Beyond Monolithic Ground Systems: Building Adaptive, Interoperable Ecosystems for Commercial Space

Edoardo Cocci[a], Stewart Hall[a].

[a] *Telespazio Germany GmbH, Europaplatz 5, 64293 Darmstadt, Germany,* edoardo.cocci@telespazio.de, stewart.hall@telespazio.de

Abstract

Modern space missions are becoming increasingly diverse, rapidly evolving, and commercially driven. As a result, mission operators face an operational landscape where **no single ground system can meet all requirements** across the mission lifecycle. Each mission requires a unique combination of capabilities, including command and control, planning, ground station network orchestration, flight dynamics, collision avoidance management, payload data management, and many other specific functions. At the same time, the operational constraints, timelines, and interfaces continue to shift once spacecraft are in orbit.

Traditional monolithic architectures struggle in this environment. They are rigid, slow to adapt, and costly to evolve, often imposing long-term dependencies on a single supplier or a tightly coupled industrial supply chain. Through direct experience supporting multiple commercial missions and building cloud-native mission operations tooling, we observed that operators increasingly need modular, interoperable ecosystems rather than "all-in-one" systems. A best-of-breed ecosystem enables each component to deliver specialised value while relying on open interfaces and integration layers to facilitate seamless end-to-end workflows.

Beyond operational flexibility, ecosystem-based architectures introduce a critical commercial advantage: they give operators the freedom to choose, combine, and replace service providers as mission needs evolve. By avoiding vendor lock-in and enabling competition at the subsystem level, operators can maintain technological agility, ensure long-term competitiveness, and mitigate the risks associated with underperforming or inflexible suppliers. This approach supports a more resilient and dynamic commercial space market, where innovation can occur independently across multiple domains without being constrained by monolithic system boundaries.

This paper examines why ecosystem-based architectures are better suited for the dynamic environment of commercial space operators. We present lessons learned from real customer interactions, highlighting the importance of user-centric workflows, rapid adaptation, and open architectural choices. We discuss how standard interfaces and open integration models are fundamental not only to operational efficiency but also to sustaining a competitive and adaptable commercial ecosystem. We argue that the future of mission operations will be defined by highly integrated yet loosely coupled toolchains, enabling operator-driven collaboration, long-term flexibility, and sustainable growth as space activity continues to accelerate.

Keywords: Ground segment architecture, interoperability, open architecture, commercial ecosystem, vendor independence

Acronyms:

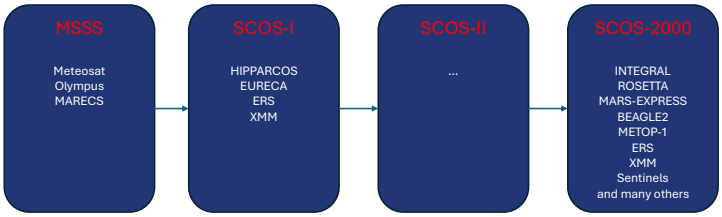
AI	Artificial Intelligence
API	Application Programmable Interfaces
CORBA	Common Object Request Broker Architecture
COTS	Commercial Off-the-Shelf Software
ESA	European Space Agency
HTML5	Hypertext Markup Language Version 5
JSON/YAML	JavaScript Object Notation / Yet Another Markup Language
OEM	Original Equipment Manufacturers
REST	Representational State Transfer
SSA	Space situational Awareness
TRL(9)	Technology Readiness Level (9=flight heritage)
VAR	Value Added Reseller

1. Introduction

Europe has operated its own satellites for decades and in that time, it has evolved strategies for both space and ground segment solutions, evolving from mission dedicated infrastructure to standardisation. However, for reason that will be later explained, there has been a "Cambrian Explosion" of new strategies that have disrupted the path to ever greater standardisation: New Space.

In Europe, standardisation in ground segment is a strategy promoted by the European Space Agency (ESA) with the primary aim of reducing its infrastructure and operations costs. Standardisation has meant that the time and cost to customise a standard infrastructure for a mission is significant reduced. Moreover, the high cost of training operators and their cross-mission mobility has been eased due to the commonality of standard tools and methods. This trend continues today. A side benefit of developing standardised ground infrastructure software has been that European industry, who developed it on behalf of ESA, are granted rights to further develop, sell or operate it for non-ESA missions. Because the software in licenced for zero-cost, the advantage for Industry is that it has significantly reduced the investment to create products suitable for the non-ESA European market. The advantage for ESA has been this commercialisation has supported a strong developer community ensuring high industry availability and competitiveness to deliver ESA's ground segments.

Since the 1980, ESA has developed a succession of monolithic software infrastructure: MSSS, SCOS-I, SCOS-II and SCOS-2000.



Arguably, the most significant event in this history was the development of SCOS-II around 1995 [REF1]. The SCOS-II software infrastructure for Mission Control Systems was intended to provide a complete, sophisticated and low-cost approach for the development of Control Centre systems. It supported a variety of different hardware platforms and architectures ranging from large multi-processor networks for complex or high-performance missions to single workstation configurations suitable for small missions or backup sites. The functionalities offered by the software were carefully tailored to the needs of the operations community and designed to be easily configurable and extensible for mission specific needs. A SCOS-II system could range in size from a single workstation to a network of 40-50 workstations depending on the number of users to be supported and the performance requirements of the mission. SCOSII was not completed or fielded before it was superseded by SCOS-2000. Changes in the marketplace meant that the hardware, operating system or COTS baseline for the system become obsolete. Hardware and software maintenance become prohibitively expensive, so rather than freezing on SCOS-II a migration to SCOS-2000 was undertaken to initiate a migration to a new state-of-the-art ground segment, initially based Sun/Unix and subsequently PC/Linux [REF 2].

SCOS-2000 achieved a high level of stability from Release 3.1 in the early 2000s, and it was at this point that European Industry undertook commercialisation, with several companies developing products or offering solutions based on this and subsequent releases. Products derived from this solution became widely used by European industrial primes and operators and are still widely used today for larger satellites where infrastructure is hosted locally. Today, ESA continues its strategy of renewing its infrastructure software focussed on standardisation and modern software paradigms such as Java, HTML5 user interfaces, automation tools, Space-Link Protocol, Packet Utilisation Protocols and hardware virtualisation, but not cloud computing. This technology is known as European Ground Segment Common Core (EGS-CC) and the product suite known as PULSE. It is expected that PULSE will replace SCOS-2000 at ESA for new missions starting in 2030 [REF 3].

In parallel with these activities there was a major shift in the market with the emergence of new small satellite commercial operators, a phenomenon widely known as New Space. These were new operators not coming from the ESA background and therefore looking at ground segment architecture with new eyes and different challenges to

overcome. The key characteristics of New Space were low budgets, (in principle) low complexity satellites and very diverse mission requirements. New operators found the legacy commercial products available at the time, base don monolithic legacy architectures, to be expensive and cumbersome and poorly adapted to the non-standard protocols used for small satellites. At the same time, cloud computing was becoming the mainstream technology for IT projects in all domains such as manufacturing, engineering and finance, but the legacy architectures were not well adapted to Cloud. New operators with limited capital investment budgets wanted to take advantage of cloud provision instead of investing in hardware infrastructure. As they scaled up, they also wanted to take advantage of the resource elasticity and resilience offered by the cloud computing to enable low-cost scale-up to medium and large constellations.

Another consequence of being new operators was that they were learning as they procured and deployed their architecture. This experience was quite unlike the experience of working with ESA where solutions were specified against large and relatively stable requirement sets derived from a well-used operations concept. Typical new space operators come with few, or no requirements and conversations begin with "I'm launching X satellites on Y date to do Z mission: what do I need?". Initial missions are often demonstrations of technical and commercial ability designed to prove readiness for significant investment into operational systems. Therefore, the ground segment requirements are never stable since they have support constant learning and evolution of operating needs.

The overarching effect was to create demand for a new generation of mission products and services. The classical core of these is: telemetry monitoring, commanding, planning and scheduling and flight dynamics. However, there are additional demands emerging for: advanced space situational awareness (SSA), Artificial intelligence (AI) assistance and automation, ground station services, simulators, and the integration of data processors. This has led to the problem of an expanding toolchain with increasingly complex workflows.

Budgets in new space, and therefore returns on investments, are smaller. No one industrial company can afford the time and cost to invest in a monolithic fully integrated architecture that does everything. General architectures have changed from monolithic software infrastructures of the previous era to favour smaller building blocks that can easily be deployed, integrated and scaled in the cloud.

The building block approach can be viewed as an evolution of functional module architectures such that each block robustly performs a precise task and exposes a simple service via Application Programmable Interfaces (APIs) designed for external interaction. Service APIs enable architects to assemble and deploy easily in the cloud and to add on additional tools, creating new workflows without all the inhibitions of the monolithic architecture. This enabling approach has however presented a new issue of how new space operators should integrate and validate and end-to-end system. Broadly, there are two approaches: to sub-contract a prime integrator, or for the operator to architect, self-integrate and validate the system. In our experience, the chosen approach depends primarily on the availability of software and system engineers in the operator's team, and their desire to "own" the solution rather than depend on a prime integrator.

Classical satellites tend to be very hardware focused due to the complexity of integrating many space hardened hardware components and subsystems. Modern small satellites have simpler pre-integrated hardware architectures where the central processor has more responsibility for managing the components and therefore has a larger onboard software component. Additional, satellite payloads do more onboard processing to achieve faster timeliness. Onboard software can be a key intellectual property for an operator and not subcontracted. This insourcing of software has led to operators having larger software teams that are also capable of integrating ground segment software. Black box monolithic solutions which offer limited or expensive customisations are not so attractive to software skilled teams that can configure, tailor, integrate and validate systems based on building blocks.

Building block solutions offer the advantage of allowing the operator procurement teams the option to source the best value building blocks on the market to suit their needs. However, there is still a trade off between the integration and validation risk that this entails compared with the alternative of pushing this entire risk profile onto a ground segment prime contractor.

2. The Monolithic Model: Technical Rigidity and Commercial Lock-In

By “Monolithic” we mean classical software architecture consisting of functional modules with interfaces that are designed for internal use only, i.e. with prescriptive workflows and internalised data structures. This type of architecture evolved from single process systems. Modularisation eased development by chopping the problem space into small parts that could be developed by a small team or individual and finite time. Modularisation eased deployment by dividing compute demand across multiple processors. The fundamental issue for modern systems is that these internal module interfaces are completely unsuitable for external interactions. Typical issues include lack of documentation, internalised data structures that are difficult to process, bugs arising from undefined workflows, inconvenient functions, protection of intellectual property (code).

From a technical perspective, it was needed to break the tight coupling of the monolith; what do we mean by this? We mean that internalised module interfaces are only designed to work with each other – they are completely coupled and cannot be used without all the complementary modules being present. A common approach to monoliths is the specification of a “middleware” so that all the modules communicate using a common internalised messaging protocol. This has the significant advantage of reducing design and test effort for code dedicated to inter-process and inter-module communication, and it enabled the use of COTS middleware. SCOS-2000 uses the Common Object Request Broker Architecture (CORBA) and COTS for approximately half its internal messaging; CORBA was used to replace much of the previous in-house sockets/packet architecture of SCOS-II. Nevertheless, the CORBA interfaces are entirely for internal consumption; no eternal tools or entities can make CORBA calls into the system.

Being designed entirely for internal consumption with prescriptive workflows makes it virtually impossible to integrate additional modules between core modules or to affect new workflows. It would also require the proprietor of the software to release the definition, documents describing the internal interfaces, under relaxed licencing conditions i.e. an external developer cannot access the internal interfaces, and they risk compromising the stability of the systems if they could. Modules are impossible to replace without being developed by the original developers.

The overall commercial effect is one of creating a long-term dependency for the entire ground segment on a single software supplier or prime. Contrast this inflexibility, supply chain vulnerability and lack of competitive pressure with the low-cost, flexibility and learning that new space operators need and it is easy to understand why new space operators have rejected monolithic ground segment architectures.

3. Mission Reality: Diversity, Evolution, and Specialization

The operational diversity of different new space missions creates a need for architectural modularity that operators can access and utilise to meet their exact needs. In this commercial environment, “best-of-breed” ecosystems for modular components naturally emerge.

In our experience, each mission has specific concerns and needs. For example, in mission control systems, some operators want to have a high level of manual control leading to the need for convenient and ergonomic user interfaces. Others, especially constellation operators, demand highly automated systems and demand carefully considered APIs.

The approach that has been taken with Telespazio Germany’s own product “EASE-Rise” is to focus on the supply of core service for Monitoring and Control, Planning and Scheduling. Our partnership programme takes full advantage of the marketplace for the supply of non-core services: Artificial Intelligence, Flight Dynamics, Cybersecurity, Ground Station Networks, Space Situational Awareness and Simulation. Thus, the cost and time to “reinventing the wheel” and the challenge of customisation for each operator has been completely avoided. New Space operators are free to choose their suppliers from a highly innovative and competitive ecosystem of suppliers, knowing that whatever they buy has been pre-integrated with the core services of EASE-Rise [REF 4] i.e. the integration risk has be pre-retired. Some examples are explained below:

Flight Dynamics is highly dependent on the orbit, orbit precision and propulsion of the satellite and therefore makes use of basic tooling that is heavily customised for the satellite and orbit design; the need for heavy customisation made it unsuitable to be core service. Space situational awareness is a growing concern, and general methods rely on conjunction warnings from free 3rd party sources. While these can be very helpful, they are hard to rely on for manoeuvre decision making. Where space objects are persistently close to satellites, there is an emerging need for close and accurate monitoring to ensure fast and efficient decision i.e. these are also highly dependent on the mission design and therefore not a core service. Mission planning typically mixes and optimises requirements for routine

operation include telemetry tracking and control ground station passes, with user/payload tasking requirements, thus the tailoring is in the "rules" rather than the software itself i.e. is more generalised and potentially a useful core service.

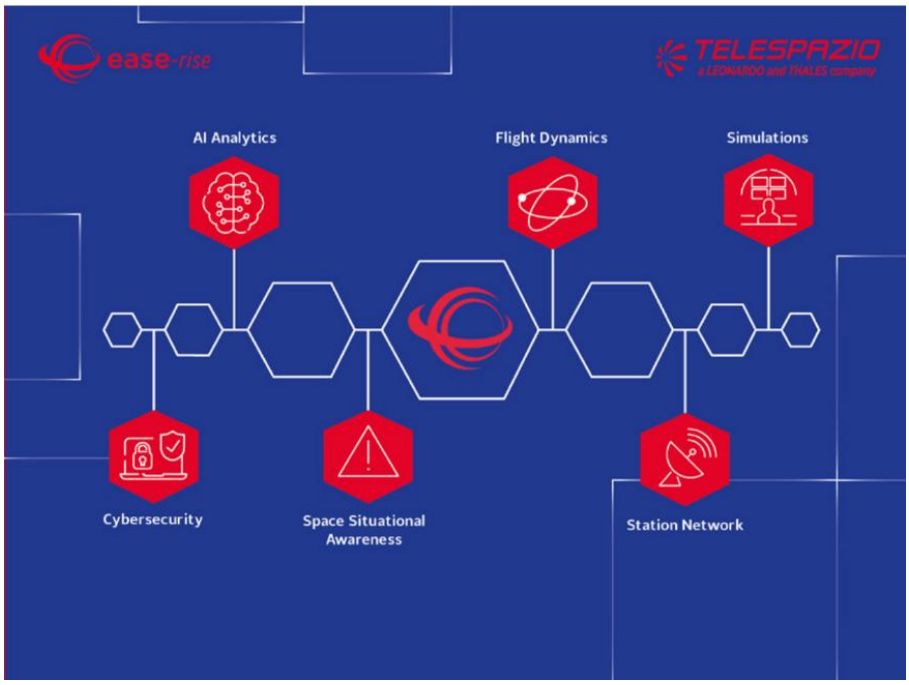
In classical control systems, the operator would own ground stations and their networks, being responsible for the monitoring, control and maintenance of the equipment and sites of these expensive assets. A key challenge is always to have sufficient utilisation from one or more missions to amortise the cost of the investment in ground stations. Modern systems largely outsource ground station services to specialist companies who either own these ground station networks or rent time from other underutilised networks. In both cases, the use of grounds stations becomes much simplified for the operator: using the mission planner to request bookings which are scheduled by the service provider and making telemetry and tele-commanding connections and data flows at the right time. Although using different service definitions, there is sufficient similarity that modularising compatibility is a simple and cost-effective add-on service.

Another emerging area is the use of Artificial Intelligence (AI) assistants and automation in the monitoring of the satellite and ground segment. 24/7 automatic monitoring using AI platforms enables the automatic identification of conditions leading to anomalies and the automatic characterisation and root causes of anomalies when they happen. It can also automate the process of taking a satellite out-of-service, restoring it to a stable and operable state, and returning-it-service efficiently. Not least, AI platform offers insights to operators for their diagnosis. The use and concept-of-use for AI is highly dependent on the operators' needs and preferences and therefore an add-on service.

Some consideration is given to occasional requests for Automation or Orchestration tools. Orchestration means operating ground and space segment in a coordinated and automatic way. In our experience, different operators have different automation concepts. Constellations force operators to use software-based automation while complex missions and the military favour automated manual procedures. Our approach is to assert that built-in tools are not the answer. The real automation asset is a reference service-oriented architecture with automation-ready APIs.

4. Ecosystem Architecture: Technical Foundations

The technical foundations is still a modular architecture but with different goals from a monolith. The decisions about how to organise the modules changes from development and deployment convenience to consideration of usage, always with external users in mind. The primary step is to understand use-cases considering the different architectural boundaries that could be preferred by different operators. In the EASE-Rise architecture we identified the access/authorisation, telemetry metric and telecommand data-model, user interfaces and APIs as an inseparable concern, core services forming a "backbone" up which other add-on services integrate. The key to the re-usability of this backbone has been the definition of a library of protocol adapters that allow it to be easily customised for different satellites while retaining the homogeneity of the data model and user experience.



Mission Planning: As noted in the previous chapter, mission planning and scheduling is a fundament of all systems and sufficiently generic to benefit from in-house development, even though it is an easily separable concern. EASE-Rise is not a core service but an optional add-on service. Nevertheless, EASE-Rise was designed to make the in-house mission planner dispensable and at least one EASE-Rise operator has done so, implementing their own specialist mission planner and integrating it into the backbone monitoring and control services using the same API paradigm. This use-case has proven the benefit of a flexibility service-oriented architecture.

Having defined the architecture, core services and workflows, the next key decision is definition of APIs before any other internal interfaces i.e. "APIs first". Open standards and the willingness to document, publish and train operators to use the service interfaces is key. EASE-Rise uses REST APIs plus JSON/YAML file interfaces and trains operators to integrate their system using Python, aided by a small Software Development Kit that is provided. The combination of the data-model backbone and APIs has resulted in very simple technical integrations with other services, for example, the AI platforms of Intella [REF 5] and AIKO [REF 6] were integrated and validated into the backbone in a few weeks.

5. Ecosystem Architecture: Commercial Foundations

There is a long history of partnerships in IT services, such as referral partners, resellers, Value Added Resellers (VAR) and Original Equipment Manufacturers (OEM). But historically only OEMs or VARs offered ground segment solutions, we believe in-part because of the monolithic nature of legacy products. The flexible service-oriented architecture has opened to industry the possibility of referral parters and resellers and re-shaped the relationship with OEMs. For example, without having to develop their own ground segment, satellite OEMs can buy, integrate and re-sell ground segment systems pre-integrated with their satellites which offers a selling point over OEMs who sell only space hardware and basic tools.

The EASE-Rise commercial model supports parters for re-selling, value added reselling and simply referrals, all of which help to drive business to all the partners in the partnership programme. Ecosystems of partners distribute innovation and risk across industry, thus strengthening long-term market sustainability to benefit operators. Such an approach should be seen as a long-term strategic investment as it simplifies evolution as the mission and market also evolve. For example, EASE-Rise is integrated with five different ground station networks, thus enabling operators to complete and select their ideal choice of ground station services, while relying on EASE-Rise to provide the integration and orchestration out-of-the-box. In another example, EASE-Rise is integrated with two flight dynamics services

which reduces vendor lock in risk for the operator, meaning that they can later choose to replace a potentially under-performing or unsuitable service with the alternative at a late decision point. The existence of multiple parters pre-integrated with the EASE-Rise architecture create a competitive market dynamic and a healthy European space industry.

6. Conclusion

The decades of experience developing legacy ground segment has been successfully leveraged by Telespazio Germany. The key was to let-go of the architectures which have held back adoption of ground segment products by new space operators. EASE-Rise was architecture and developed as a result of listening to the market and re-casting the solution using modern design paradigms and software technologies.

Once the core services stabilised (TRL9) the next step was to fully realise the benefit of the architecture by integrating with parters for the add-on services. The result is the partnership programme; an ever-growing ecosystem of industrial partners who align with the technical modularity of the architecture and help deliver commercial freedom to satellite operators.

7. References

1	SCOS II: ESA's new generation of control systems. Nigel Head, Acta Astronautica Volume 35, Supplement 1, 1995, Pages 515-524
2	Solving the Long-term Support Problem – Migrating Operational Mission Control Systems to SCOS-2000 S.J O'Leary1 , M. Couch1, R. Kay1, AIAA 10.2514/6.2004-511-306
3	**ESA - Pulse: ESA's new approach to flying satellites**
4	**EASE-Rise**
5	**Intella - Transform Sat Operations**
6	**AIKO, infinite ways to autonomy.**