a request-response service (not pub-sub). The client must continuously issue requests.

# MO services assessment – activity monitoring

- Using the MO action service
- C2LOCO maps its internal object ID to MCS-CC object IDs. This means that the client can only monitor activities (e.g. TCs) that it has initiated.
- Information provided by the service is very limited (not sufficient for real use).

# MO services assessment – events

- Events can be used for carrying useful data.
- Example: TC progress can be monitored by monitoring corresponding event messages.
  - PUS 13 file download produce corresponding event message that can be monitored.

# MO services – problems encountered

## Problems encountered:

- No logging – all components throwing exceptions at startup but the exceptions are not logged anywhere.
  - Solution: created a parallel logging functionality creating log files in /tmp/.
- Initialization problem – all components start at the same time and fail because they depend on each other.
  - Solution: added some delays in the initialization to make sure that the components start in the right order
- TCP connections are not well managed – if a MO client disconnects, the MO provider (MOA in EGS-CC) does not seem to notice. This makes it so that if Yamcs closes the connection without logging out first, it is impossible to reconnect (duplicate connection error).
  - No solution for the moment (requires frequent restarts of EGS-CC)
- No commanding: sending a command (activity) fails with NullPointerException.

# MO services - conclusions

- MO monitorParameter service, as defined by the standard, is not useful for real-time asset parameter monitoring.
- C2LOCO activity monitoring service very limited. Only possible to monitor own activities initiated by the client.
- MO API is unclear and difficult to know what to use – not easily usable. It would be easier to use native EGS-CC interfaces.
- Mapping of information / data fields between EGS-CC domain and MO domain causes problems, such as loss of information.
- Operability using MO services is a valid goal, but only works if the actual data exchanged is sufficient.
- MO services are not designed to allow easy extension of the defined services.

# Data exchange – using EGS-CC native APIs

- EGS-CC is designed to support such functionality.

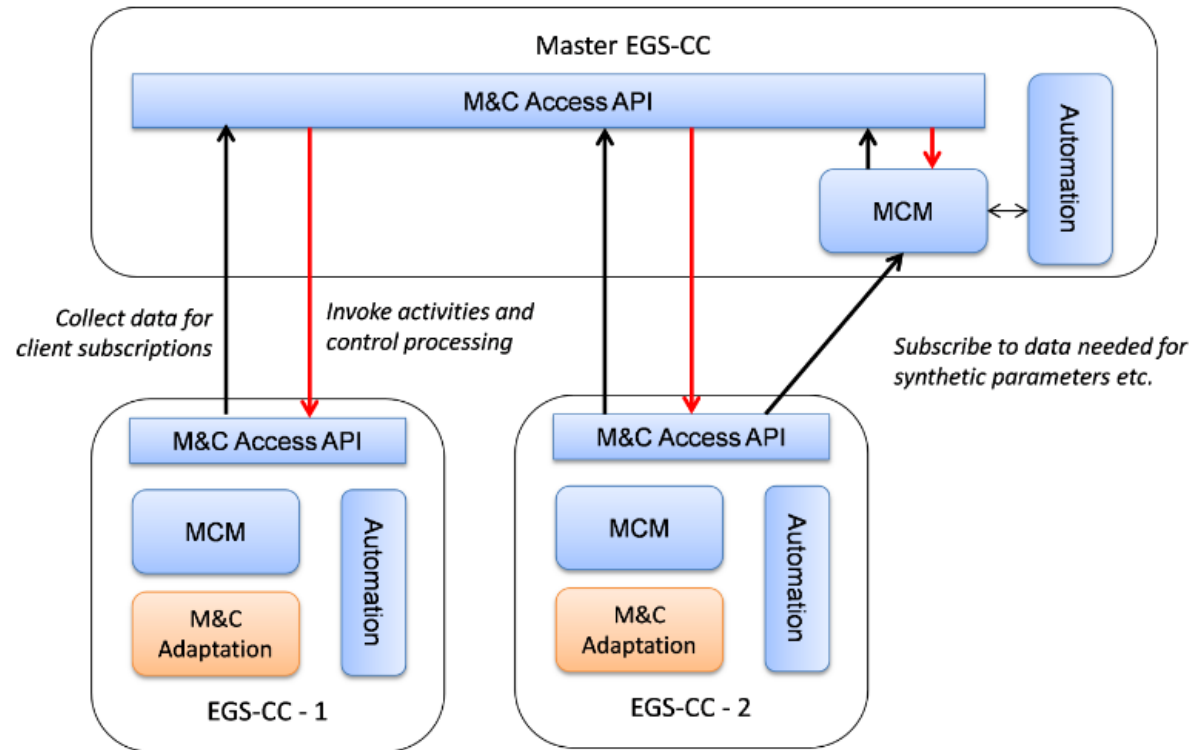


Figure 13 Interaction Between Master and Child Systems

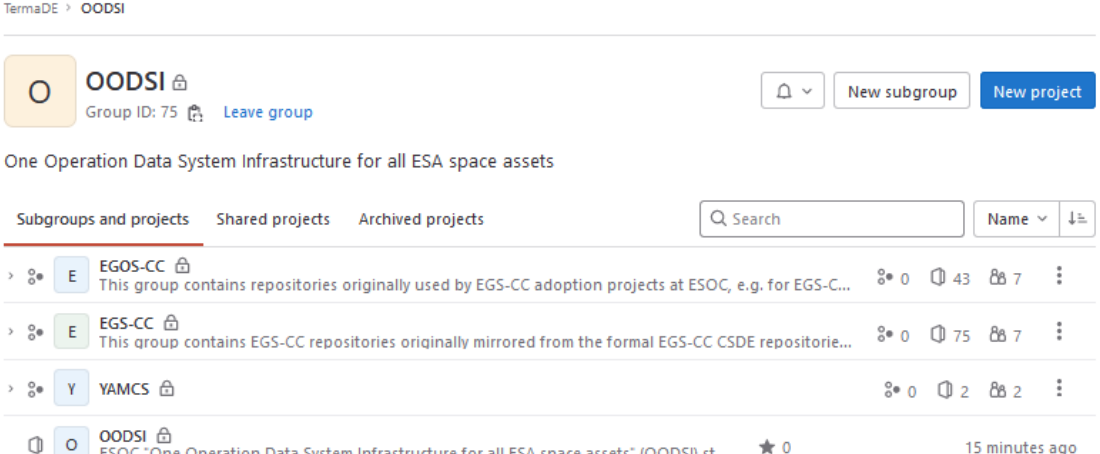
- MCM definitions can be differently distributed (OPEN-M tailoring) across collaborating EGS-CC entities (data access restriction).
- Each EGS-CC entity is meant to be configured (Sysconfig) for remote M&C Access API subscription, MCM parameter retrieval etc.

**But such functionality seems not fully implemented nor validated, including in latest EGS-CC 1.6. TBC. Further implementation and configuration is needed. Implementation does not match EGS-CC documentation.**

<https://csde.esa.int/confluence/display/EGSCCKB/.EGSCC-SYS-SSDD-1000+EGS-CC+v1.0>

# PoC prototype software

- All code now in Terma OODSI Gitlab project, including Yamcs repository.
- Pipeline for building two separate MCS-CC and Yamcs containers.
- Container naming convention of registry:
  - mcs-cc\_\*
  - yamcs\_\*



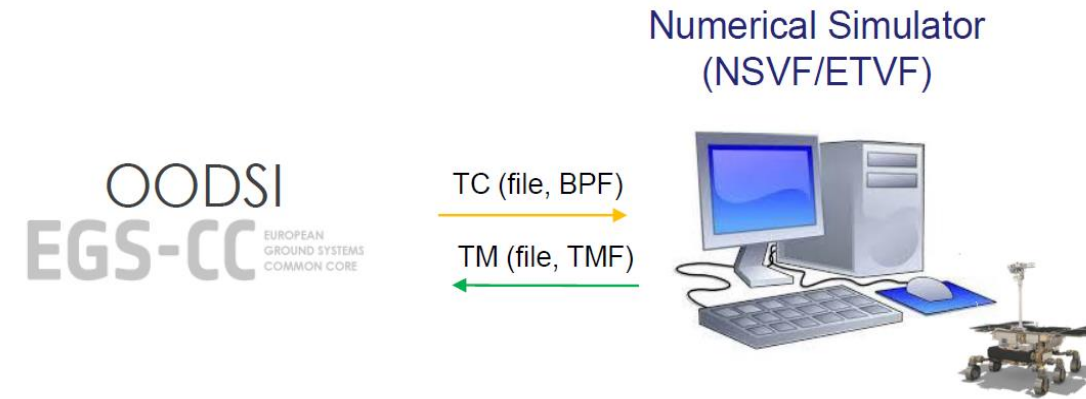
The screenshot shows the GitLab interface for the 'OODSI' group. At the top, it displays 'TermaDE > OODSI'. The group name 'OODSI' is shown with a lock icon and 'Group ID: 75'. There are buttons for 'New subgroup' and 'New project'. Below this, the group description reads 'One Operation Data System Infrastructure for all ESA space assets'. There are tabs for 'Subgroups and projects', 'Shared projects', and 'Archived projects'. A search bar and a 'Name' dropdown are also visible. The main content area lists three subgroups: 'EGOS-CC' (description: 'This group contains repositories originally used by EGS-CC adoption projects at ESOC, e.g. for EGS-C...'), 'EGS-CC' (description: 'This group contains EGS-CC repositories originally mirrored from the formal EGS-CC CSDE repositorie...'), and 'YAMCS'. At the bottom, there is a project entry for 'OODSI' with the description 'ESOC "One Operation Data System Infrastructure for all ESA space assets" (OODSI) st...' and a star icon.

## 6. PoC prototype validation



# PoC validation approach

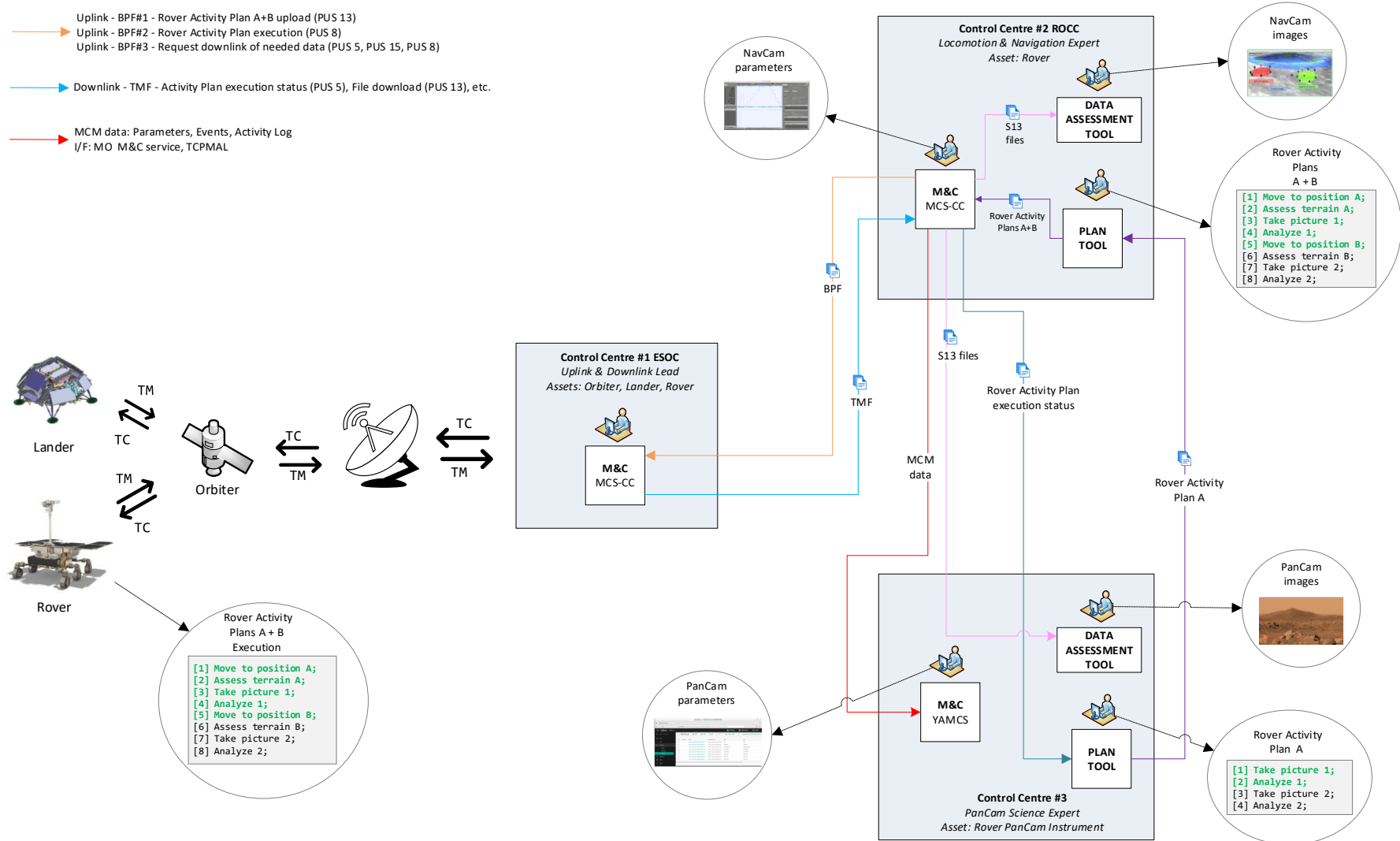
1. Representative rover activity plan provided as input by ROCC.
2. OODSI MCS-CC encodes activity plan file using PUS Service 13 packets, and generates Bit Pattern File (BPF) containing CLTUs for the rover.
3. BPFs are ingested into the Rover Numerical simulator (NSVF), the commands executed and the rover telemetry dumped to a TM file.
4. TMF is received and ingested into the OODSI MCS-CC, packets extracted, processed and mission files reconstructed from PUS Service 13 packets.
5. Mission artefacts (images, etc) are extracted by processing the reconstructed files using ROCC-provided utility.



This approach allows to validate the proper TC generation and TM ingestion/processing in the PoC prototype.

# Validation scenario – real operations

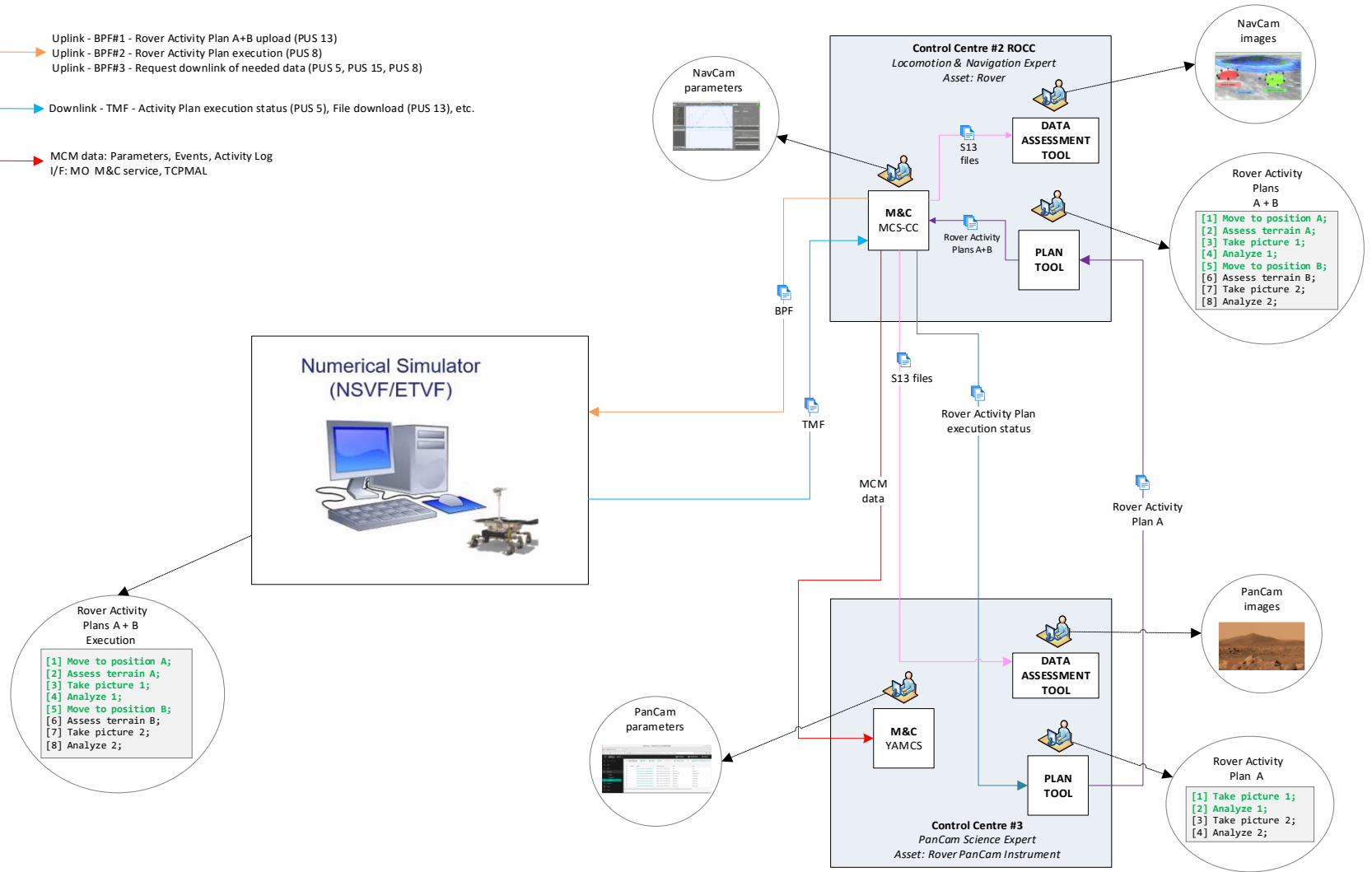
- Uplink - BPF#1 - Rover Activity Plan A+B upload (PUS 13)
- Uplink - BPF#2 - Rover Activity Plan execution (PUS 8)
- Uplink - BPF#3 - Request downlink of needed data (PUS 5, PUS 15, PUS 8)
- Downlink - TMF - Activity Plan execution status (PUS 5), File download (PUS 13), etc.
- MCM data: Parameters, Events, Activity Log  
I/F: MO M&C service, TCPMAL



# Validation scenario – simulated operations

- Uplink - BPF#1 - Rover Activity Plan A+B upload (PUS 13)  
→ Uplink - BPF#2 - Rover Activity Plan execution (PUS 8)  
→ Uplink - BPF#3 - Request downlink of needed data (PUS 5, PUS 15, PUS 8)
- Downlink - TMF - Activity Plan execution status (PUS 5), File download (PUS 13), etc.
- MCM data: Parameters, Events, Activity Log  
→ I/F: MO M&C service, TCPMAL

Details of the validation test steps are in the SVTD and OODSI project wiki on Gitlab.



Live demonstration....

## 8. Results, conclusions and recommendations

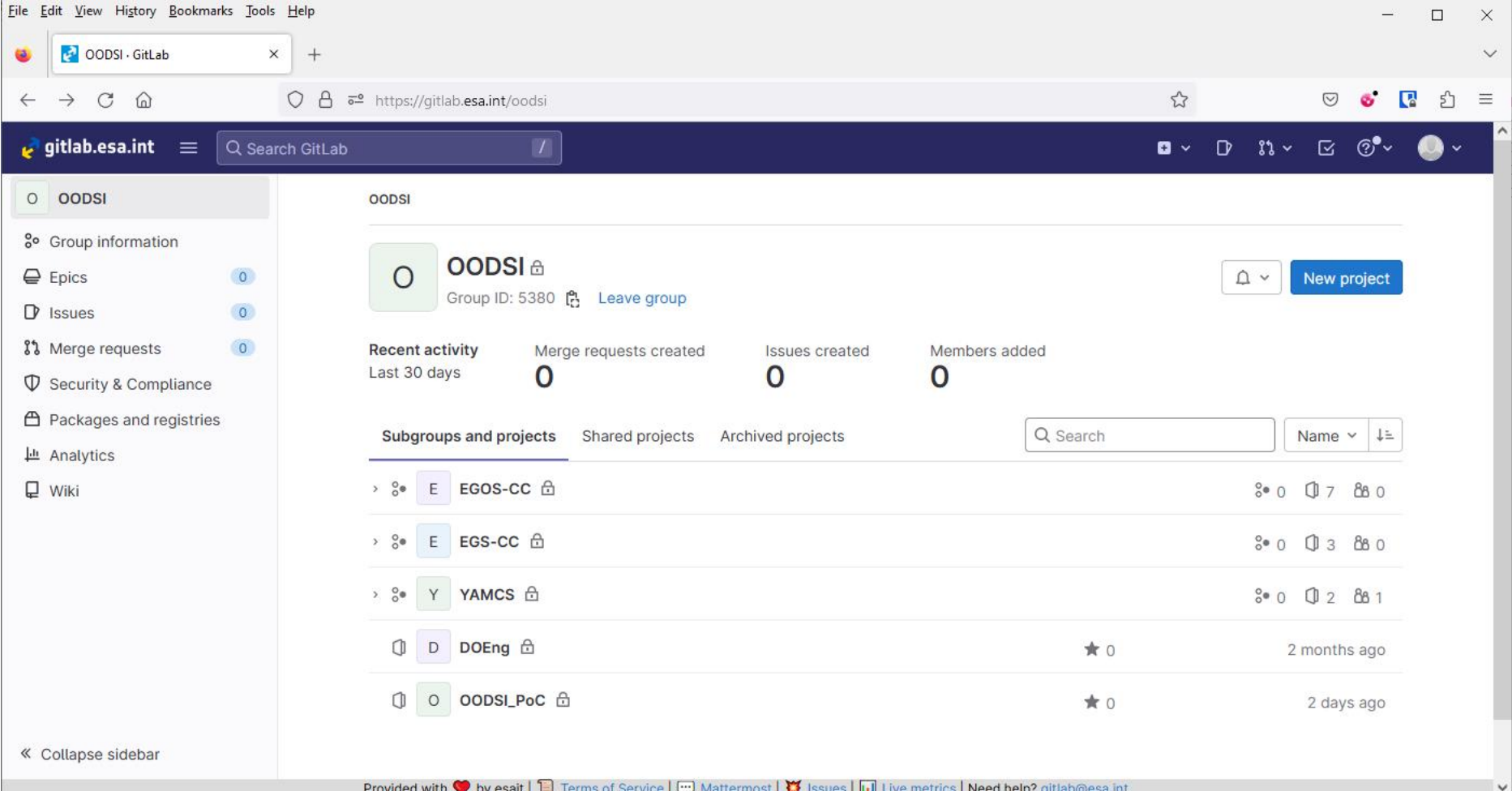
# Study deliverables - documents

Documents:

ID	Title	Reference
D1	Control Systems Scenarios and Functional Breakdown Analysis Technical Note	ESA-DOPS-STU-TN-OODSID1
D2	System Requirements Specification (SRS)	ESA-DOPS-STU-TN-OODSID2
D3	Requirements mapping to ESA infrastructure Technical Note	ESA-DOPS-STU-TN-OODSID3
D4	Architectural Design Report	ESA-DOPS-STU-TN-OODSID4
D5	PoC implementation backlog	<a href="https://sdejira.esa.int/issues/?jql=project%20%3D%20OODSI">https://sdejira.esa.int/issues/?jql=project%20%3D%20OODSI</a>
SUM	User Manual	SA-DOPS-STU-SUM-OODSI
SVTD	SVTD (test procedures)	SA-DOPS-STU-SVTD-OODSI
SRN	Software Release Document	SA-DOPS-STU-SRN-OODSI
AB	Abstract	SA-DOPS-STU-ABS-OODSI
BR	Brochure	SA-DOPS-STU-BR-OODSI
ESR	Executive Summary Report	SA-DOPS-STU-ESR-OODSI
FR	Final Report	SA-DOPS-STU-FR-OODSI

# Study deliverables - software

Software: <https://gitlab.esa.int/oodsi>



The screenshot shows the GitLab web interface for the OODSI group. The browser address bar shows <https://gitlab.esa.int/oodsi>. The page header includes the GitLab logo and a search bar. The left sidebar contains navigation options: OODSI, Group information, Epics (0), Issues (0), Merge requests (0), Security & Compliance, Packages and registries, Analytics, and Wiki. The main content area displays the OODSI group details, including the group name, ID (5380), and a 'Leave group' link. Below this, there are statistics for recent activity: Merge requests created (0), Issues created (0), and Members added (0). A table lists subgroups and projects:

Subgroups and projects	Shared projects	Archived projects	Search	Name	Sort	
> E EGOS-CC				0	7	0
> E EGS-CC				0	3	0
> Y YAMCS				0	2	1
D DOEng			★ 0			2 months ago
O OODSI_PoC			★ 0			2 days ago

At the bottom of the page, there is a footer with links for Terms of Service, Mattermost, Issues, Live metrics, and a contact email: [gitlab@esa.int](mailto:gitlab@esa.int).

# Results, conclusions and recommendations

- EGS-CC/EGOS-CC has been demonstrated to be capable of supporting basic M&C functions for ExoMars RSP operations, using the rover simulator and the same file-based interfaces as the operational ROCC. The proof-of-concept prototype includes reconstruction of rover-generated mission files from PUS Service 13 TM packets. A detailed procedure is provided for executing the validation scenario.
- Collaboration in multi asset operations involving different control centres is more at the level of planning and mission data assessment, rather than M&C. These functions need to be considered for the OODSI infrastructure, at least in terms of how they are supported.
- Pros and cons of different technologies for data exchanges to support collaboration were explored and assessed. No single solution fits all needs considering the different needs depending on the type of collaboration.
- Generic MOI support for collaboration between control centres could be provided by templates and working examples for data exchange, rather than generic applications.
- The currently-defined MO services do not appear to provide a simple solution for ground-to-ground system interoperability between control centre applications.
- The OODSI-CC prototype is easily accessible as docker containers for further use and evaluation by ESOC.
- The OODSI-CC prototype is based on an early release of MCS-CC. Workarounds were devised for some problems found, while some ideas for the study were abandoned. Only core EGS-CC functionality was used. Upgrade to a newer version of MCS-CC/EGS-CC will likely resolve a number of issues.



**Thank you for your attention**