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

SpaceOps-2025, ID # 54

Elevating Operations: Centralized Flight Dynamics Monitoring for EUMETSAT's Growing Fleet

Rami Houdroge[a], Jose Maria de Juana Gamo[b]

[a] *CLC Space GmbH: Kirchstraße 47, 64665 Alsbach-Hähnlein, Germany, rami.houdroge@external.eumetsat.int*
[b] *EUMETSAT: Eumetsat Allee 1, 64295 Darmstadt, Germany, jose.dejuana@eumetsat.int*

Abstract

The EUMETSAT Flight Dynamics Operations team manages multiple spacecraft using a core operational software based on the NAPEOS package, supplemented with mission-specific features. Each software instance is deployed on a dedicated ground segment. This approach has resulted in a proliferation of disparate scripts and processes, leading to maintenance challenges and technical debt.

To address this, a transformative project leveraging Django, Nuxt, Highcharts, Kubernetes, and DevOps was introduced with the aim of centralizing monitoring and enhance efficiency, reliability, and issue resolution across missions. The initiative seeks to consolidate access to operational data, provide real-time monitoring, anomaly detection, trend analysis, and post-processing capabilities.

Keywords: Flight Dynamics, Operations, Monitoring, Automation, Efficiency.

1. Introduction

The EUMETSAT Flight Dynamics Operations team is a multi-mission team operating multiple spacecraft. The core operational software remains, to this date, a customized implementation of the popular NAPEOS package that has, over the years, been complemented with features required by new missions. The ground segments remain very similar in architecture, running largely the same Flight Dynamics software as independent instances. Future missions are thought out in the very same way.

At the time of writing, the operational infrastructure comprises 17 virtual machines running NAPEOS, supports operations for 11 satellites, consists of more than 35000 configuration files and 50000 lines of code in scripts stored in different repositories. These systems produce more than 100000 log files per month corresponding to separate program executions, all of which contain important Flight Dynamics data about the flying spacecraft. Over the years, ad-hoc scripts and processes tailored to a single mission and with a single purpose proliferated, as did the number of related automated emails. The result is many ad-hoc monitoring tools that are difficult if not impossible to maintain, most of which are not suited to support new missions and that come with ever growing technical debt.

With entry into operations of new missions, the number of log files and running processes is foreseen to increase significantly. Quick and reliable access to key Flight Dynamics metrics and troubleshooting data become critical to ensure safe operations.

The dashboard presented here is a transformative project designed resolve those issues by enhancing efficiency, reliability, and issue resolution in operations. It is a monitoring system that centralizes all Flight Dynamics monitoring

aspects for all missions and systems but also builds the long-term archive of Flight Dynamics operations and provides easy access to all relevant operational data.

It provides the team with essential features such as near real time monitoring of operational systems, anomaly detection and notification, trend analysis of key Flight Dynamics metrics, visualisation and post-processing of operational orbital data, ingestion, and storage of manoeuvre information as well as conjunction events. Just as importantly, it includes daily backups to a safe location as well as associated recovery procedures. Crucially, a comprehensive CICD process, much refined over the years, allows us to have a versatile dashboard to support our operational needs and on-board future missions.

The dashboard entered service in October 2020 – date corresponding to the deployment of the operational database – and has since processed more than 2,300,000 logs, 53,000 of which are orbit determination logs.

2. Methodology

2.1 Introduction

The guiding principles of the project are to relay accurate information as quickly and as reliably as possible, eliminate the need for manual tasks that don't have any real added value, and allow the deployment of new releases seamlessly. Unlike typical web applications, almost all user interactions are READ operations.

Over the years, we sought to increase the scope of features offered by the dashboard, aiming at providing better and quicker access to the information typically found in manually edited reports.

2.2 System description

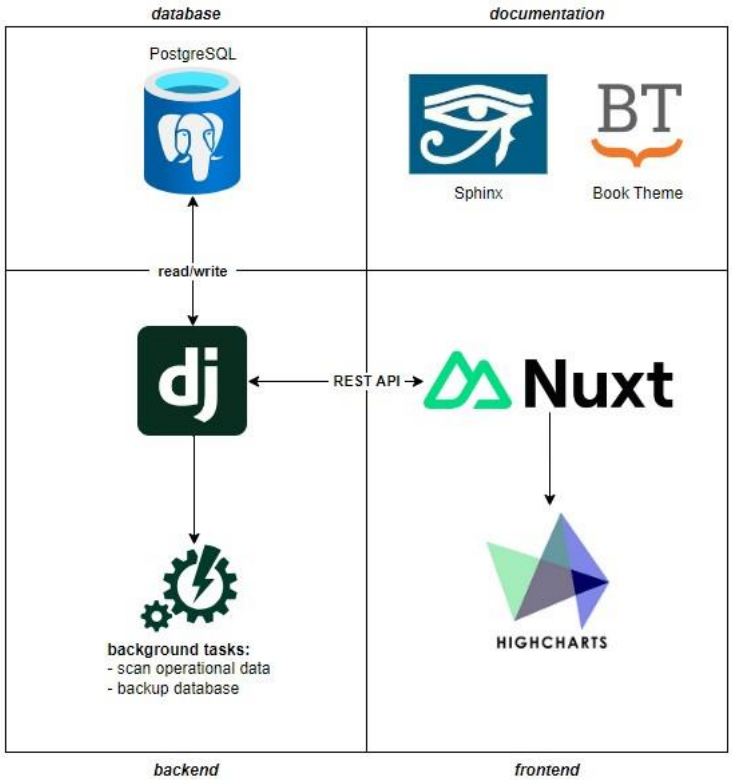

Fig. 1. The complete stack

Since its inception, the project grew significantly in scope. It started out as a single experimental instance of the popular Django server installed on a rogue VM and now comprises a stack of containerized services running on Kubernetes clusters. The backend service is a Django server running on a container image that supports Orekit, Geneos, and hosts an in-house built python library everything else. The frontend service is a Nuxt server running the Vue.js JavaScript framework as well as the outstanding Highcharts charting library.

The project documentation is maintained in a Sphinx site and includes a guide to get started maintaining the project, key elements of code architecture including automatically generated database relationship diagrams, answers to the essential questions a developer would ask, as well as release notes. In addition to that, the backend serves an API documentation thanks to the popular Swagger tools.

2.3 The role of automation

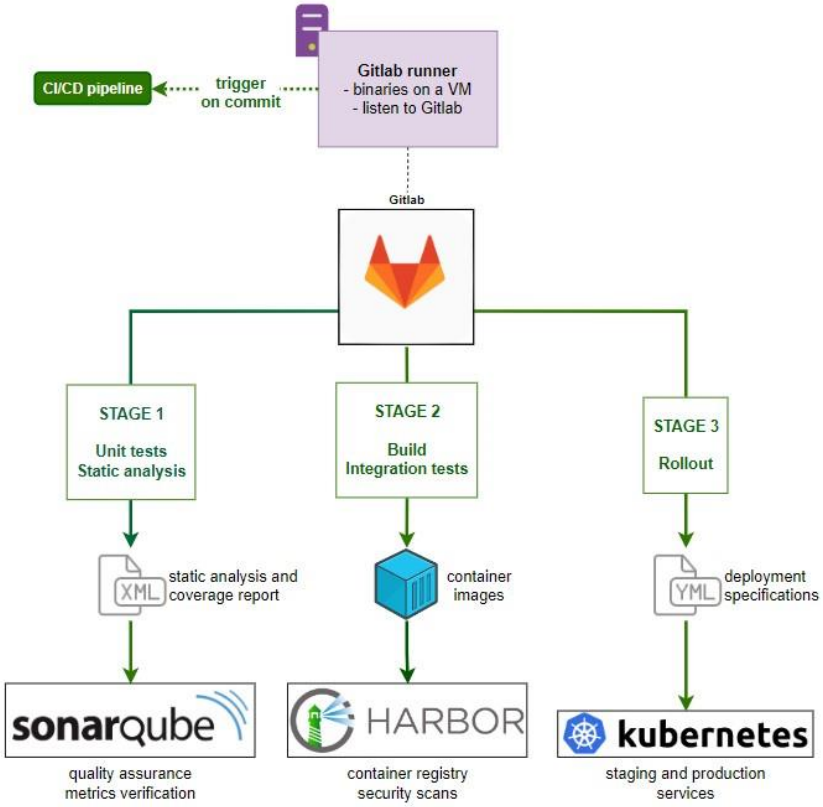

Fig. 2. The role of automation

The project has come to rely on a comprehensive pipeline that leverages the services provided by the EUMETSAT Quality and Infrastructure teams. The pipeline itself applies the general principles of DevOps. It automates verification and integration tests, quality assurance scans as well as build and deployment tasks for two independent services, one for validation and one for operations. The overall strategy ensures that the system is built on solid foundations and reduces the risk of critical issues in production as much as possible.

The first stage runs unit tests automatically, performs a static code analysis and generates code coverage results. This stage uploads the results to the quality assurance server to verify compliance with the organization's standards for software products. The quality assurance stage is blocking, meaning that if the required standards are not met, the pipeline is blocked, and the deployment cannot proceed. This ensures that the quality metrics are met well before deployments are undertaken.

The second stage is triggered when the first stage completes successfully. It oversees building the container images used for deployments, and performs integration tests, making sure that the containerized services start successfully and do not show evidence of errors.

The third stage is triggered afterwards and rolls out the new services to the targeted clusters. This is performed in two steps. The first applies the updated deployment specifications, which contains all the configuration elements required for the service to run (resource allocation, environment variables, secrets, volume mounts, ingress rules, services etc…) and is kept in the repositories. The second step rolls out the new containers (as updated "dev" containers in staging or versioned containers in production).

2.4 A staged approach

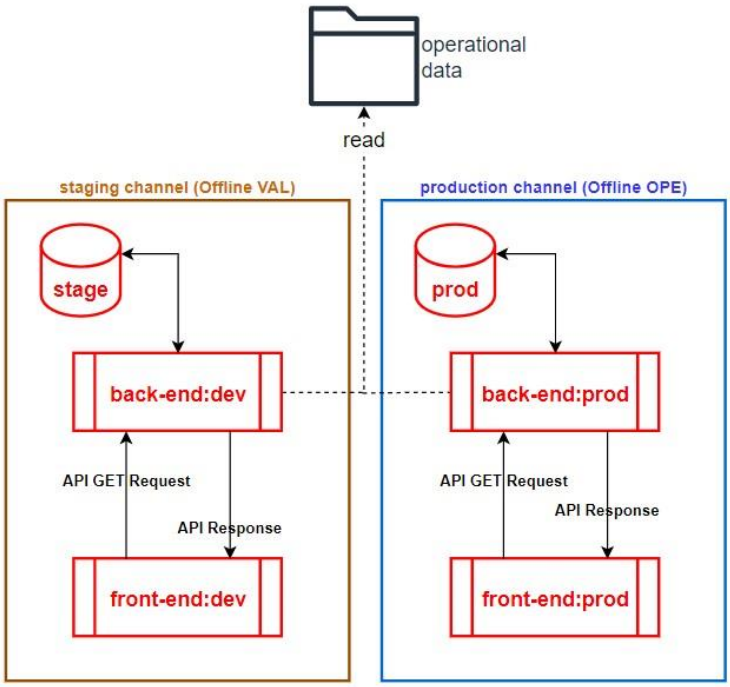

Fig. 3. A staged deployment strategy

The chosen deployment strategy is a standard one relying on a staging as well as a production environment and is also managed by the pipeline. With every push to the master branch of the repository, the staging channel is updated. The deployment to the operational channel is, on the other hand, performed only when the repository is tagged and tagged container images are generated.

The staging channel is identical to the production channel and runs the same tasks as the latter. Its purpose is to verify that the deployment in operational conditions does not result in unexpected regressions and side effects, and that the overall performance of the system is unaffected.

Just like operational flight dynamics systems, parallel operations are performed to ensure that the system runs as expected. Naturally, parallel operations vary in nature. For small changes and bugfixes, a limited amount of time is needed to ensure that no unexpected regressions are introduced. Additions of large new features and reworks of existing features do require, on the other hand, longer parallel operations.

Once the changes are deemed acceptable, the repository is tagged. The pipeline runs same battery of tests and generates the tagged container images that are then rolled out to the production channel.

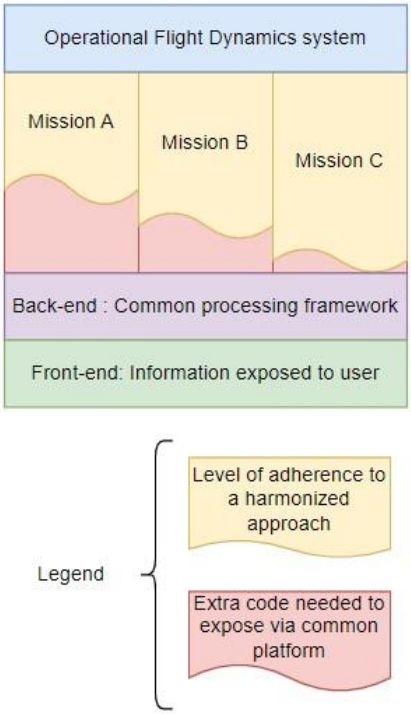

Fig. 4. Thinking the future

The dashboard provides several services that allow the operations team to on-board new missions seamlessly, allowing streamlined operational processes to be followed across missions and teams.

As new missions enter preparatory and development phases, their Flight Dynamics configurations are built adhering to common harmonized concepts (e.g. manoeuvre preparation) and thus leverage the existing services, minimizing the number of new implementations required to effectively operate the new mission.

As new features are thought out and implemented, they contribute to the common processing framework layer and are thus made available to all missions.

3. Features

3.1 Introduction

The main features of the dashboard are built to reduce the amount of time needed to access essential if not critical information about the different missions. Successive refinements have only been possible through an iterative process enhancing the existing features through the filter of user experience and feedback as well as performance observed processing real data. The monitoring system is built around the operational data generated by the operational systems and is by design able to handle that data.

3.2 Monitoring of operational systems

Data produced daily by the operational systems is key to understanding what the status of each system is and what the orbital status of a given satellite is. Log files are processed as they are generated and are made available through the dashboard. The logs are scanned for errors and erroneous logs appear in the dashboard in a simple table that makes accessing the erroneous logs very easy.

Fig. 5. Accessing erroneous logs

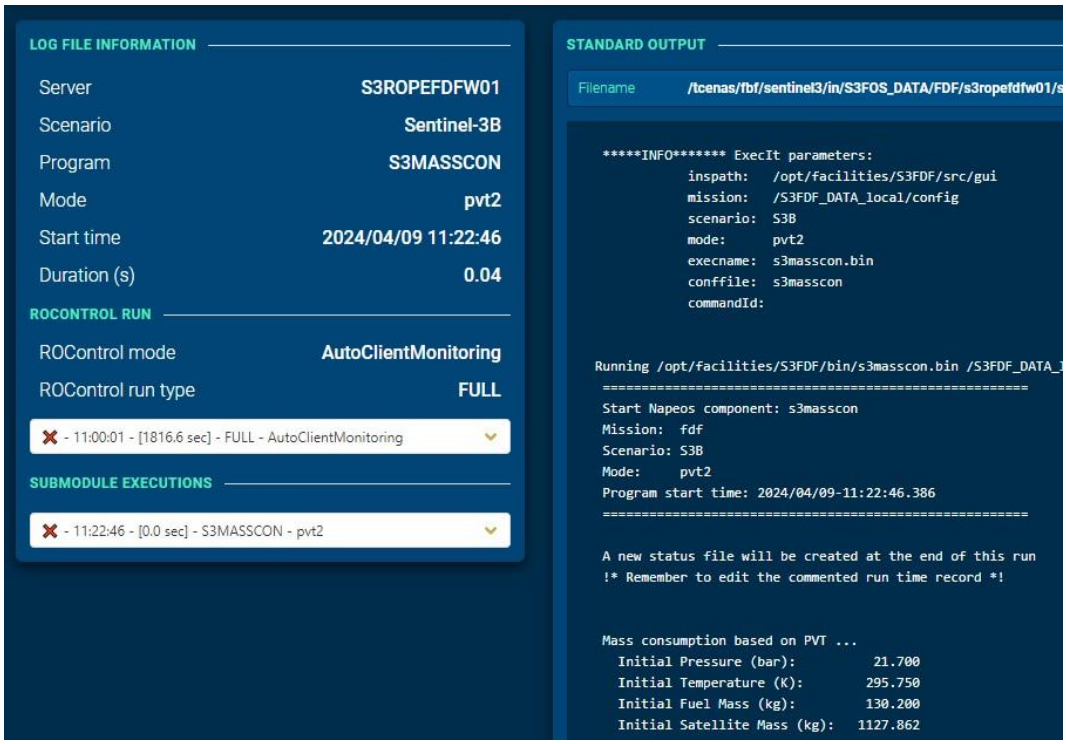

Fig. 6. A log file where an error was detected

3.3 Long term storage of key metrics

In addition to scanning for errors, the monitoring mechanism is augmented by a parameter monitoring framework that extracts metrics from specific log files.

The standard mechanism relies on a manual definition of the log files to be analysed (e.g. generated in a given server for a given satellite) as the associated parsing rule (e.g. line to be matched, field number in line). The extracted metrics are stored in dedicated database tables and made available to the user via a standard interface.

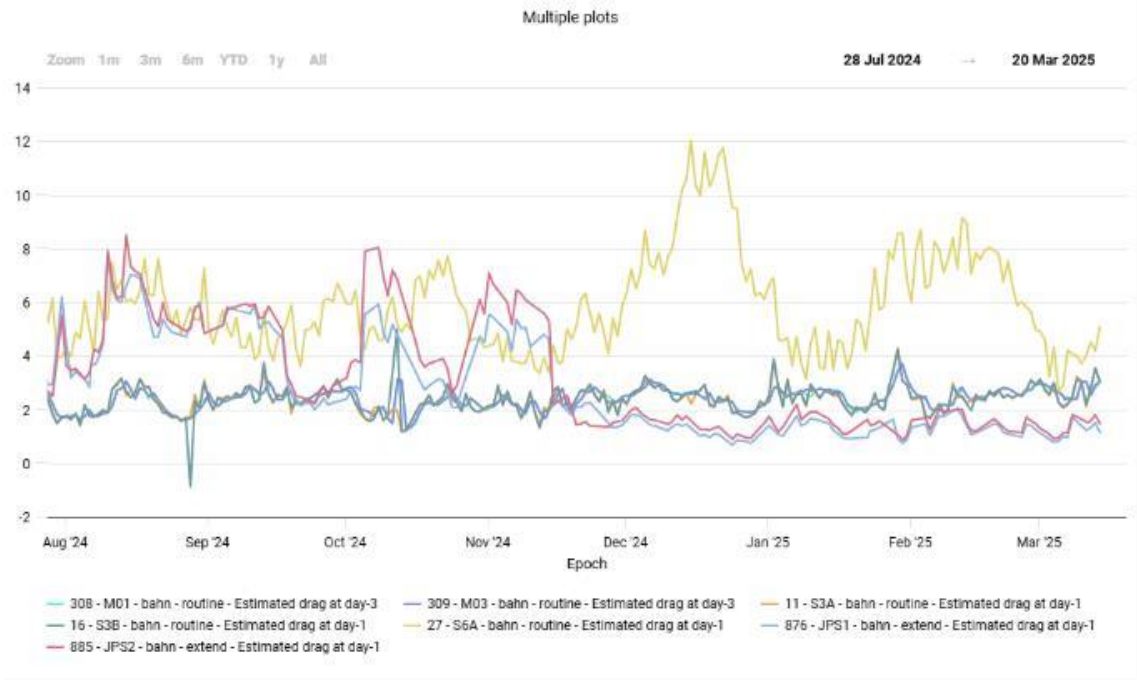

Fig. 7. Estimated drag coefficient in routine OD

A complimentary mechanism is implemented for processing station ranging and Doppler data. As the number of parameters of interest is large and cumbersome to configure for each station, the process was automated and autonomously creates, for each station, the parameters to be monitored (number of used observations and rejected observations, RMS of the OD process and estimated bias value).

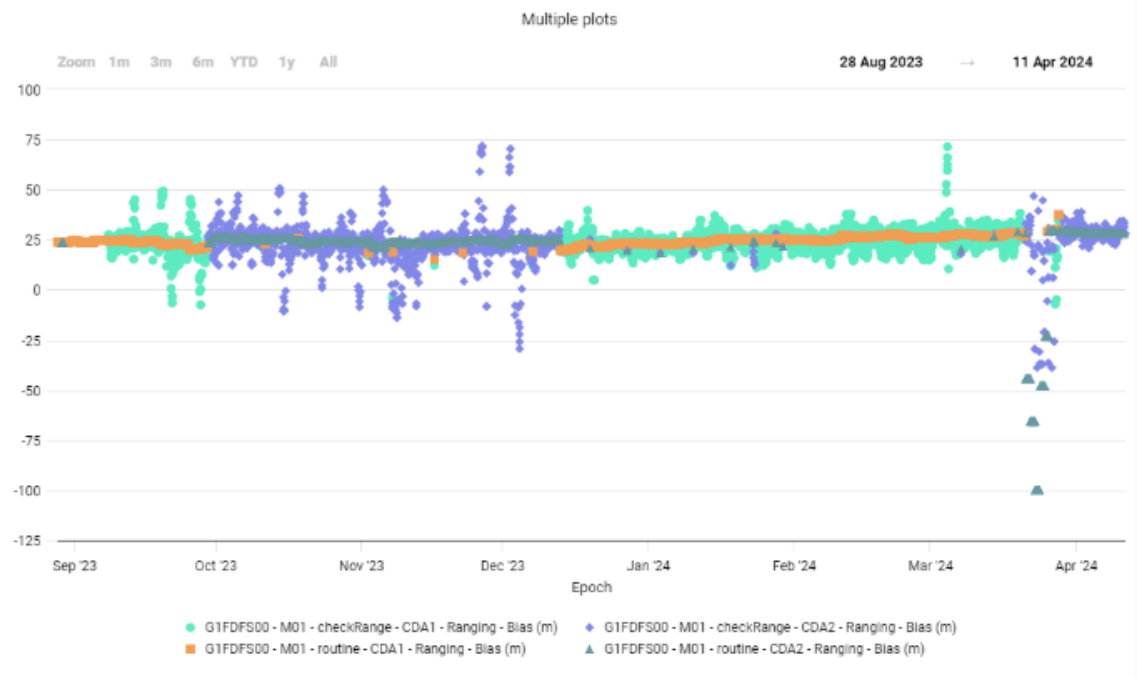

Fig. 8. Estimated ranging biases for Metop-C CDA1 and CDA2 antennas showing evidence of issues with CDA2 due to a power outage

The monitoring framework supports several operational systems: NAPEOS for LEO operations, FocusSuite for GEO operations, Geneos for future missions, and can also handle other systems in a generic way with default rules.

3.4 Visualisation and access to mission data

Most of the standard Flight Dynamics plots are made available through the dashboard and allow the user to access operational as well as historical data whenever needed.

The plotting function relies on the Highcharts library in the front-end, with the backend in charge of generating the plot data in the expected formats. The following inputs are supported:

- Data files (e.g. orbit determination residuals)
- Orbit files (NOF, IPF, OEM files)
- Data from the parameter database

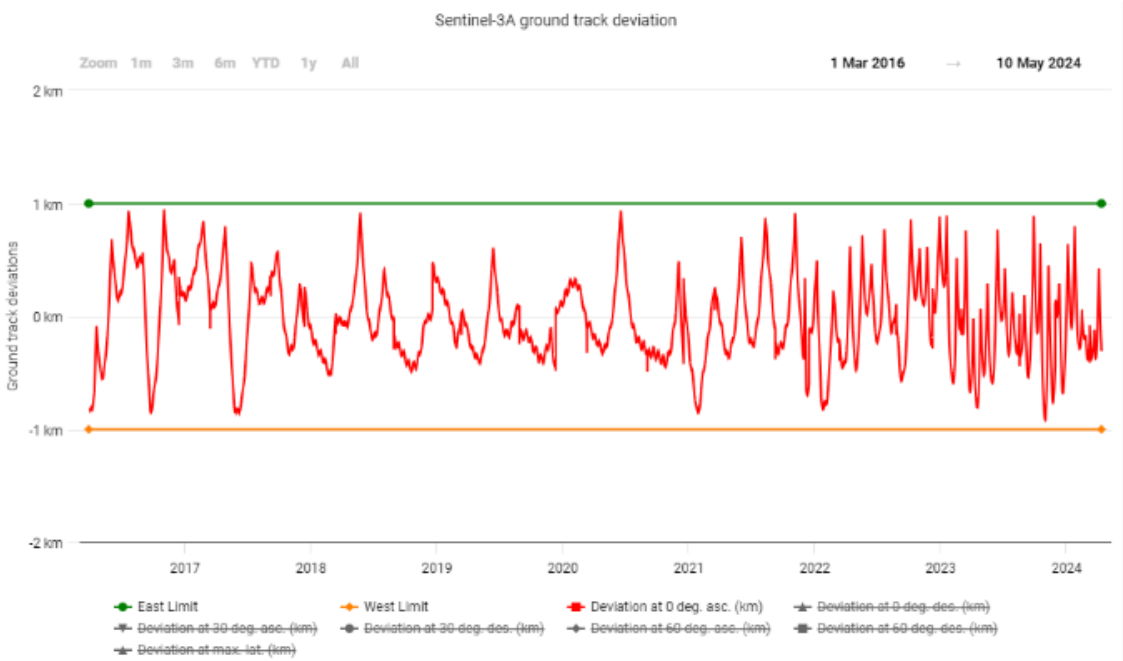

Fig. 9. Sentinel-3A ground-track deviations at ANX since launch showing evidence of increased solar activity

Plot types currently supported include:

- Osculating orbital elements
- Mean orbital elements
- Deviations wrt reference values
- OD residuals (navigation, tracking)
- Ground-track deviations
- Cross track deviations at ANX
- Orbit differences
- Orbital events timeline
- Station visibilities timeline
- Pass duration vs AOS
- Solar activity evolution and forecasts
- and more

Naturally, multiple data sources can be defined and selected for visualisation. Nominal operational files usually spanning a few weeks, long term propagations over a year, historical files since launch, minimum and maximum drag scenarios are all accessible with this system. Access to the visualised data is possible directly through the interface thanks to the export function of Highcharts.

3.5 Constellation monitoring

The visualisation framework is further augmented by the ability to compute differential values and thus monitor constellation flight parameters. Currently supported parameters include all osculating and mean orbital parameters, but also cross track deviations at ANX (used for Sentinel-6/Jason-3 tandem control).

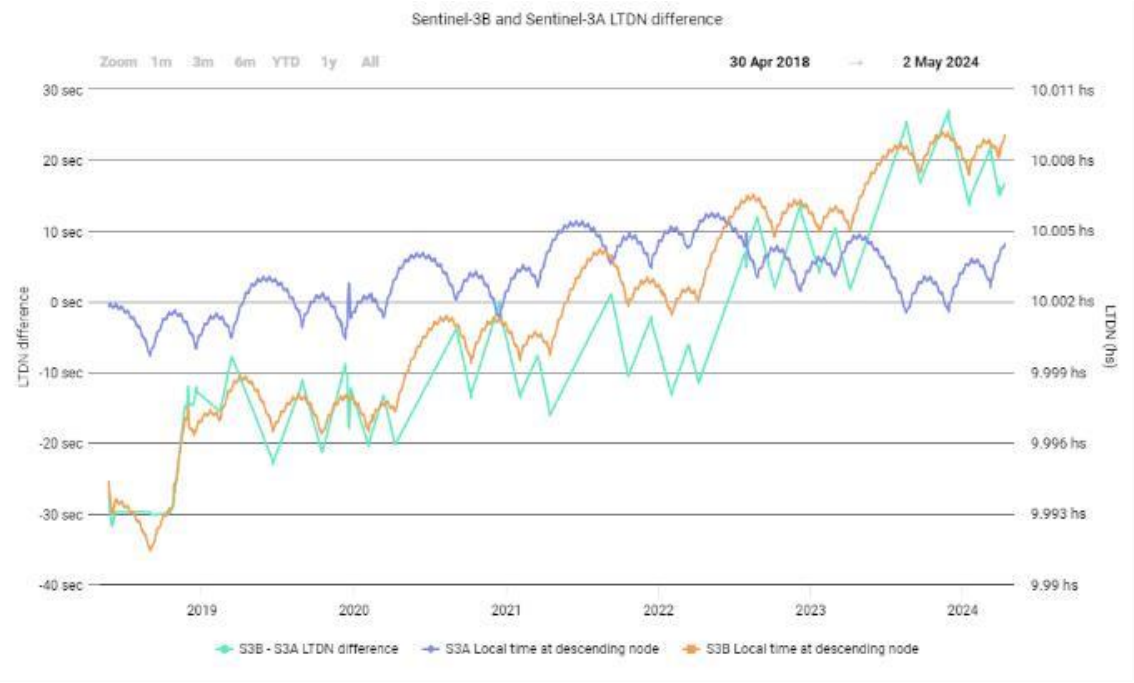

Fig. 10. Sentinel-3A and Sentinel-3B local time at descending node difference since S3B launch

3.6 Visualisation of manoeuvre information

In a similar way, a manoeuvre ingestion function is implemented and allows storing the details of each manoeuvre in the system. A view that shows the main parameters and provides access to the most important information is also provided. Figure 10. shows the dramatic impact high solar activity has on predictive capabilities (the dashed line corresponds to operational orbit propagated every morning, and it contrasts sharply with the propagation performed at manoeuvre preparation time).

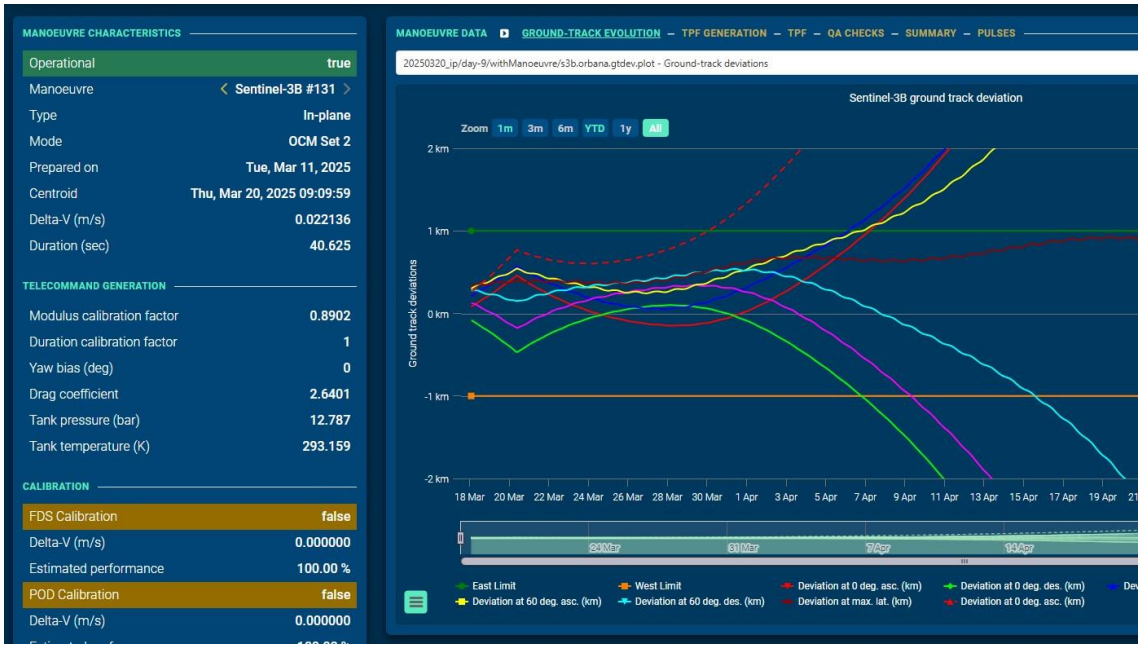

Fig. 11. Sentinel-3B manoeuvre #131

As the manoeuvre data is aggregated into standardised database tables, it is possible to derive statistical performance information and identify trends, which can directly be fed back into operations. In the case of Sentinel-3, this allowed the operations team to simplify operations by adopting fixed calibration factors per manoeuvre type and thruster set.

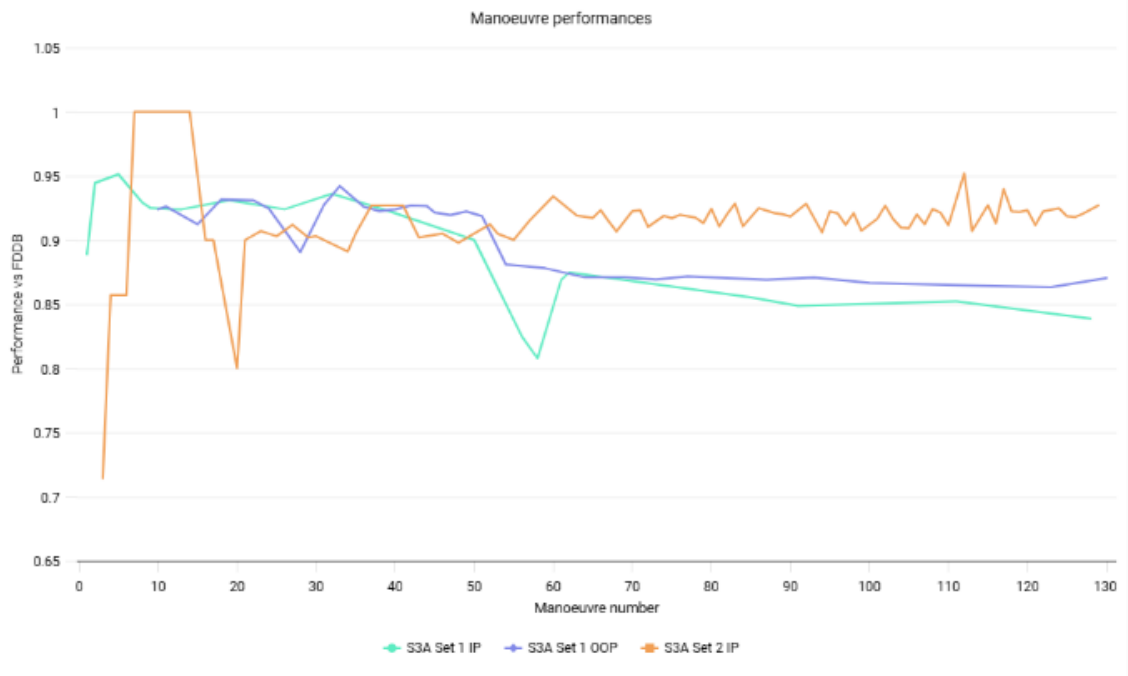

Fig. 12. Sentinel-3A manoeuvre performances

3.7 Visualisation of conjunction information

The in-house built and maintained conjunction analysis system is one of the monitored systems. It processes CDMs issued by partner entities and post-processes the conjunction information using the latest operational orbit for a more accurate estimate of the collision risk.

After scanning logs generated by that system for potential errors, a subsequent step is triggered to extract the relevant conjunction information and store it in the database. The result is a centralized situational awareness database that provides instant insight into the safety of the operated spacecraft, and detailed information about each conjunction.

Fig. 13. Riskiest conjunction on March 13th

The estimated PoC for the operational orbit as well as the independently estimated orbit are provided. The independent orbit has been found sometimes not to match the operational orbit, raising the question of knowing which to trust, and which criteria to observe, with the ensuing obvious improvements to the associated operational procedures.

Fig. 14. Metop-C vs CZ-6A 61351 conjunction

Like manoeuvres, aggregated information can be further investigated to identify trends and perform statistical analyses.

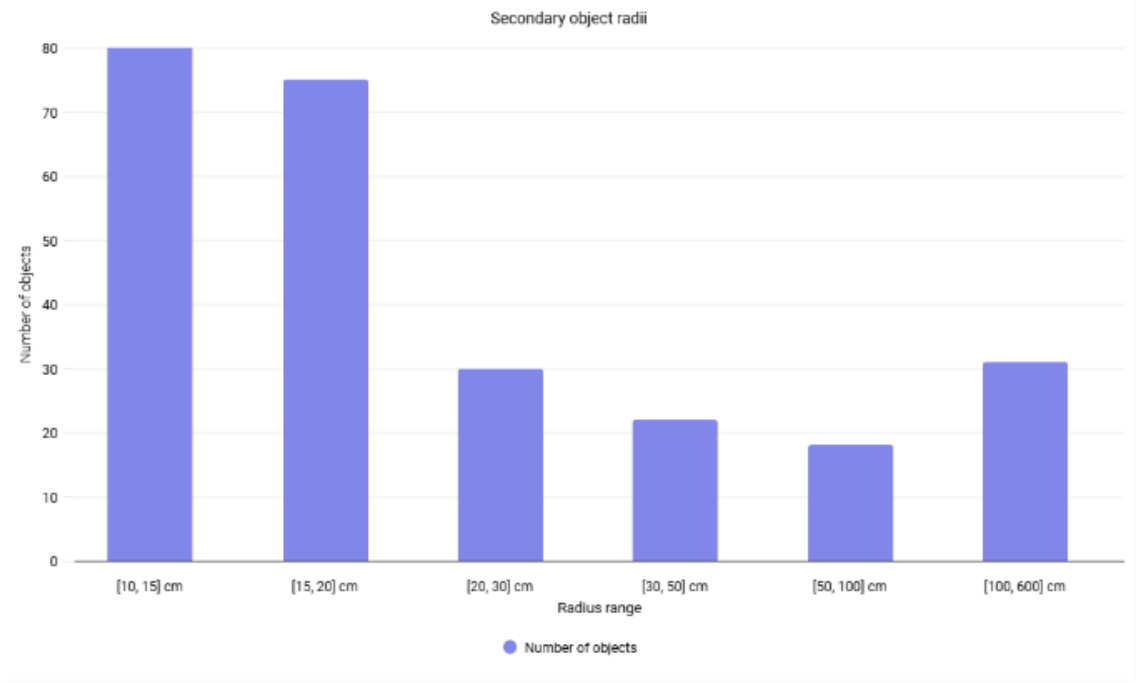

Fig. 15. Trends: Radii of secondary objects

3.8 Notifications to Teams

Further aiming at integrating the monitoring system with existing assets for streamlined workflows, an early stages notification framework has been built and send notifications relaying the key information to Microsoft Teams. Current notifications include:

- Error messages when an error is detected in a log,
- Information messages for created, updated, or calibrated manoeuvres, - Information messages sent when new tracking campaigns are detected.

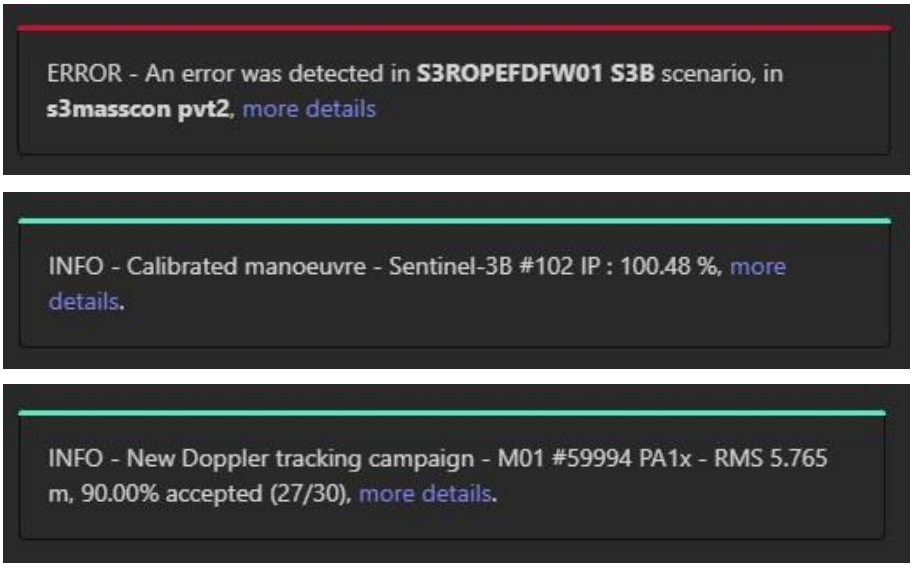

Fig. 16. Examples of notifications

4. Conclusion and way forward

The project has demonstrated the effectiveness of the chosen methodology. Successive features have been built and deployed with no outage of the main service. Despite the added complexity of building and maintaining a REST API and front-end service, the chosen architecture has proven simple and robust enough to support the large number of monitoring tasks for the many systems that are monitored.

The overarching goal of this project is to increase the efficiency and quality of actual work tasks undertaken by the operations team. Thus, future releases of the dashboard are focused on providing quicker and easier access to the data usually found in traditional emails, reports, spreadsheets, presentations, and meetings, and generally aim at progressively phasing these out by mere natural adoption.

Future work includes improvements to the notification's framework, improvements to the conjunction analysis part, onboarding of GEO FD operations, trend monitoring for the key metrics as well as potential exchanges with other teams for generalized automated operations reports. A host of other innovative features can be easily thought of, as well as new services based on the same stack and working process.