18th International Conference on Space Operations, Montreal, Canada, 26 - 30 May 2025.

SpaceOps-2025, ID #569

Open-Source Ops Tools and Vertical Integration Throughout the Development Lifecycle

Alejandro Sela[a]*, Louis-Jerome Burtz[b], Sam Richards[c]

[a] *JAOPS, Tokyo, Shinjuku, Japan*, asela@jaops.com
[b] *JAOPS, Tokyo, Shinjuku, Japan,* ljburtz@jaops.com
[c] Meridian Space Command, Space City, Leicester, United Kingdom, sam@meridianspacecommand.com
* Corresponding Author

Abstract

The rapid growth of NewSpace companies has placed significant pressure on both the labor and software markets related to operations. Many companies are rushing to launch their products under tight timelines while attempting to systematically reduce costs. Traditional operational tools, however, tend to be either too expensive or too complex to deploy within such short timeframes. As a result, many NewSpace companies are resorting to in-extremis development, often repurposing their engineers into operators.

Open-source tools such as Yamcs [1], NOS3 [2], and Grafana [3], along with free tools such as Rabbit (JAXA) [4] and Omniverse (NVIDIA), are increasingly being used to support operations and operational preparations. JAOPS views this as a natural market evolution, with the core operational tools soon to become commoditized. Standardizing the interfaces will help reduce costs and allow Operations (OPS) and Ground (GND) teams to focus on the added value specific to their mission, rather than on the infrastructure and supporting tools.

Furthermore, maintaining configuration control and traceability across the various phases of development has become critical for quality assurance and efficiency. Currently, many teams develop bespoke scripts or customizations to interface with the hardware, and yet another set of tools is used during testing and integration phases. Despite the differences in these tools, they all share similar requirements: they need to monitor, control, and archive data, and follow both manual and automated procedures. For this reason, JAOPS advocates for the use of operations tools from the early stages of development. The teams at JAOPS and Meridian Space Command are collaborating with Yamcs Gateway to provide low-level interfaces to hardware, enabling the use of the operational Mission Control System, Yamcs, throughout all development lifecycle phases. The benefits of this approach include unified databases from design through testing and operations, improved communication between stakeholders, reduced complexity during phase transitions, and enhanced traceability of testing.

From the perspective of planning and navigation processes, JAOPS is testing, evaluating, and developing modules in collaboration with both commercial and public partners. These modules start from the Yamcs timeline to ingest necessary information and deploy automation features for both monitoring and control as done in the past with other tools [5].

This paper will explore and analyze the operational concepts required to reduce complexity and costs substantially while ensuring quality and reliability, with the ultimate goal of establishing robust infrastructures for operations that could be utilized for OaaS (Operations as a Service). We will share the development status and availability of JAOPS-selected tools, including the on-orbit services simulator, lunar surface rover simulator, and newly developed Yamcs features, such as Yamcs Gateway.

Keywords: Operations, Ground Segment, Vertical Integration, Open Source, Mission Control, Simulation

Acronyms/Abbreviations

AIT: Assembly, Integration, and Testing
API: Application Programming Interface
AR: Anomaly Report
BOM: Bill of Materials
CANBUS: Controller Area Network Bus
CCSDS: Consultative Committee for Space Data Systems
CDM: Conjunction Data Message
CDR: Critical Design Review
CFDP: CCSDS File Delivery Protocol
CFS: Core Flight System
CMS: Content Management System
COTS: Commercial Off-The-Shelf
CSP: Cubesat Space Protocol
CSV: Comma-Separated Values
DEV: Development
EGSE: Electrical Ground Support Equipment
EM: Engineering Model
F': Flight Software Framework (used in NASA's F prime)
FD: Flight Dynamics
FLATSAT: Flat Satellite (typically used for integration testing of satellite hardware)
GIT: Git (version control system)
GMAT: General Mission Analysis Tool
GND: Ground (usually referring to Ground Segment in space missions)
GUI: Graphical User Interface
HILS: Hardware-in-the-Loop Simulation
ICD: Interface Control Document
ISS: International Space Station
JAOPS: Japan Operations (Mission Control and Operations)
JAXA: Japan Aerospace Exploration Agency
JOIP: Joint Operations Interface Procedure
LEO: Low Earth Orbit
LVDS: Low Voltage Differential Signaling
MBSE: Model-Based Systems Engineering
MCC: Mission Control Center
MCS: Mission Control System
MDB: Mission Database
MIL STD: Military Standard
Milbus: Military Bus (a communication standard)
NASA: National Aeronautics and Space Administration
NCR: Non-Conformance Report
NewSpace: The commercial space sector focusing on emerging and innovative space technologies
NOAA: National Oceanic and Atmospheric Administration
NOS3: NASA Operational Simulator for Small Satellites
NTP: Network Time Protocol
NVIDIA: A company specializing in graphics processing units (GPUs), often used for AI and space simulations
OaaS: Operations as a Service
OBC: On-Board Computer
OD: Orbit Determination
OpenMCT: Open Mission Control Technologies (software for space operations)
OPS: Operations
OREKIT: Open-source library for space flight dynamics and trajectory computations
PDR: Preliminary Design Review
PoC: Proof of Concept
RABBIT: Risk Avoidance assist tool based on debris collision proBaBIliTy

ROS: Robot Operating System (used in robotics and some space systems for control and simulation)
RS422: Recommended Standard 422 (serial communication standard)
SCIP: Spacecraft Control Interface Protocol
SIL: Software-in-the-Loop
SLA: Service Level Agreement
SLE: Space Link Extension
SMAD: Space Mission Analysis and Design
SpaceWire: High-speed communication network for space missions
SSA: Space Situational Awareness
SW: Software
TCP: Transmission Control Protocol
TRR: Test Readiness Review
TVAC: Thermal Vacuum Test
UDP: User Datagram Protocol
VOIP: Voice Over Internet Protocol
XML: Extensible Markup Language
XTCE: XML Telemetric and Command Exchange

1. Description of the Problem

1.1. Introduction

As space becomes more accessible, the rise of NewSpace companies has introduced new dynamics, and new pressures, into mission operations. Startups and small satellite providers are often driven by rapid development cycles, limited budgets, and the need to demonstrate capability quickly. While this momentum fuels innovation, it also exposes critical gaps in operational readiness. In many cases, decisions about tooling, staffing, and infrastructure are postponed or improvised late in the development phase or even during the mission. This approach introduces risks that traditional space programs mitigate early on. The following sections outline key operational challenges faced by NewSpace missions, specifically from tooling limitations, and highlight how these issues impact both mission reliability and long-term scalability.

While some companies are starting to develop robust and scalable operational products, the market remains rigid to commercial software implementations. Many NewSpace companies that could benefit from these advancements are often reluctant to allocate the necessary budgets. This leads to a situation where future operators develop their own tools, assuming the process to be linearly incremental; however, in most cases, the effort required to achieve a reliable and quality product proves to be exponential. Current challenges, such as unstable communications or highly demanding payloads, place considerable stress on the ground segment, which risks being inadequate when faced with strict Service Level Agreements (SLAs) or demanding customers.

One issue frequently identified by the JAOPS team is file transfer. For Proof of Concepts (PoCs) or prototypes, self-made solutions can be suitable in terms of complexity and required resources. However, as projects grow in criticality, losing files or needing to re-download them becomes costly for both the Operations and Ground Segment teams. In such situations, there's a tendency to reinvent the wheel by creating applications to address specific needs, often overlooking the availability of established standards such as Consultative Committee for Space Data Systems (CCSDS) File Delivery Protocol (CFDP).

Implementing these standards can be complex for both ground and space segments. Having developed custom-made software for their platforms, many operators face a difficult decision: continue using their current software, into which they have already invested significant effort and which might even have "flight heritage", or transition to Commercial Off-The-Shelf (COTS) or open-source solutions to address evolving requirements.

This scenario often replicates during the early development phases. Testing components, subsystems, and flatsats are frequently done in rudimentary ways. JAOPS conducted dozens of consultations with subsystem developers and spacecraft builders. When asked about the types of EGSE tools they used, the majority reported relying on self-made software and ad hoc scripts. Regarding data acquisition and handling, many still rely on manually managed Coma Separated Values (CSV) files stored in shared folders.

Some companies have recognized this gap and are using open-source products designed to facilitate testing while maintaining traceability and configuration control. Pushing operational tools into earlier development phases can partially address this challenge by providing direct access to visualization and archiving tools. However, interface barriers remain, requiring developers to learn and deploy operational tools that are not always user-friendly.

1.2. Space Operations Processes

As described in the New SMAD [6], operational processes can be divided into several common building blocks: planning, monitoring and control, navigation, analysis, anomaly detection, data processing, and customer delivery.

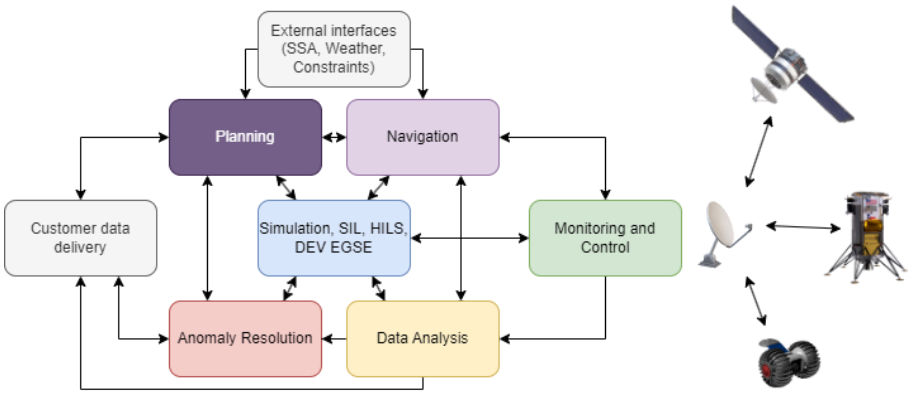

Fig. 1. Real Time Operations Processes

Each operator or mission introduces unique requirements, with customized processes tailored to specific needs. For example, a lunar rover mission requires real-time navigation, imagery, and sensor displays, while an Earth observation mission places more emphasis on scheduling and trend analysis. The process for handling anomalies also varies. In International Space Station (ISS) operations, safety is paramount, prioritizing system stability and deferring troubleshooting tasks to offline engineering teams.

Nevertheless, the core processes are similar, which means that tools, training, and skill development can be built upon shared foundations.

From the technical point of view, space operators must be eager consumers of documentation such as manuals, Interface Control Documents (ICDs), Non-Conformance Reports (NCRs) and any other information that can build their knowledge about the platform or payload that they are going to operate. This means that all this data should be properly handled and always available for OPS teams.

1.3. Space Operations Tools and Associated Problems

Despite these differences, there are strong commonalities, and generic tools can address a wide range of needs. Monitoring and Mission Control Systems that provide real-time data and trend analysis, such as Yamcs and Grafana, could suit virtually any mission.

Efficient scheduling tools are essential. Consolidating all relevant information into a centralized planning platform may seem straightforward but proves challenging in practice. Integrating external interfaces, such as ground station providers, space situational awareness data, space weather forecasts, and shift planning, requires significant effort.

Automation can address many scheduling needs without relying on external AI systems.

Rather than layering AI onto outdated planning tools, rethinking the underlying processes and designing adaptable frameworks can simplify the work of operators, ground controllers, and planners, often resulting in solutions that are more powerful and reliable than current AI-based platforms.

From a data management perspective, it's essential to treat key operational aspects, such as anomalies, configurations, assets, and procedures, as structured data entities. These must be consistently recorded, linked, and maintained across the system to ensure integrity and traceability.

One of the most critical is Non-Conformance Reporting, also known as Anomaly Reporting. For this, it is essential to ensure a seamless link between the data generated during the event, the contextual information, and any documentation relevant to the components involved.

Other structured data elements might include configuration control, asset management, approval cycles, risk management, procedures, and more. Managing these databases centrally, and establishing native, traceable links between them, is crucial to ensuring consistency, efficiency, and operational integrity.

Navigation tools are also mission dependent, for Rover missions or drones for example a map-based display becomes mandatory while for interplanetary missions, Orbital Dynamics plays the most relevant role and for LEO missions Flight Dynamics and Space Situational Awareness are key. Requirements for these tools can become very complex and the ones currently available are in many cases overwhelming for certain missions and require difficult learning curves for engineers and operators. On the Open-source side, the main difficulties are on the interfaces and the lack of easy-to-use GUI. Most of the companies invest significant resources on this either by building a team from scratch with their own software, by implementing expensive COTS, by outsourcing this part to experts in the field or a combination of the above.

Another key topic is simulation. The term "simulation" can be broad and somewhat ambiguous, so it's important to clarify the scope we are referring to. In our case, the focus is on simulation tools used primarily to train the operations team and, in some instances, to test onboard software, processes, and procedures related to operations.

Each mission typically requires a custom-built simulator tailored to its unique characteristics. However, the underlying interfaces and capabilities should remain largely consistent across missions. A robust simulator should be able to integrate with essential tools such as the Mission Control System (MCS), planning tools, and, when necessary, the NCR/Anomaly Reporting tool.

One of the main challenges in this area is the limited availability and the costs of commercial off-the-shelf simulators. As a result, many NewSpace companies are often forced to either develop their own simulators in-house or proceed without one.

Despite these challenges, simulators are a critical component of mission operations. They enable realistic training scenarios both with and without hardware involvement. Simulators based on Software-in-the-Loop (SIL) and Hardware-in-the-Loop (HIL) architectures significantly enhance operational readiness. They allow teams to troubleshoot and replicate both nominal and abnormal conditions throughout the mission lifecycle, ultimately improving mission reliability and performance.

Data access is another critical consideration. Operations teams cannot afford to spend hours collecting, exporting, and distributing brittle CSV files to stakeholders. The data processing levels in space missions refer to the stages of converting raw satellite telemetry or science data into formats that are usable by engineers or scientists. These levels are commonly defined by NASA and other space agencies, and while terminology can vary slightly, the general structure is widely accepted and described on the table below.

Table 1. Data processing levels and users

Level	Description	Users
L0	Raw, unprocessed packets	Ground ops, engineers
L0P	Cleaned-up raw data, error-checked	Ops, telemetry analysis
L1	Calibrated to engineering/physical units	Engineers, scientists
L2	Geophysical/georeferenced variables	Scientists, analysts
L3	Gridded, time-averaged, mapped data	Researchers, institutions
L4	Model-derived or fused data (inferred, simulated)	Policy, applications

Data processing tools would require an entire separate article so we will concentrate on L0 to L1 and the needs and tools required to complete the initial processing for telemetry data and low level payload data that can be directly used by scientists and engineers.

2. Requirements Analysis

At JAOPS, we view the commoditization of operational tools as a natural evolution of the maturing space industry. The standardization of interfaces will significantly reduce costs and empower Operations (OPS) and Ground (GND) teams to focus on mission-specific value-added tasks instead of the infrastructure and supporting tools.

2.1. Starting from the Needs

Any project must begin by understanding the needs of stakeholders across all phases of the mission, from development to decommissioning. This foundational step helps establish clear objectives and align resources effectively. Previous chapters have outlined various problems and the corresponding needs; we summarize them in the following chart for clarity:

Table 2. Main Needs for Operational Tools per User

Users	Main objectives
Dev Engineer	Access to data visualization and compatibility with simulators and low-level protocols (RS422, CAN bus, LVDS, Milbus, etc.)
AIT Engineer	Real-time and trend data visualization, traceability of activities, and telemetry integration within planning tools for testing execution.
Ground Controller	Proper monitoring and control of the different interfaces between tools and deployed hardware such as network devices, servers, computers or firewalls.
Planner	Effective planning tools for current, past, and future activity execution with seamless access to context information and configuration control
Operator	Tools for monitoring, anomaly detection, navigation, and simulation, ensuring critical mission operations are executed seamlessly.
Customer	Access-controlled interfaces that guarantee security while offering trend analysis and insights into mission performance for the acquired telemetry and data.

2.2. Key Requirements Across Tools

A proper definition of the requirements has been completed and is under implementation at JAOPS and Meridian Space Command. Based on the main processes and related tools, we will highlight some of the tools and their related requirements. The next section will go deeper on the proposed solution if already available or will elaborate on how a good coverage of those requirements could and will be completed as part of the development roadmap of JAOPS.

Mission Control System (MCS)
- Visualization of real-time and historical trend data.
- Secure access to mission-specific data levels.
- Compatibility with interfaces including Space-to-Ground protocols and simulators.
- Pre-integration with standard protocols and ground stations
- External connection with visualization and commanding tools

Planning Tool
- Visualize past, present, and future mission events and activities.
- Automate execution of planned activities while ensuring full traceability.
- Enable telemetry visualization and configuration control directly within the schedule.

Structured data entities and Knowledge management tools (NCR, Risks, Tasks, Schedule, Requirements…)
- Traceability of changes and actions.
- Flexibility for deployment of different workflows and templates
- Ready access to historical data for comparison and anomaly investigations.
- Possibility to generate, review and approve content.
- Access control tailored to specific users.

Data Analysis and Processing
- Integration of specialized processing tools to streamline workflow.
- Trend analysis powered by automated anomaly detection mechanisms.

Navigation
- External support for precise trajectory monitoring and risk assessment.
- Able to provide manoeuvring products in short time
- Map visualization for context and situational awareness

Simulation
- Proper representation of space asset up to the defined scope
- Easy to use interface
- Able to replay data and change start time in real time
- Integration with standard OPS tools such as MCS, Planing and Navigation tools
- Ability to simulate anomalies and their resolution

3. Solution: Advancing Operational Capabilities with JAOPS Solutions

JAOPS and Meridian Space Command team members have been advancing processes and tools for over 20 years, continuously integrating new technologies to transform the space community. To address the challenges outlined, JAOPS proposes a versatile toolkit built on open-source and widely adopted solutions.

3.1. Mission Control System (MCS)

JAOPS leverages Yamcs, an open-source tool developed by Space Applications Services [1], supported by a growing community including ESA and NASA and with flight heritage on diverse missions from the ISS to Earth observation to lunar landers and rovers. Through the Yamcs API, seamless interfaces are built to enhance operational efficiency. Visualization tools such as Grafana integrate effortlessly to provide developers with real-time monitoring capabilities, ensuring accuracy during testing campaigns.

3.1.1. Visualization and Compatibility

Yamcs provides its own display tool called Yamcs Studio, and it can also integrate seamlessly with other visualization platforms including OpenMCT, Grafana, and Spacefarer. Its versatile API is user-friendly and meets the majority of operational needs. Grafana, in particular, has been widely adopted by software development engineers for real-time monitoring and trend analysis during development and testing campaigns, complementing Yamcs effectively.

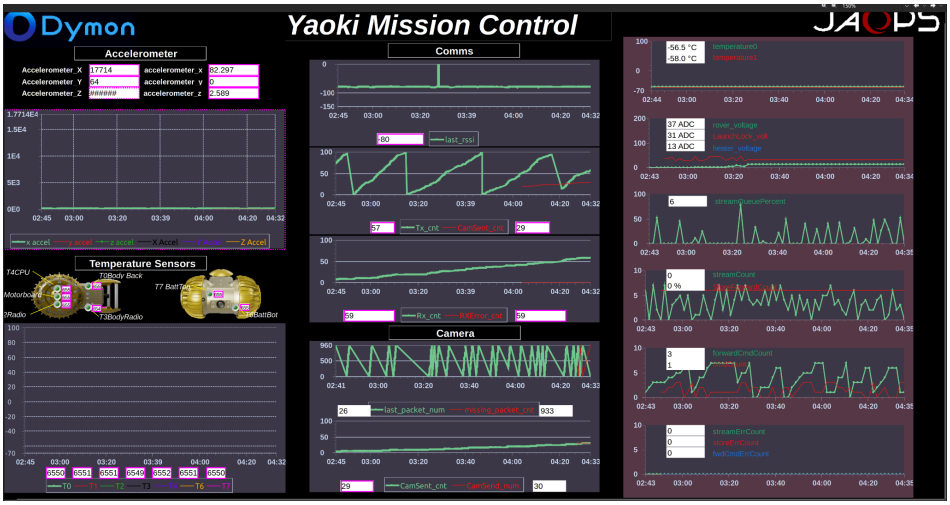

Fig. 2. Example UI: Dymon Yaoki lunar rover display generated with Yamcs Studio

3.1.2. Standards integration

One of the most complex and time-consuming activities from the ground segment perspective is the implementation of standards. To simplify this process, the Mission Control System (MCS) must be compatible with both the Ground-to-Ground protocol (if any), which governs communication between ground systems, and the Space-to-Ground protocol, which defines the communication with the spacecraft.

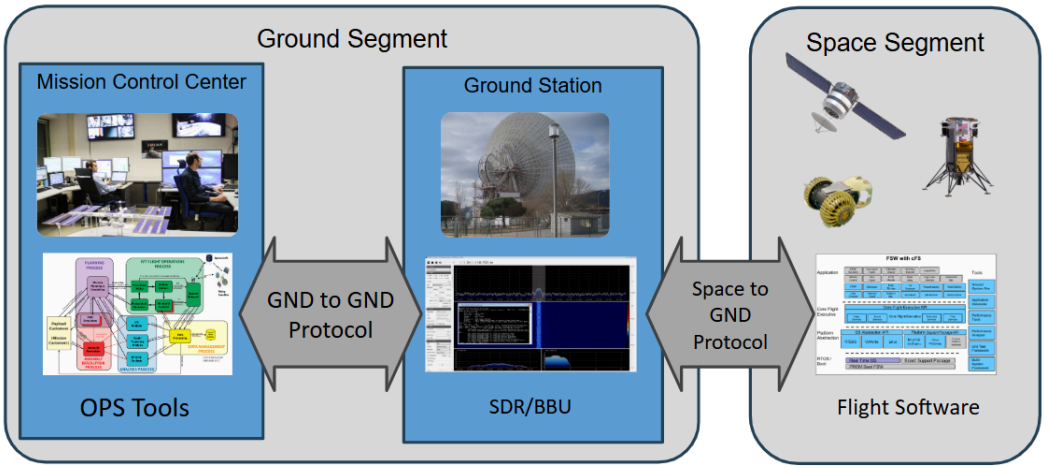

Fig. 3. Overall mission architecture schematic illustrating the protocols between segments

In addition to these two protocols, it's also important best practice to develop the Mission Database (MDB) using established standards to ensure interoperability and maintainability.

Typically, the following are among the most widely used standards in both new and legacy space missions. For the sake of simplicity, we are not separating them by OSI layers:

- Ground-to-Ground Protocol:
  - SLE (Space Link Extension)
- Space-to-Ground Protocols:
  - CCSDS (Telemetry and Telecommand standards)
  - CFDP (CCSDS File Delivery Protocol)
  - CSP (CubeSat Space Protocol)
- Mission Database Standard:
  - XTCE (XML Telemetric and Command Exchange)

All these standards are supported in a standard Yamcs deployment, which is one of the key reasons why Yamcs has grown rapidly in recent years. Operators can quickly deploy ground segments that are already compatible with space assets built using these protocols and standards.

At JAOPS, we strongly advocate for the standardization of interfaces, and for good reason. In our experience, custom or self-made protocols often turn into black holes for resources. What may seem reliable and easy to use for a small CubeSat frequently proves inadequate, unreliable, or incomplete when adapting it to another cubesat or scaling up to more complex missions.

Our recommendation at JAOPS is to always deploy the space segment using open-source, well-tested flight software frameworks such as NASA's F´ (F Prime) or Core Flight System (CFS). Additionally, we encourage selecting generic, standardized protocols to:

- Facilitate interoperability
- Reduce development and operational risk
- Enable mission scalability

By doing so, companies can leverage over 60 years of accumulated lessons learned from some of the best space engineers in the world.

3.1.3. DYMON Yaoki lunar rover example case, deploying an MCS in record time

For most space missions, ground segment development, deployment, and testing projects are typically completed over several months or years. However, JAOPS successfully deployed a Dymons MCS in just weeks, including design, testing on EM, flatsat, and simulations. This remarkable achievement was made possible not only by the previous experience of our team [11] but also by the flexibility and ease of deploying Yamcs and its seamless integration with current space-to-ground standards. The comprehensive documentation and examples provided ensured that the deployment was efficient and effective, paving the way for this success (See Figure 2, showing data from the rover on the lunar surface, acquired via Yamcs and displayed with Yamcs Studio).

3.2. Engineering Ground Support Equipment

Throughout the lifecycle of space assets, one or more Engineering Ground Support Equipment (EGSE) units are typically required for development and testing. These systems are often complex and rigid, frequently housed in portable or fixed racks. They provide a range of interfaces, including thermal sensor lines, RS-422, CAN bus, and MIL-STD-1553 (Milbus), to interact with various subsystems.

EGSE development and maintenance demand a significant investment of time and resources and usually evolve in parallel with the space asset itself. However, with the rise of NewSpace and its focus on speed and cost efficiency, dedicated EGSE is sometimes bypassed. In such cases, teams may opt to use the onboard computer (OBC), either in engineering model (EM) or flight model (FM) form, for testing purposes.

While this shortcut reduces upfront complexity, it can create bottlenecks, especially in larger projects where multiple subsystems are under development simultaneously. Relying solely on the OBC for integration and testing may limit test coverage, slow progress, and introduce avoidable risks.
That is the reason why software-based alternatives, like the one presented by JAOPS, could become extremely interesting thanks to their versatility and affordability.

3.2.1. Real-Time Data and Low-Level Protocols, Vertical usage of YAMCS

Originally designed for real-time data interfacing, whether Ground-to-Ground (SLE) or Space-to-Ground protocols (CSP, CCSDS), Yamcs offers a highly configurable setup. Some entities use it on custom made Ground Stations [7]. It supports data transmission via UDP/TCP and can accommodate FlatSat connections as needed. Furthermore, Yamcs is a reliable and lightweight tool and meets many of the AIT team's requirements out of the box [8].

During the early phases of missions, many companies allocate software teams and resources to develop interfaces between subsystems and the so-called EGSEs. Often, these EGSE systems are massive racks filled with ad-hoc, custom-made interfaces that require significant maintenance and are typically too costly for NewSpace companies.

To address this, the JAOPS team introduced (or rather revisited) the idea of replacing traditional EGSE systems with software-based solutions, basing the architecture on Yamcs and Yamcs gateway.

This approach offers several advantages. Firstly, ad-hoc developments using Python, Matlab, or C++ are no longer necessary. Users can interface their subsystems directly, either through centralized cloud-based infrastructure or via local laptops or small computers running Yamcs and Yamcs gateway. Secondly, different departments and engineers use the same tools across all mission phases. Developers, AIT, and Operations teams can effortlessly share resources, data, procedures, and displays, fostering seamless communication and collaboration. Everyone will, metaphorically speaking, be on the same wavelength, working more efficiently and effectively.

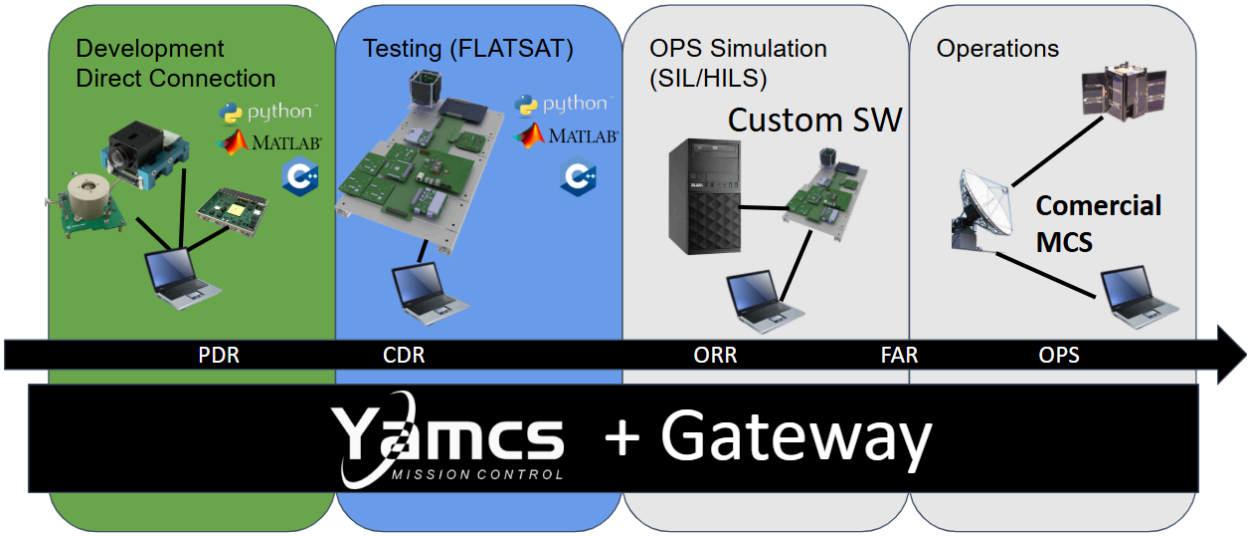

Fig. 3. Integration of Yamcs and Yamcs Gateway with Space Mission development Phases

To reach this situation, a previous limitation, the lack of interface with low-level protocols such as CANBus, RS422, MilBus, and SpaceWire had to be removed. To bridge this gap, the Yamcs community with JAOPS contribution initiated the development of Yamcs Gateway, an Open Source software that serves as the interface between these low-level protocols and Yamcs, effectively transforming it into a software-based EGSE.

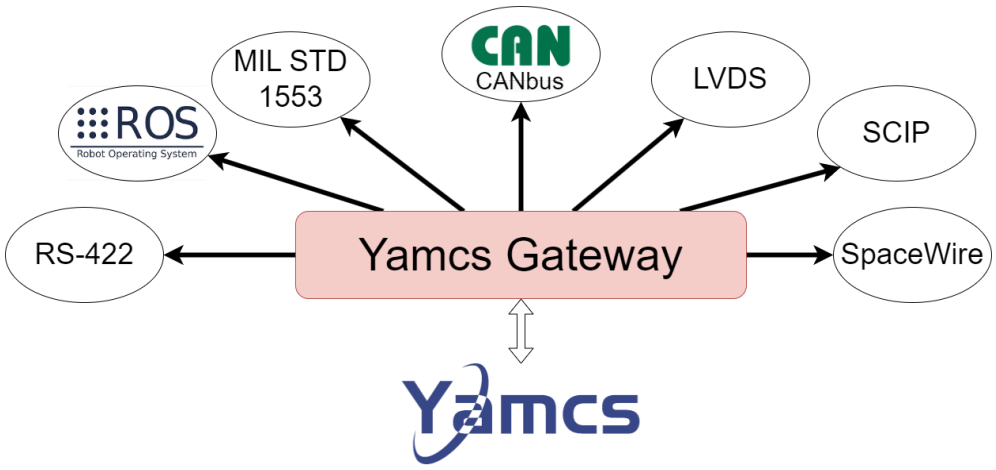

Fig. 4. Current interfaces of Yamcs Gateway

Yamcs Gateway simplifies configuration control and deployment by automatically generating the required Mission Database (MDB). This capability empowers teams to centralize every test using one or more Yamcs servers,

which can operate locally or in the cloud. All data is securely stored, making it accessible to stakeholders via shared tools for monitoring and control. This fosters seamless communication between departments, enabling Dev Engineers to share displays and data with Operators during both troubleshooting and nominal procedure preparation.

3.2.2. Subsystem Developer Support

Subsystem developers can greatly benefit from this approach by leveraging Yamcs and the Yamcs Gateway not only for unit control but also for managing supporting equipment such as power supplies, resistive loads, valves, pumps, and TVac chambers. This setup enables seamless data correlation and access to contextual information across all development phases, from testing to integration.

We envision a software-defined EGSE architecture with standardized interfaces for all support systems, allowing both monitoring and control via Yamcs. This unified control layer simplifies automation, enhances repeatability, and supports detailed analysis.

A potential implementation could involve deploying one Yamcs instance per subsystem or unit under development, enabling direct access to historical data, anomalies, and NCRs related to that specific component. This modular approach improves traceability, strengthens configuration control, and supports system-level integration efforts.

JAOPS provides a service to implement and maintain such infrastructure, tailored to the needs of subsystem developers. For example, a thruster manufacturer might deploy the architecture as illustrated in the following diagram:

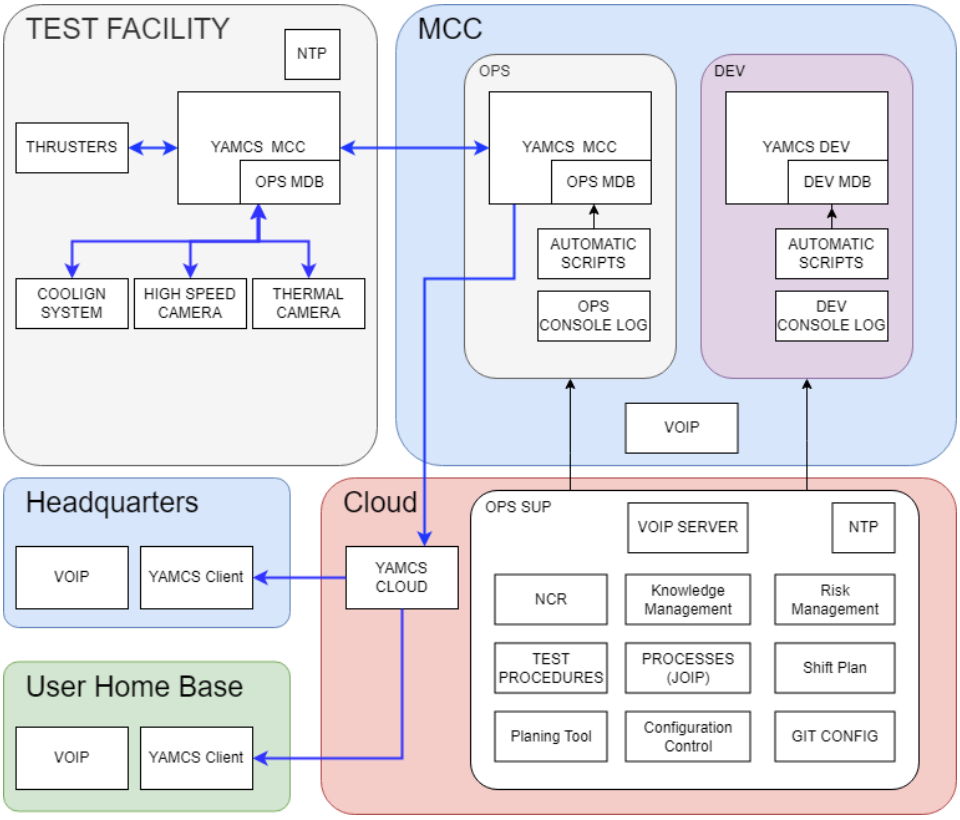

Fig. 5. Example of implementation of Yamcs on a Test Control Center for a thruster company

Furthermore, this approach also offers sales advantages. Companies can provide a lightweight version of Yamcs to customers, delivering a ready-to-use EGSE that simplifies and accelerates acceptance testing for delivered units. For instance, functional tests can be completed within hours of receipt, eliminating the need for flatsat infrastructure

on the customer's side, a resource that is often limited or parsing complicated ICDs to implement ad hoc test scripts. The delivered software could include procedures, displays, and the MDB, further enhancing its value and usability.

Customers would also receive an electronic MDB and a cutting-edge operational tool (Yamcs) for mission use. This system also allows remote support from suppliers to their customers by just connecting the EGSE unit to the internet. Remote troubleshooting and training on the received units could be performed remotely.

3.3. Planning Tools

The planning tool is the cornerstone of any mission. JAOPS advocates for integrating the Past, Present, and Future into the same tool, reducing interfaces and enabling operators to have a complete contextual view of past mission events and future plans. Ideally, all activities with time constraints should be incorporated into the tool.

- Past: Every mission requires a timeline of events "as they happened". This includes knowing who was on console, communications between mission stakeholders, logged information from each position, the status of key telemetry items, and whether activities were executed as planned. This information is invaluable for closing the Plan, Do, Check, Act cycle or addressing troubleshooting efforts.
- Present: Whether a platform operates autonomously or manually, understanding ongoing operations is critical. Monitoring the evolution of each activity and upcoming real-time events plays a key role in mission success.
- Future: Planning processes have evolved significantly in recent years. In the past, schedules heavily depended on personnel staffing with fixed shifts, such as 8/7 or 8/5. However, due to more demanding business models requiring 24/7 operations, planning cycles transitioned to weekly, daily, or monthly reviews. These reviews often involved frozen schedules requiring manual triggers for replanning and conflict resolution. Today, planning demands continuous reevaluation with minimal operator intervention. The advent of optical and inter-satellite communications will remove ground pass constraints, enabling continuous space-to-ground communications with on-demand or 24/7 availability. This will transform planning tools and operational processes entirely, further supported by AI for optimized planning activities.

Unifying all three aspects, Past, Present, and Future, under a single framework is the most effective solution [10]. JAOPS utilizes Yamcs Timeline as a core scheduling framework, leveraging its robust telemetry access and forward-looking planning capabilities to adapt to evolving mission requirements. JAOPS also intends to implement automatic scheduling systems to enable dynamic adjustments based on constraints and resource availability.

One of the planning tool's critical interfaces is its integration with the Mission Control System (MCS). Automating or sending commands directly from the scheduler is a vital task, but implementation can be challenging due to varied codebases and standards among planning tools. JAOPS proposes using Yamcs Timeline as the foundational element for planning tool development. Visualizing telemetry data on the planning tool, such as temperature ranges, mode status, or battery status, could be enhanced with color-coded bands using Yamcs Timeline.

Yamcs Timeline also seamlessly integrates ground station passes and Space Situational Awareness (SSA) information. SSA providers can contribute risk analyses, suggested maneuvers, and Collision Data Messages (CDMs), which operators or automated systems could use to optimize mission phases.

JAOPS is actively developing additional modules to enhance the planning tool's capabilities:

- Automatic Execution: Modules for automated activity execution and direct access to mission procedures.
- Automatic Scheduling: In subsequent phases, JAOPS will develop algorithms for automatic scheduling based on mission needs, resource availability, and operational constraints.

3.4. Structured Data Entities and Knowledge Management System.

Handling knowledge and products among the different teams or simply inside the operations team, is a very complex task that can be improved by deploying centralised inter-connected systems. Reducing the copy-paste tasks and the document or knowledge duplication should be among the top priorities of any Operations team. For that reason JAOPS considers that a tool capable of storing and handling what we call structured data entities and knowledge can become extremely efficient.

In this category we would like to include any item that could be handled with databases, as an example

- Requirements:

For relatively low-complexity missions, a basic system able to trace requirement type, inter-relations, approval, and verification and validation processes becomes very important. This ensures consistency and traceability throughout the lifecycle of the project. For more complex missions, this tool should evolve into a more elaborate system or integrate into a broader Model-Based Systems Engineering (MBSE) approach. Such an evolution allows for tighter integration between systems, better impact analysis, and improved communication across teams.

- Risks:

Risk traceability throughout all development and mission phases becomes crucial for any mission aiming to meet minimum quality standards. A well-integrated system should provide visibility into risks, their status, and associated mitigation actions. Direct traceability between risks, analysis tasks, mitigation activities, and the project schedule significantly enhances transparency. This facilitates proactive management at the team and organizational levels, ultimately supporting smoother day-to-day operations.

- Procedures:

All mission-critical operational and test procedures should be centrally documented, version-controlled, and linked to specific system requirements, tasks, and configurations. These procedures should include step-by-step instructions, pre-conditions, safety notes, and verification steps. Having traceability from procedures to verification results and assets used ensures quality and accountability.

- Tasks:

Tasks should be managed in a centralized system, with links to related requirements, risks, procedures, and assets. Each task should be traceable in terms of its origin (e.g., requirement, issue, risk mitigation), progress status, responsible team/person, and due dates. Integration with scheduling tools and progress dashboards improves efficiency and resource management.

- BOM (Bill of Materials):

The Bill of Materials should include a hierarchical listing of all hardware and software components used in the mission. Each item should be uniquely identifiable, with links to configuration documents, requirements, tasks, and test results. Versioning and change-history of the BOM are essential to ensure consistency and enable audits or reviews. It also becomes extremely important during the mission itself to track anomalies and relation with current or past software/hardware.

- Assets Inventory:

An up-to-date inventory of assets (e.g., hardware units, test benches, tools, ground systems and software) is essential. Each asset should have a unique identifier and metadata including current status, location, calibration/maintenance records, and usage history. Linkage between assets and tasks, procedures, and test campaigns ensures proper resource planning and traceability.

- NCR or AR:

By integrating NCR capabilities directly into the MCS framework, JAOPS eliminates redundant processes, enabling centralized access to necessary data during mission execution or troubleshooting.

On the other hand, access to document-based or knowledge management platforms is highly relevant during both mission preparation and execution. Historically, most knowledge resided in documents produced for key mission milestones such as the Preliminary Design Review (PDR), Critical Design Review (CDR), and Test Readiness Review (TRR).

While some entities have already begun implementing Model-Based Systems Engineering (MBSE), a valuable intermediate step between traditional documentation and full MBSE could be the adoption of a robust content management system (CMS). These tools have evolved significantly from basic wiki-like systems into powerful platforms capable of integrating diagrams, collaborative editing, and linking with other tools.

Modern CMS platforms also support integrated review and approval cycles, with change logs and version histories automatically tracked, making manual version control largely obsolete. A notable open-source tool previously used by the JAOPS team is Redmine. However, its features and user interface are becoming outdated. In contrast, commercial solutions like Atlassian Confluence offer a more user-friendly experience and native integration with Jira, allowing for item-based traceability and increased versatility.

The JAOPS team is currently exploring alternative options, including the possibility of developing an open-source solution tailored to gradually meet the evolving needs of the community.

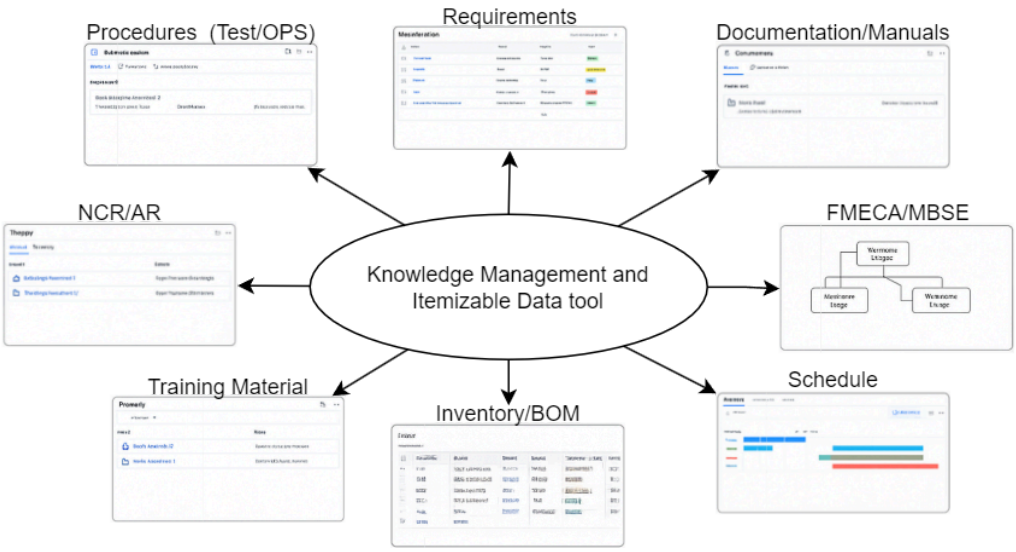

Fig. 6. Integration of different structured data entities and content on a unified knowledge management tool

When focusing on supporting activities like Non-Conformance Reports (NCRs) or Anomaly Reports (ARs), integrating these knowledge tools with the Mission Control System, in this case, Yamcs, can significantly streamline processes. For example, leveraging the Yamcs API to retrieve data from specific timeframes related to anomalies can drastically reduce time-consuming copy-paste operations. This ensures that anomaly investigations are more efficient, more data-rich and less prone to human error.

3.5. Simulation

As discussed in the introduction, the scope of simulation in this document is focused on training and testing of software and operational products. Depending on the mission profile, different tools may be required. JAOPS is currently concentrating on three simulation setups for the following mission types:

- LEO Satellites (Earth Observation, Communications, and other payloads)
- On-Orbit Servicing
- Lunar Rovers

For LEO satellites, Meridian Space Command has developed a tool called BOSSY (Basic Open-source Spacecraft Simulator with Yamcs). This tool features a lightweight simulator that generates dummy telemetry for basic subsystems and supports a limited set of commands. It integrates directly with Yamcs and includes a simple payload capable of capturing images.

Despite its limitations, BOSSY effectively addresses the core requirements for simulation and, when paired with basic training, supports a range of introductory-level activities, such as pass prediction for a target and foundational Flight Dynamics concepts. It serves as a practical entry point for small training campaigns and mission rehearsals.

For LEO satellite missions, JAOPS is also evaluating NASA's NOS3 (NASA Operational Simulator for Small Satellites) as part of its simulation ecosystem. NOS3 is a robust, open-source simulation framework designed to support the development, integration, testing, and operation of small satellite missions. It features an instance of NASA's Core Flight System (CFS) and provides simulated hardware interfaces, flight software, and ground system components, making it ideal for end-to-end mission simulations.

JAOPS leverages NOS3 in combination with Yamcs, running both systems in parallel to create a realistic and flexible simulation environment. Meridian Space Command actively contributes to this integration, enabling seamless command and telemetry exchange between Yamcs and the simulated spacecraft running on NOS3.

As part of this deployment, NOS3 also integrates with "42" software (A Spacecraft Dynamics Simulator) developed by NASA, which provides advanced capabilities for Earth orbit visualization and spacecraft dynamics simulation. This integration allows for the simulation of realistic orbital behavior, as well as detailed sensor models and actuator responses.

Together, this simulation stack supports a wide range of use cases, including:

- Operator training and mission rehearsals
- Software verification and validation
- Hardware-in-the-loop (HIL) and software-in-the-loop (SIL) testing
- Early integration of ground and flight software

This approach is especially valuable for teams transitioning from CubeSat-class missions to more complex LEO satellite operations, offering a scalable and open foundation for mission success.

For On Orbit Servicing and Rover missions, JAOPS is deploying a simulator based on the NVIDIA Omniverse. Drawing from years of experience developing simulation-powered robotic applications using several different simulator frameworks, we selected the NVIDIA Omniverse framework because it provides the best match to the training use cases and realistic visual rendering and accurate physics simulation requirements.

Omniverse delivers scalable, photorealistic and physically accurate virtual environments for building high-fidelity simulations. IsaacSim is an extension of Omniverse specifically designed for robotic applications. IsaacSim is an extensible robotics simulator that provides a faster, better way to design, test and train robots of all kinds.

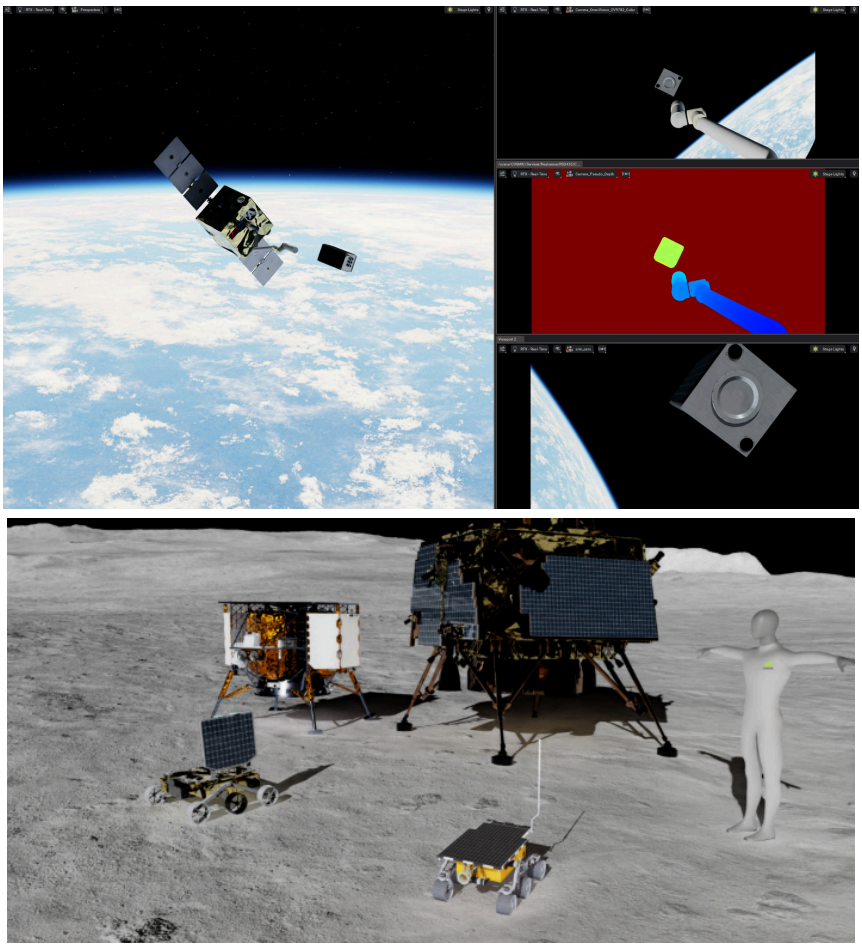

Fig. 7. JAOPS simulator in action with (top) a demonstration of in-orbit servicing and (bottom) several rovers and landers in a high resolution lunar surface environment

To fulfill the operator training requirements, the simulator must run in real time, adjusting to the commands sent by the operators and producing the expected telemetry to display on the mission control center screens. Of prime importance are:

- The realistic visual rendering to provide image sensor data suitable for both human operator displays and downstream image processing algorithms that also participate in the overall situational awareness.
- The accurate physics simulation: for example the contacts between rover wheels and lunar regolith, or the zero gravity environment and joint torques in the on-orbit services scenario
- Thermal, power and radio communication simulation to provide the telemetry the operators need to properly assess the situation and make informed decisions [9].

The simulation environment must be flexible and reconfigurable enough to provide a wide array of training scenarios pre-mission and then adapt to match the effective situational developments during the mission. The figure below shows the JAOPS lunar surface simulation for the Yaoki rover being reconfigured from the expected landing configuration to the actual position (within a crater) and orientation (laying on the side) of the lander on the Moon.

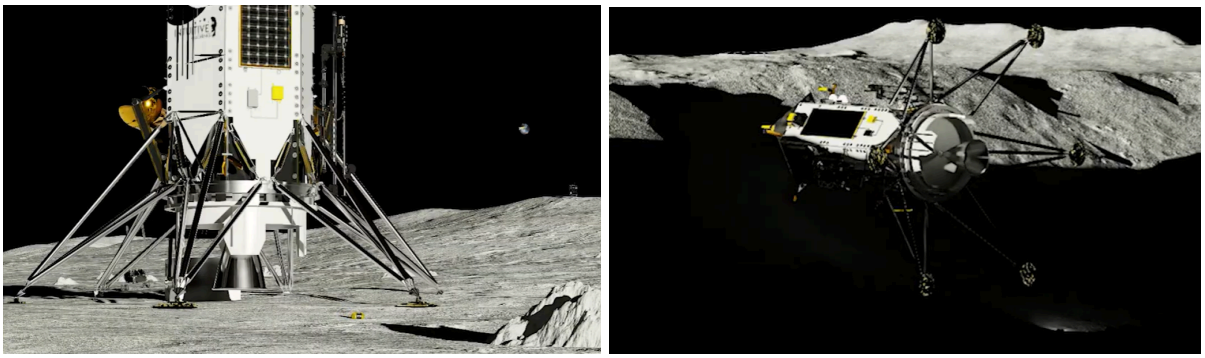

Fig. 8. JAOPS simulator for the Yaoki lunar rover mission onboard the Intuitive Machines IM2 lander showing (left) the expected mission scenario and (right) the actual orientation after landing

3.6. Navigation

Understanding the trajectory of space assets and being able to anticipate key events, such as eclipses, ground station passes, maneuver planning, and conjunction analysis, is critical across all mission types, whether for lunar landers, communication satellites, or Earth observation platforms. These capabilities are essential for ensuring mission safety, maximizing operational efficiency, and planning successful mission timelines.

For this reason, JAOPS is actively evaluating both the existing capabilities and the potential gaps in widely used open-source tools for Flight Dynamics (FD) and Orbital Dynamics (OD). The goal is to identify areas where enhancements or additional modules may be required to meet operational demands, particularly for integration with tools like Yamcs and for supporting scalable, mission-ready Ground Segments.

Among the tools currently under testing and validation are:

- Orekit: A high-precision, Java-based library for orbit propagation, event detection, visibility analysis, attitude dynamics, and maneuver planning. JAOPS is testing its performance and adaptability, especially via its Python bindings, to fit within modern development workflows.
- GMAT (General Mission Analysis Tool): A comprehensive mission design and orbit simulation tool developed by NASA. GMAT is being assessed for its potential use in early-phase mission analysis, as well as for generating maneuver plans that could be integrated into operational mission control flows.

This ongoing evaluation will help JAOPS determine which tools can be reliably adopted for operational and training use, and which may require additional development or customization, either internally or in collaboration with the open-source community.

3.7. Data Analysis

Data analysis continues to be a critical area of ongoing development. Future modules will focus on integrating advanced analytics tools to process mission data, conduct in-depth trend analysis, and implement automated anomaly detection features. Currently, Yamcs offers the capability to bridge L0 to L1 data using custom processors, enabling seamless data flow and initial processing for mission control. Additionally, the exploration of AI-based technologies from various global companies is underway, with the potential to enhance predictive analytics, anomaly detection, and real-time decision-making. However, the availability of high-quality training data remains one of the primary challenges, as obtaining sufficient labeled datasets for training machine learning models is critical to unlocking the full potential of AI-driven solutions in space operations.

As this area evolves, JAOPS will continue to explore new ways to incorporate cutting-edge analytics while addressing data challenges, ensuring that mission data is processed efficiently and reliably to support the growing demands of space missions.

3.8. Current Tool Selection Map

Based on a thorough analysis of existing tools and the identification of operational requirements, JAOPS has crafted the current tool map to address all identified needs effectively. The tool map is designed to provide a structured overview of the tools employed in various categories of operations, ranging from planning and navigation to monitoring and control. While this represents the current state of progress, the map remains a dynamic framework subject to refinement and enhancement. Additional ideas, constructive feedback, and suggestions are welcomed to ensure the continuous improvement and adaptation of JAOPS' operational capabilities.

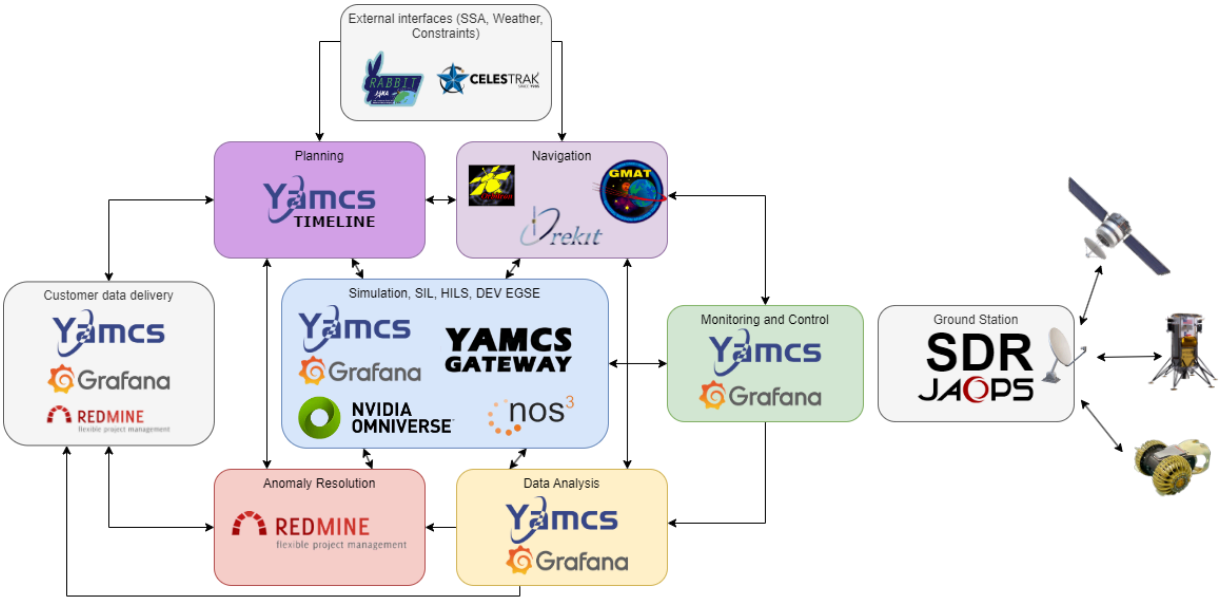

Fig. 8. Proposed JAOPS Tool Map Aligned with Real-Time Operational Processes

4. Conclusion

JAOPS and Meridian Space Command are collaborating with various entities worldwide to deploy affordable and reliable Ground Segment solutions that address early development phases, including AIT (Assembly, Integration, and Test) and development. At the core of this evolution is the open-source tool Yamcs, which is complemented by other tools to meet the diverse needs and requirements of the most demanding customers.

The methodologies and strategy developed by JAOPS are built upon a wealth of experience from different areas of space operations, including the International Space Station (ISS), Earth Observation (EO), lunar landers, rovers,

and even drones. Both JAOPS and Meridian Space Command are fully committed to the open-source community and will continue supporting the development and evolution of both current and future tools.

As described in the paper, not all of the identified requirements are currently covered by available software. However, JAOPS is committed to filling these gaps by developing and integrating new modules, adding to the Yamcs ecosystem, in collaboration and partnership with other companies and universities interested in contributing to this infrastructure. The ultimate goal is to reduce waste and promote standards via an international effort to enhance the operations of any space mission, ensuring better efficiency and sustainability across the space sector.

One thing is clear: the global shortage of tools and talent poses an imminent risk to space sustainability. JAOPS is dedicated to contributing to this cause by shaping a flexible and robust generic Mission Control Center to help address the challenge.

As our motto states: NEVER FLY ALONE

Acknowledgements

We would like to express our sincere gratitude to **Meridian Space Command** for their unwavering support and trust. Their vision and collaboration have been instrumental in shaping and advancing the JAOPS approach.
We also extend heartfelt thanks to all the partners who have contributed to the evolution and growth of JAOPS. Your shared commitment to innovation and operational excellence continues to inspire and drive us forward.
A special thank you to **Space Applications Services**, and in particular to the **Yamcs team**, for their outstanding work, dedication, and the meaningful contributions they have made to the world of mission control systems. Your efforts are helping to redefine the standards of reliability, adaptability, and accessibility in space operations.

Together, we move toward a future where sustainable, collaborative, and open solutions lead the way.

References

[1] A. Sela, "Yamcs - A Lightweight Open-Source Mission Control System", *Conference Paper*, Jun 2012.
[2] D.M. Geletko, M.D. Grubb, J.P. Lucas, J.R. Morris, M. Spolaor, "NASA Operational Simulator for Small Satellites (NOS3): the STF-1 CubeSat case study", *Preprint*, Jan 2019.
[3] J. Bartolomeo, "How Japan's space agency used Grafana to monitor its first moon landing in real time", *2024-08-01*, https://grafana.com/blog/2024/08/01/how-japans-space-agency-used-grafana-to-monitor-its-first-moon-landing-in-real-time/ , (accessed 07 April 2025).
[4] *JAXA RABBIT Project*, "RABBIT: Real-time Application-Based Testbed for Space Exploration", *Japan Aerospace Exploration Agency (JAXA)*, https://rabbit.jaxa.jp/index_en.html, (accessed 07 April 2025).
[5] A. Diaz, J.-M. Wislez, S. Klaï, C. Jacobs, D. Van Hoof, A. Sela, A. Karl, A. Michel, N. This, D. Moreau, "SOLAR Predictor: A Knowledge Management Tool Supporting Long Term Console Operations", *Space Applications Services*, Leuvensesteenweg 325, 1932 Zaventem, Belgium, and B.USOC, Ringlaan 3, 1180 Uccle, Belgium,
[6] W.J. Larson, J.R. Wertz, *Space Mission Analysis and Design*, 4th ed., Springer, New York, 2016.
[7] A. Greenberg, K. Rice, G. LeBrasseur, A. Meneely, A. Tuan, "University Class Open Ground Station (UniClOGS)", *Conference Paper*, Aug 2024.
[8] M. Schmitt, F. Diet, N. Mihalache, "Yamcs for lean Commercial Control Centres: The ICE Cubes Control Centre", *Conference Paper*, May 2018.
[9] L.-J. Burtz, T. Sinsunthorn, A. Sela, "Lunar Surface Visual Rendering, Dynamics, Solar Power and Thermal Simulation for the Operations of Lunar Rover Missions", *75th International Astronautical Congress (IAC)*, Milan, Italy, 14-18 October 2024
[10] S. Klai, A. Michel, D. Moreau, J.-M. Wislez, et al., "SOLAR Predictor: A Knowledge Management Tool Supporting Long Term Console Operations", *Conference Paper*, May 2014.
[11] M. Alzaabi, L.-J. Burtz, S. Amilineni, S. Almesmar, M.S. Khoory, M. Albalooshi, A. Salem, S. Almaeeni, S.G. Els, H. Almarzooqi, L.B.N. Clausen, M. Battler, M. Cross, "Operational Concepts and Rehearsal Results of the First Emirates Lunar Rover: Rashid-1", *Space Science Reviews*, 220(8), November 2024.