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

SpaceOps-2025, ID # 26

A Digital Twin Architecture for Operating a Large Satellite Constellation

**Ulrich Kling*, Nils-Holger Kaul, Michele Campanelli, Luca Pizzuto,
Sabrina Moser, David Hiebl & Spencer Ziegler**

Galileo Competence Center, German Aerospace Center (DLR),
Münchener Straße 20, 82234 Weßling, Germany,
*Corresponding Author, email: ulrich.kling@dlr.de

Abstract

Operating large satellite constellations in low Earth orbit is challenging, especially with the increasing number of satellites and space debris. In this context, digital twins, digital models synchronized with physical assets, have the potential for cost-effective and sustainable constellation management. Digital twins, widely used in industries like aviation, can simulate scenarios, predict maintenance needs, detect faults, and enable autonomous operations.
For satellite constellations, digital twins could manage tasks across various phases, from launch and early orbit (LEOP) to routine operations and end-of-life. Initially, operations may be manual, but as data and experience grow, digital twins can take on a greater role. The paper proposes an architecture for integrating digital twins into ground control systems without altering existing setups, enabling retrofitting to current missions. It introduces a testing approach using an operational simulator before deployment.

Keywords: Satellite Operations, Digital Twin, Artificial Intelligence, Root Cause, Automation, Satellite Constellations

Acronyms/Abbreviations

AI	Artificial Intelligence	LLM	Large Language Model
DT	Digital Twin	TC	Telecommand
LEO	Low Earth Orbit	TM	Telemetry

1. Introduction

Large satellite constellations in Low Earth orbit (LEO) are currently being operated by, for example, Starlink, Eutelsat OneWeb and Planet. In addition to these, further constellations consisting of hundreds if not thousands of satellites are being discussed, planned and commissioned by commercial, institutional and military entities in North America, Europe and Asia for applications such as communication, internet from space, earth observation, remote sensing and navigation. Operating a large constellation poses in itself a new technological challenge that grows when needing to fly alongside other constellations and ever-increasing space debris. Therefore, how does one operate large satellite constellations cost effectively and sustainably?

In various industrial sectors, such as aviation, Digital Twins (DT) are already employed across all phases of a project lifecycle, from design through to operations and maintenance. A DT encompasses a digital model of the physical asset, where the telemetry is automatically processed and remains synchronized at all times with the physical asset. Furthermore, the ability to simulate and use artificial intelligence techniques, e.g. machine learning, allows the DT to perform what-if test scenarios and predictive maintenance, as well as fault detection, self-healing and a step towards autonomous operations.

A large satellite constellation will likely consist of identical satellites manufactured in a series of batches, which allows a data driven approach to be leveraged. For this reason, we believe a DT is a promising approach for the operation of a satellite constellation. In this paper we address sample use cases from the launch and early orbit phase (LEOP), routine operations and end of life management for the digital twin. Moreover, for these use cases we consider from which point during the constellation deployment a DT could be employed. For example, the launch of the initial demonstrator satellites or first production batch would likely be operated manually, whereas with more data and operational experience a DT could handle aspects of the LEOP for later batches.

In the paper we present an architecture for how the DT can be integrated into the ground control segment without requiring changes to a ground segment's standard elements. This has the added benefit that DTs can be retrofitted to

or tested with existing missions. Finally, we present a test and validation approach based on an operational simulator for the constellation that allows the digital twin to be tested prior to its use in actual operations.

2. Digital Twin Definition

There is no standardized definition for a DT. Rather, it encompasses a collection of ideas tailored to various use cases across different phases of the product lifecycle. Generally, a DT involves replicating a physical object as a digital counterpart, meaning that all available data about the real object is stored, continuously updated, and used for analysis and decision-making.

In this context, a more specific definition of a DT focuses on the data flow between the real-world object and its digital representation. This can be categorized into a digital model, a digital shadow, and a DT, see Fig. 1.

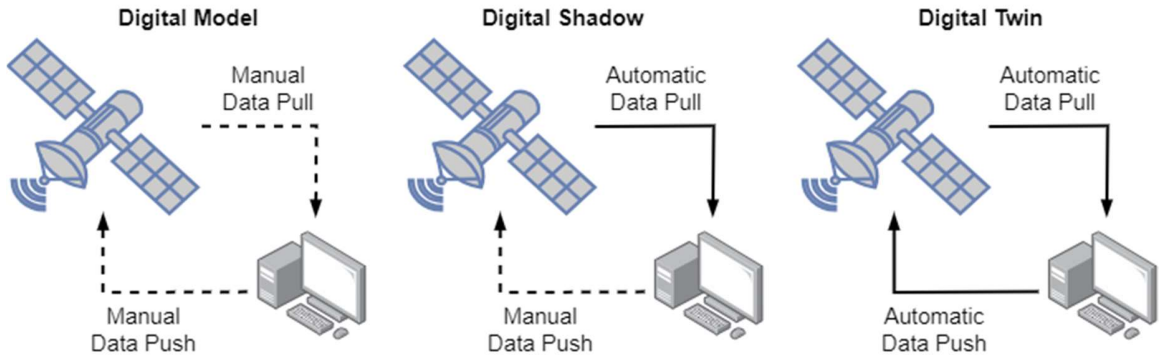

Fig. 1: Definition of a Digital Twin

For a digital model, the data flow between the real object and its digital representation is handled manually, resulting in a significant workload for a human operator. In the case of a digital shadow, the data generated by the real-world object is automatically transferred to its digital counterpart. Finally, a DT is achieved when the data exchange between the real object and the digital object is performed automatically.

This implies that the data generated by the real object is automatically analyzed by the digital object, and decisions are derived, including which data (e.g., commands) is sent back to the real-world object.

In satellite operations, this means that the planning of telemetry (TM) to be downloaded must be automated. Subsequently, the received telemetry must be autonomously analysed. If abnormal behaviour is detected in the satellite or any of its subsystems, an assessment is needed to determine the severity of the issue and which commands must be sent to ensure the satellite fulfils its mission objectives.

Initially, a human operator can assess anomalies and prepare the corresponding commands, as this requires specific knowledge about the satellite, its functions, and the relationships between its subsystems and payloads. However, to achieve a “full” DT, this step must also be automated. Decision trees are one possible implementation for such automation [1].

These relationships lead to a system architecture that incorporates machine learning algorithms to analyze satellite data and may include one or more satellite simulators for what-if scenario analyses to support decision-making in the event of deviations in satellite behavior. A similar setup has been presented in previous research. Additionally, machine learning algorithms can be utilized to plan onboard satellite activities, such as managing power resources for inter-satellite links in a constellation.

3. Digital Twin Overview

DTs in space have emerged as a transformative technology that enables the enhanced monitoring, simulation, and optimization of space missions. This innovative approach has significantly impacted spacecraft and satellite design,

operational efficiency, and predictive maintenance. With ongoing advancements, DTs are poised to play a central role in deep space exploration and satellite network management.

The concept of creating digital representations of physical systems dates back to the 1960s when NASA pioneered early models known as "living models". These early efforts aimed at predicting system behaviours, enabling better decision-making, and improving the design of space systems. These "living models" were primitive digital representations of spacecraft and satellite systems designed to monitor their real-time behaviour. This foundational work set the stage for what would later become known as digital twins.

The term "digital twin" was coined by Dr. Michael Grieves in 2003 at the University of Michigan during his work on product lifecycle management.[2].

He proposed the concept of a virtual replica of physical systems, which could be updated in real-time and used for performance monitoring and predictive maintenance. The concept evolved to emphasize the connection between physical and virtual systems, allowing for accurate predictions of system behaviour and proactive interventions to prevent failures. As Grieves stated in 2015, DTs represented a new paradigm in manufacturing excellence by replicating systems virtually and using real-time data to monitor and optimize performance.

The adoption of DTs in space accelerated in the 2010s with the integration of Internet of Things (IoT) technologies. IoT-enabled sensors on spacecraft and satellites allow for the continuous collection of real-time data, feeding directly into DT models. This development has enabled space systems to become more dynamic, capable of simulating current conditions and predicting future performance with high accuracy [3].

According to Fuller et al. (2020), this integration of IoT technologies facilitated the real-time monitoring of spacecraft, allowing for more precise control and optimization of operations, as well as predictive maintenance. The integration of Artificial Intelligence (AI) and machine learning with DTs has further enhanced their capabilities. Rathore et al. (2021) highlight the critical role AI and machine learning play in improving the predictive analytics of space systems. These technologies enable the real-time analysis of massive data sets, optimizing operations and allowing for proactive decision-making based on the performance and health of the spacecraft or satellite. AI-driven predictive models help forecast potential failures, streamline maintenance, and optimize system designs for future missions [4].

NASA has been at the forefront of utilizing DTs for space exploration. One of the most notable examples is the Mars Perseverance Rover mission. NASA used a DT of the rover for simulations, performance tracking, and predictive maintenance, allowing engineers on Earth to monitor the rover's systems and predict potential issues before they occurred [5]. This DT provided valuable insights that contributed to the rover’s successful landing and ongoing operations on Mars.

In addition to Mars missions, NASA is developing DTs for future lunar exploration. These digital models simulate the lunar environment, helping to optimize the design of equipment and ensure its reliability during missions. By creating virtual representations of lunar rovers, habitats, and other critical systems, NASA can predict how they will behave in the harsh lunar environment and address potential issues proactively [6].

DTs have evolved from their origins in NASA's Apollo missions to play a critical role in various fields like personalized medicine, autonomous space operations, and manufacturing. NASA first introduced DTs after the Apollo 13 incident to address spacecraft challenges. Today, DTs are crucial for monitoring complex systems, such as the James Webb Space Telescope. They help test and simulate its temperature control and the delicate deployment of its sunshield [7].

Looking ahead, DTs are expected to play an increasingly central role in deep space exploration and the management of satellite networks. The continued integration of advanced technologies such as AI, IoT, and machine learning with DTs will drive further improvements in mission success rates, safety, and efficiency. As deep space exploration becomes more ambitious, the complexity of systems required to support long-duration missions will also increase. DTs will be critical in monitoring these complex systems, predicting failures before they occur, and enabling real-time adjustments to ensure the success of missions. The future of space exploration will rely on DTs to enable more efficient, safer, and cost-effective operations, reducing the risks associated with deep space missions and satellite network management.

With advancements in IoT, AI, and machine learning, DTs will continue to play a central role in shaping the future of space exploration, from lunar missions to deep space exploration, promising a new era of precision and efficiency in space operations.

Artificial Intelligence (AI) can play a major role in the set-up of a DT. It can be used to support four different classes of DT, namely Passive, Predictive, Reactive, and Autonomic [8], see Fig. 2. Each of the DT types has a different objectives and AI can be utilized to perform the corresponding tasks intelligently to fulfill these objectives.

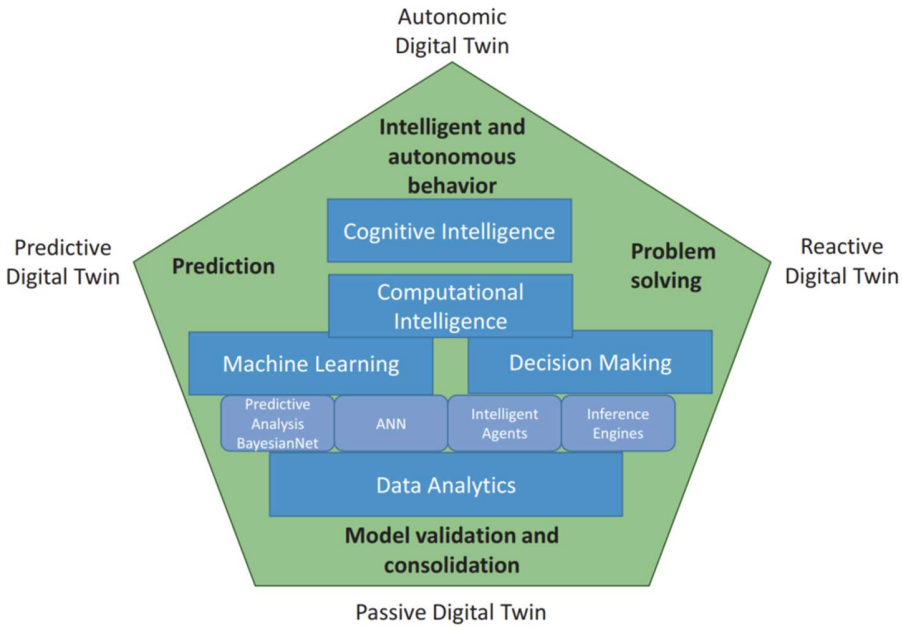

Fig. 2: AI technologies supporting different types of Digital Twins [8]

4. Impact of automation in large satellite constellation operations

In a previous study [9] the impact of launch cadence and automation on the economics of constellation operations were explored. The analysis led to several key insights that shed light on the operational dynamics of large satellite constellations.

By investigating publicly available launch data from several active constellation operators, several notable trends were identified that provide valuable benchmarks for the deployment cadence and ramp-up strategies of future constellations. First, operators typically launch a series of test or in-orbit demonstration satellites before scaling up their constellation with production-grade satellites. The test phase duration varies between 0.7 and 2.5 years. Additionally, the average launch cadence ranges from once a month to once every 4-6 months, with the deployment of production-grade satellites occurring at a linear pace over time. Interestingly, once the ramp-up of the constellation begins, accelerating the deployment rate proves to be challenging and remains relative consistent over time.

Further, the reported manpower requirements and the impact of automation on the size of operations teams for three active constellations and compared and an efficiency metric is developed based on the number of satellites an operator can manage throughout the lifecycle of a constellation. The evolution of manpower across these constellations exhibited similar patterns, with the metric increasing at different rates over time, allowing to derive an unweighted average that serves as a useful benchmark for understanding how satellite operations teams might scale and automate as constellations grow.

These insights can be applied to analyze a potential LEO-PNT constellation to assess when such a constellation would become operational, the required size of the operations team, and how the team might evolve over time as automation increasingly takes over routine tasks. Additionally, the impact of different launch cadences on the sizing and efficiency of the operations teams were examined, as illustrated in Fig. 3.

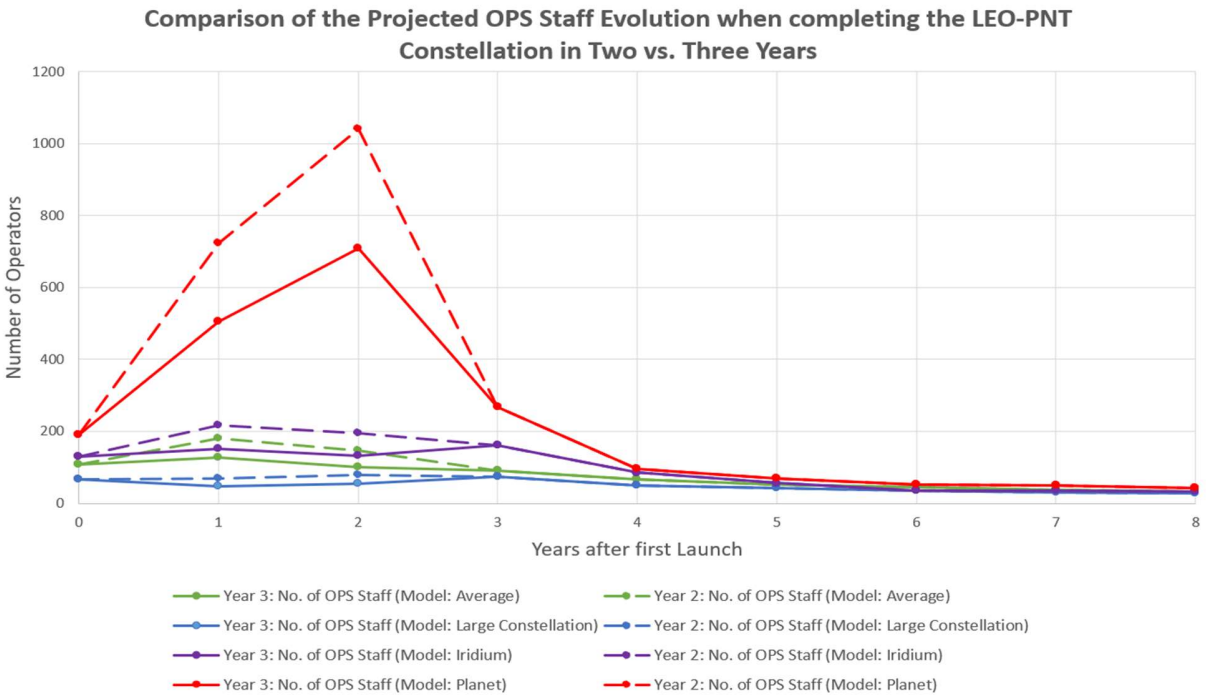

Fig. 3: Comparison of the Projected number of Operators needed to complete LEO-PNT in two vs three years, assuming different automation efficiency models.

Our study suggests that while DT technologies hold significant potential, their optimal application will likely be in the routine operations phase and during the management of the constellation's end-of-life stages. Indeed, there are considerable challenges to using DTs in earlier phases, particularly during the Launch and Early Orbit Phase (LEOP). This is mainly due to the limited amount of data available to build a reliable DT model and the intrinsic difficulty and uncertain nature of a LEOP. Automation in this phase is a complex task that demands near-perfect reliability, particularly for high-stakes missions consisting of a limited number of satellites, such as GNSS missions, or single-spacecraft operations, where even minor failures can have catastrophic consequences.

Moreover, DTs typically struggle to predict future spacecraft states due to the intrinsic difficulty of accurately modelling and simulating the space environment. This limits their effectiveness in decision-making processes that require real-time adjustments and foresight. Additionally, for many GNSS operators, onboard DTs are unnecessary as constant communication with satellites allows operators to respond to issues as they arise.

Although many operators remain skeptical about investing in onboard DTs or even ground-based DTs due to concerns over their reliability, there is still interest in their potential, especially for nominal operation. In this phase, DTs could help optimize maintenance and improve operational efficiency, but for this to happen, more reliable and robust models are necessary. Currently, the return on investment for such technologies, especially during early deployment stages, is not deemed high enough to justify widespread adoption or financial commitment.

That said, the challenges associated with incorporating DTs into phases beyond routine operations should not be seen as insurmountable. Further research and development are required to address the unique challenges of the space environment and improve the predictive capabilities of DTs. As more studies are conducted, it is likely that DTs will become a viable option for automating and optimizing a broader range of mission phases, from LEOP to end-of-life management. The potential benefits of DTs in these areas are significant, and with continued advancements, we may see their adoption expand to all phases of satellite constellation operations in the future.

5. Digital Twin Design

One of the key challenges of a DT is the autonomous decision making in case of an anomaly in the received satellite telemetry. Of course, the problem is not only to make a decision, but also to make the *right* decision. This is a highly complex task and requires cognitive intelligence.

5.1 Reasoning Chain

In this section a reasoning chain with different steps, which have to be passed when encountering an anomaly in the telemetry data of a satellite, is described and discussed. The different steps are classified regarding the complexity of the task and decision that have to be made. In Fig. 4. the reasoning chain is displayed.

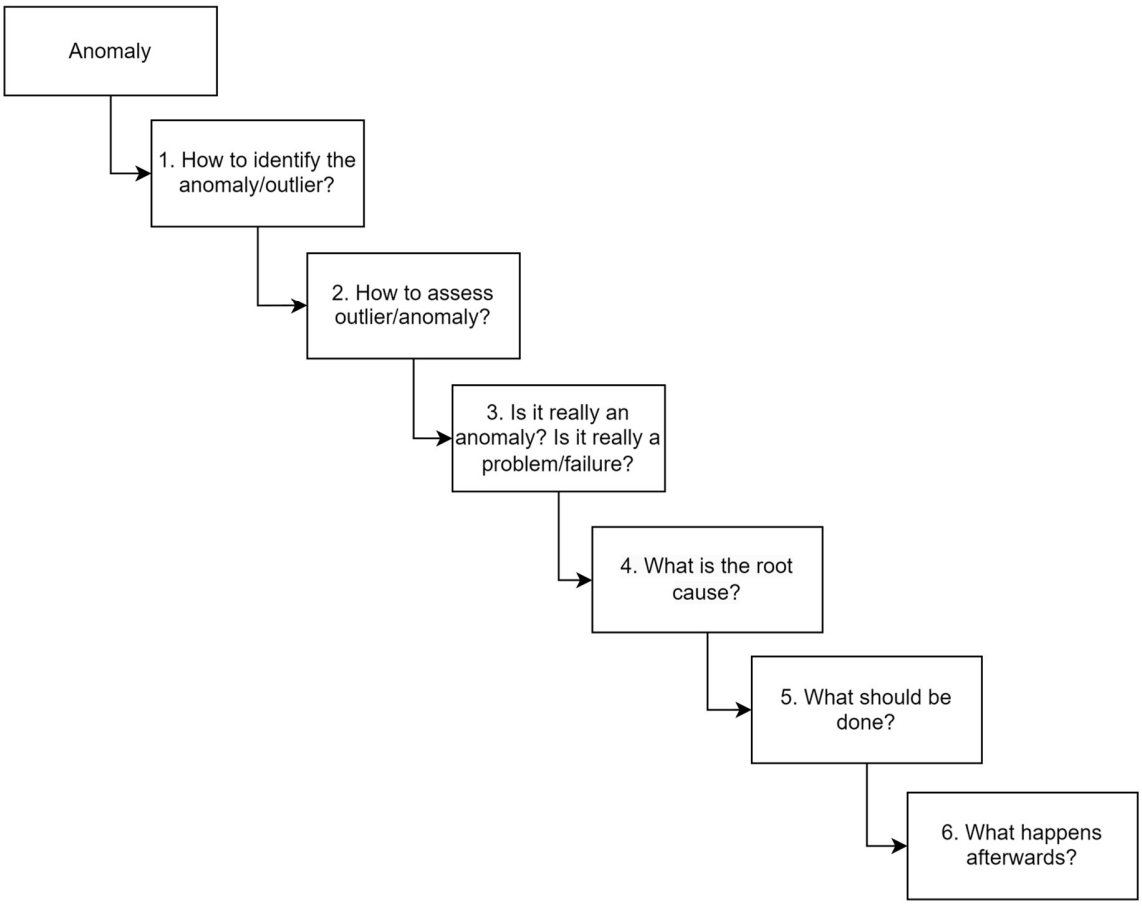

Fig. 4: Reasoning chain for decision making

If an anomaly is detected in the telemetry dataset, the first step is to identify the anomaly itself. For this step several possibilities can be used, see Fig. 5. The traditional approach is to use limit checkers with certain thresholds and whenever one or several thresholds are violated, the operator is informed. This approach is very simple and easy to implement, but the drawbacks are obviously. This approach does not allow to detect anomalies, which do not violate the limit. Also, long time trends such as degradation are not detected. Another possibility which shall overcome these shortcomings of the limit checkers and which is investigated with great effort these days is to use Machine Learning (ML) algorithms. One questions that connected to the use of ML is how to constantly update your ML model and how to retrain it. Furthermore, how much data is required to set-up the model itself? How long needs the satellite be in operation to generate enough (real world) data to be able to have a reliable working ML model? For a constellation, a large number of identical satellites within the constellation can result in synergies for the generated training data. However, the telemetry data of different satellites of a constellation, although technically identical, may still be different due to different orbits and onboard configurations.

Several methods can be employed for anomaly detection, ranging from traditional rule-based systems to modern machine learning (ML) approaches. These methods vary in complexity, adaptability, and suitability depending on the type of anomaly and the available data.

A common baseline method is the use of out-of-limit (OOL) checks, where predefined thresholds are applied to telemetry parameters. Whenever a value exceeds these static thresholds, it triggers an alert. This approach is straightforward, requires no training data, and is easy to implement. However, it lacks adaptability and fails to detect anomalies that remain within the nominal bounds, such as early-stage degradation or multi-parameter anomalies. It is also ineffective in capturing subtle or evolving patterns in the data [10].

In contrast, machine learning methods offer increased flexibility and performance, especially when dealing with complex or previously unseen anomaly types. Supervised ML approaches rely on labeled datasets to learn how to classify known anomaly types. While effective in recognizing familiar patterns, they are inherently limited by the availability and completeness of labeled data. They struggle to detect novel anomalies, which is a critical limitation in space systems where unexpected behaviors frequently occur [11].

To overcome this, unsupervised ML methods have gained traction. These models are trained solely on nominal data, learning the normal behavior of the system and flagging deviations as potential anomalies. This makes them particularly well-suited for detecting new or rare anomaly types without relying on prior labeling [12]. Moreover, their ability to generalize from normal operations allows them to adapt to real-world conditions where new anomalies emerge over time.

An additional layer of complexity arises from the variety of anomaly types, each with distinct characteristics. Robust ML architectures must be capable of identifying both sudden anomalies—abrupt deviations in telemetry that cannot be predicted in advance—and trend anomalies, which evolve gradually over time. While sudden anomalies typically require reactive detection mechanisms, trend anomalies open the door to predictive methods such as anomaly forecasting. In this context, deep learning is particularly promising, as it can model complex dependencies across multiple telemetry channels and uncover latent patterns that precede system degradation [13].

State-of-the-art approaches include architectures such as Long Short-Term Memory (LSTM) networks, which are widely used for sequential modeling and detecting temporal anomalies [14]. More recently, transformer-based models, such as Informer [15] and TimesNet [16], have demonstrated superior performance in capturing long-range temporal dependencies, making them well-suited for both anomaly detection and forecasting in telemetry time series. For unsupervised anomaly detection, Variational Autoencoders (VAEs) [17] and Deep Support Vector Data Description (Deep SVDD) [18] are popular for learning compact representations of nominal behavior. Additionally, GAN-based models such as MAD-GAN (Multivariate Anomaly Detection GAN) [19] and newer approaches like AnomalyBERT [20] use generative or self-supervised architectures to identify discrepancies between real and learned data distributions, which is especially powerful in multivariate telemetry data.

A third possibility could be to install a satellite simulator on ground, which simulates the satellite. When receiving the telemetry, the real telemetry is compared to the simulated telemetry. A difference in the received real telemetry and the synthetic simulated telemetry indicates an anomaly. However, this approach requires a very realistic satellite simulator. Moreover, after detecting an anomaly, the simulation is out of sync with the real satellite. Therefore, the question is if the simulation is now helpful to explain and eliminate the anomaly.

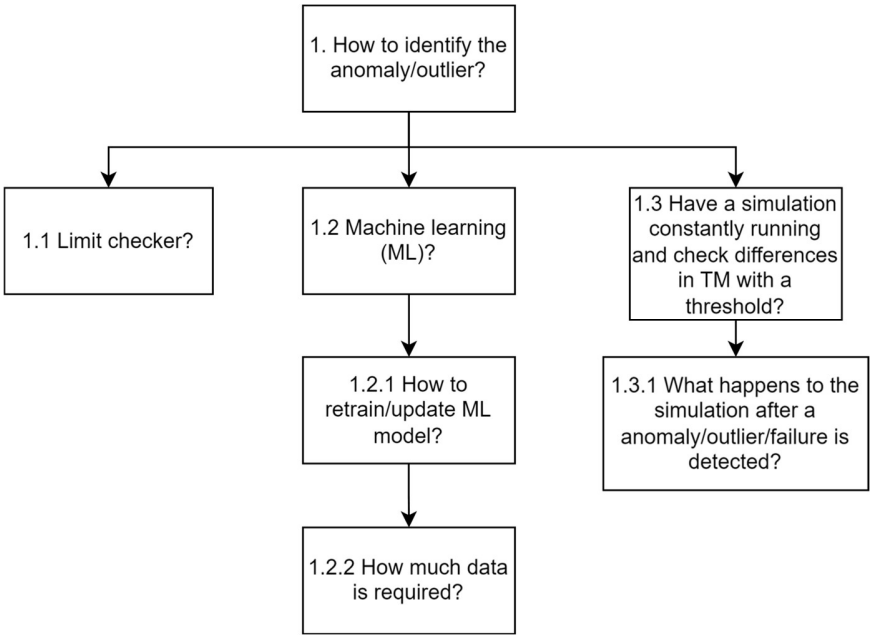

Fig. 5: Distinction how an anomaly can be detected

In the next step the anomaly has to be assessed. Fig. 6 shows the possibilities of deviation the anomaly can have from the expected telemetry values. Depending on how the received values deviate from the expected values, e.g. the value is too high, too low, a new pattern and/or several parameters are affected. The type of deviation has to be determined. This requires a mathematical formulation to describe the type of deviation that can serve as input for the next step. Of course, when using a limit checker and a value is too high, the indication is quite simple to implement. But in case a ML algorithm is used and an anomaly is detected, the distinction has to be made subsequently.

After detecting an anomaly, the next step is to assess its characteristics and classify the type of deviation from expected telemetry behavior. In the literature, most anomalies are categorized into point anomalies (a single data point deviates significantly), contextual anomalies (a data point is anomalous in a specific context, such as time or operating mode), and collective anomalies (a sequence or group of values deviates collectively). Anomalies can manifest in several ways: a parameter may exceed an upper or lower limit, deviate gradually over time, display an unexpected pattern, or involve multiple parameters simultaneously. Identifying the type of deviation—whether it is sudden, trending, or affecting a correlated group of signals—is essential for understanding the context of the anomaly and determining appropriate follow-up actions.
While threshold-based approaches implicitly convey the anomaly type (e.g., a value is too high), this is not the case for many ML-based methods, particularly those using unsupervised learning. In such cases, the anomaly must be interpreted post hoc. This requires a mathematical description of the deviation that can serve as an input for subsequent analysis or decision-making. Possible approaches include simple deviation profiling using statistical measures such as magnitude and direction, or time-series comparison techniques to identify pattern changes and shifts in behavior. Additionally, context-aware classification models—either rule-based or machine-learned—can support the labeling of anomalies based on their temporal and parameter-based signatures.
In more complex systems, explain ability tools such as feature attribution (e.g., SHAP values), saliency maps, or attention-based mechanisms can be employed to understand which input features or time segments contributed most significantly to the anomaly score. In autoencoder-based systems, the distribution of reconstruction errors across the input can indicate which parameters or time windows deviate most from the learned nominal behavior. These techniques enable the categorization of anomalies into meaningful types—such as spikes, drifts, collective anomalies, or correlated deviations—laying the groundwork for root cause analysis, mitigation planning, or predictive maintenance.

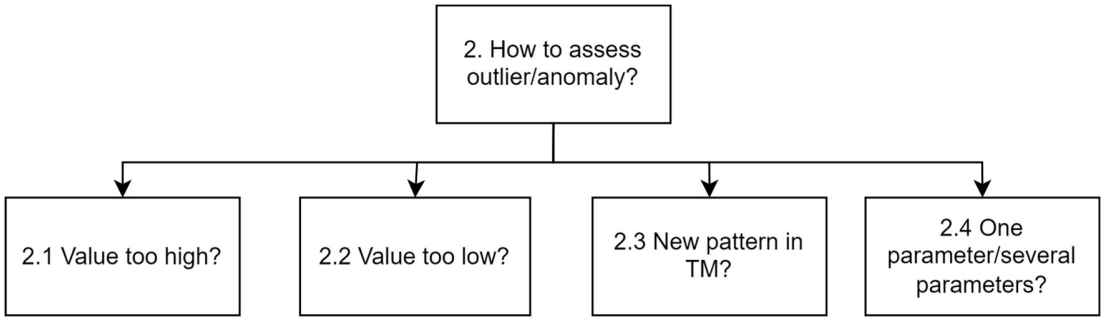

Fig. 6: Assessment of anomalies

In the third step, see Fig. 7, it has to be ensured that it is a real anomaly, e.g. a malfunction or system failure of the satellite or one of its subsystems and not a rare nominal event, and hence, a false positive alarm. In this context, especially the relation between executing a command onboard the satellite addressing on subsystem and the resulting change in telemetry for the corresponding subsystem or even another subsystem is important.

One commonly used method is command-log correlation, where anomalies are checked against recent onboard activities to determine if the observed telemetry changes are a consequence of planned operations. This can involve simple timestamp matching or more advanced sequence analysis to understand cause-effect chains between commands and telemetry behavior. Metadata tagging is another useful strategy, where telemetry data is annotated with contextual labels such as "post-maneuver," "safe mode," or "payload calibration," enabling models or operators to differentiate between expected and unexpected deviations. In addition, context-aware ML models can be trained to incorporate event metadata or operational modes as input features, improving their ability to distinguish between rare nominal states and genuine faults. Regardless of the specific approach, establishing traceability between commands, system configuration, and telemetry evolution is critical to minimizing false positives and ensuring robust anomaly classification.
In this context, DTs offer a particularly powerful capability: by simulating the expected behavior of a satellite in response to commands or operational changes, a DT can serve as a reference for verifying whether a detected anomaly aligns with anticipated telemetry behavior. This enables a more informed distinction between true anomalies and rare nominal events, making DTs a highly valuable asset for automated anomaly validation and root-cause analysis.

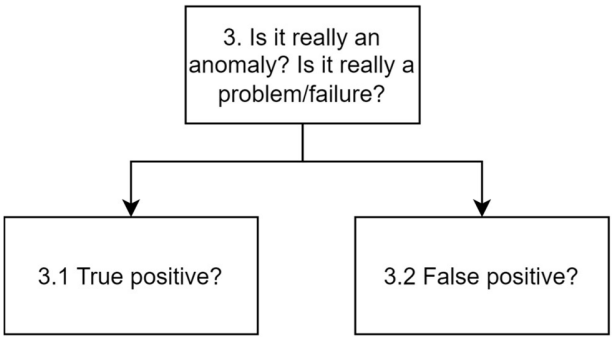

Fig. 7: Distinction between true positive and false positive

In step number four, the root cause of the anomaly has to be evaluated, see Fig. 8. This step is critical and very complex. It requires cognitive intelligence and knowledge of the satellite and its subsystems. Is the anomaly indicating a known failure that was already predicted as a possible failure for example in the design phase? Hence, a comparison has to be carried out between the received telemetry and the description of possible/predicted failures or simulated synthetic data of predicted failures to identify the present failure. Moreover, it has to be checked how many subsystems are affected and if is it only one failure or several failures at the same time. Finally, if the failure is identified, it can be proceeded to the next step in the reasoning chain.

Once an anomaly has been confirmed, the next step is to identify its root cause, see Fig. 8. Root cause analysis requires a deep understanding of the satellite's architecture, including its subsystems, failure modes, and functional interdependencies. A key challenge lies in determining whether the anomaly corresponds to a known failure scenario, potentially one that was anticipated during the design or qualification phase. To support this, a comparison can be made between the observed telemetry pattern and a repository of predicted failure signatures, which may include analytically derived models or simulation-generated synthetic data. Additionally, it is important to assess the extent of the anomaly: whether it is localized to a single subsystem or indicative of a multi-fault condition affecting several components.

This step often relies on knowledge-driven techniques, such as fault trees, dependency graphs, or failure mode libraries, as well as data-driven methods like clustering, Bayesian inference, or causality analysis to explore possible fault origins. By integrating domain knowledge with telemetry analysis, the goal is to isolate the minimal set of components or processes likely responsible for the anomaly.

DTs can play a particularly valuable role here, as they can simulate system behavior under various failure conditions, allowing engineers or autonomous systems to compare observed telemetry against realistic fault scenarios. This enables a more systematic, traceable, and data-rich approach to root cause identification, especially in complex systems with interdependent subsystems.

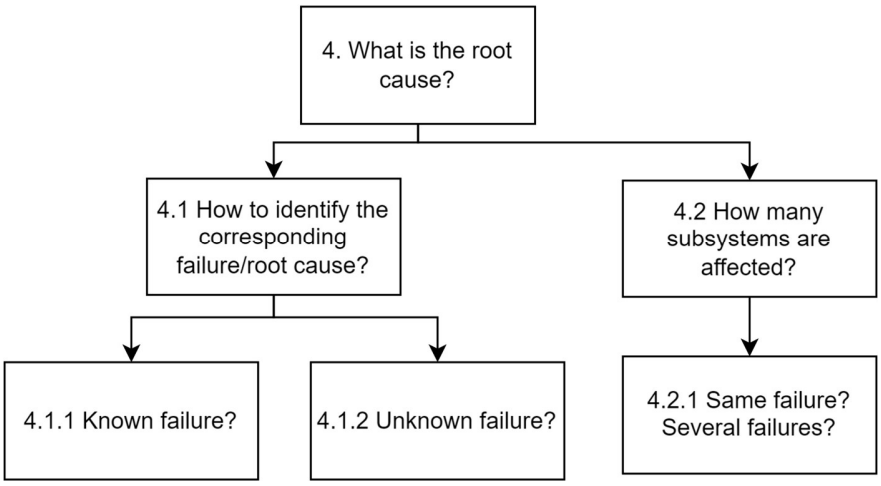

Fig. 8: Finding the root cause

In this stage, it has to be decided what to do as response to the anomaly, see Fig. 9. If the anomaly corresponds to a known failure, the overarching goal at this specific point in time needs to be considered, such as does the satellite has to perform a certain task that needs to be executed and is influenced by the anomaly. This may correlate to a set of commands that can be sent and a process needs to be carried out, to select the preferred commands and to evaluate between different possibilities.

If the failure is unknown, a what-if simulation could be a possibility to identify the root cause or to evaluate the impact. However, to start a what-if simulation automatically is very difficult. The received telemetry containing the unknown failure data needs to be analysed to define the start conditions of the simulation and the failure behaviour has to be implemented. The goal of the simulations has to be defined as well. Shall the failure be propagated in the simulation with incorporating a high simulation speed to investigate the consequences of the failure and how severe these are? Or to simulate several alternatives of commands, which should be sent in order to recover from the failure. In order to evaluate the results of the simulation(s), a further assessment has to be performed. This assessment as well requires intelligence and knowledge to interpret the results and if another simulation run or scenario has to be executed.

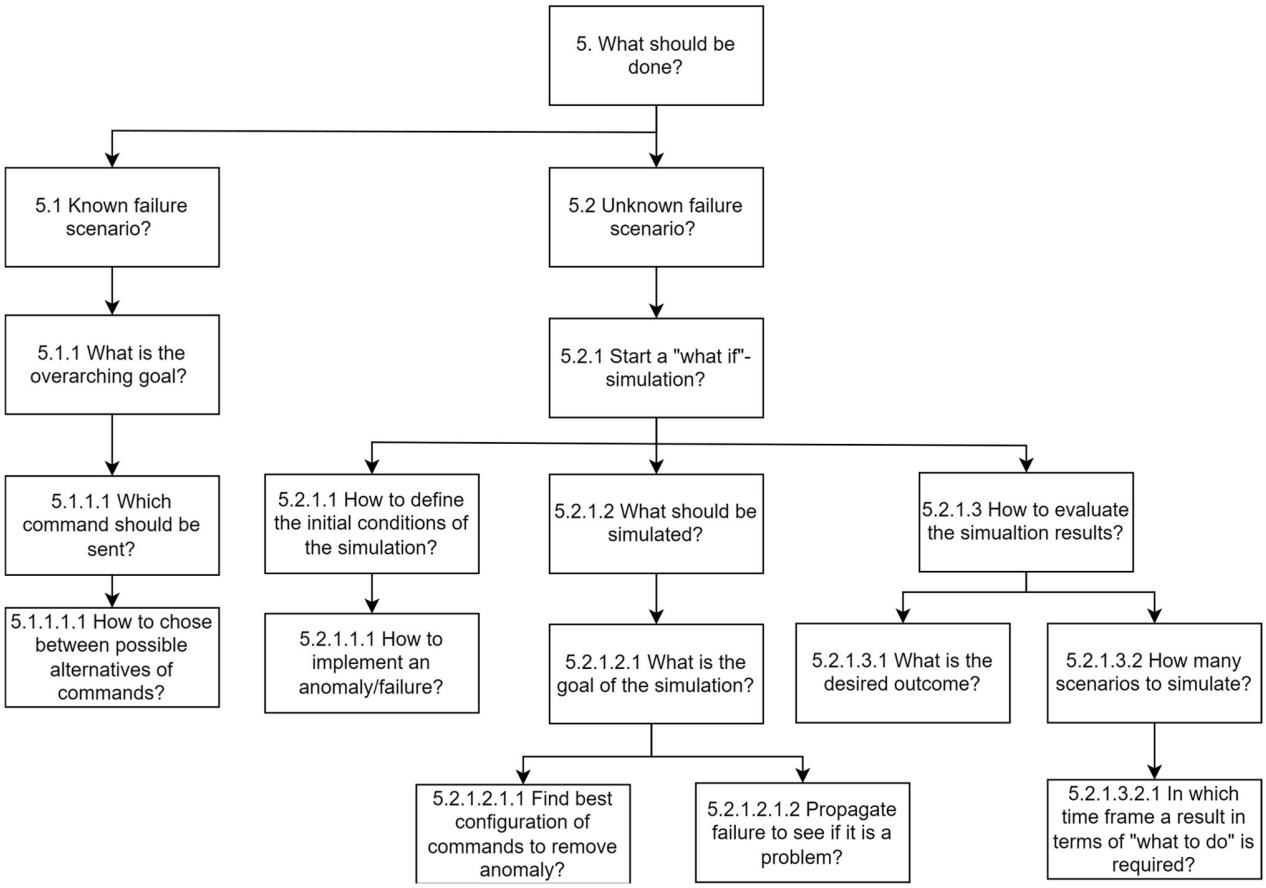

Fig. 9: What has to be done after a decision

Finally, after sending commands to the satellite as the outcome of the decision process, the newly received telemetry has to be treated as expected data and may now reflect the new nominal situation, see Fig. 10. However, in case the received telemetry values do not match with the expected behaviour, further analysis and actions are required. The system also needs to be able to identify new anomalies in the following.

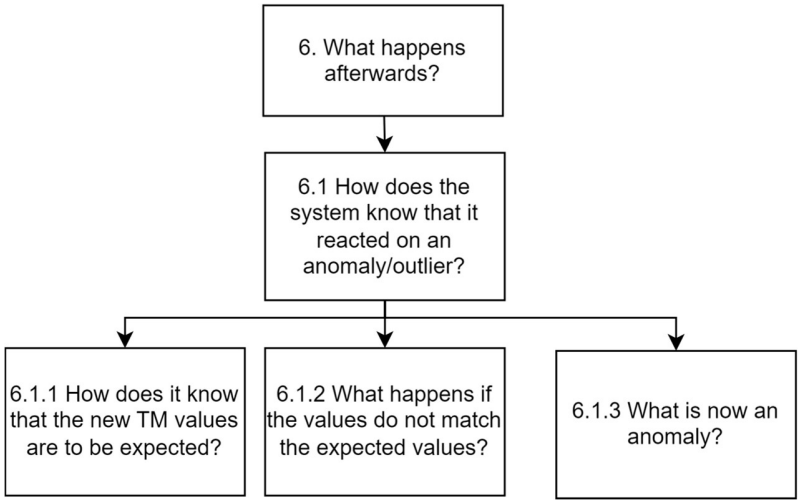

Fig. 10: What happens after the decision was executed

5.2 How to Design a Digital Twin

To be able to autonomously evaluate the different steps of the assessment process and to come to a decision on how to react to an anomaly, the following design process is proposed, in which the DT is included already in the development process and is used to prepare the operational phase of the satellite and/or the satellite constellation, see Fig. 11.

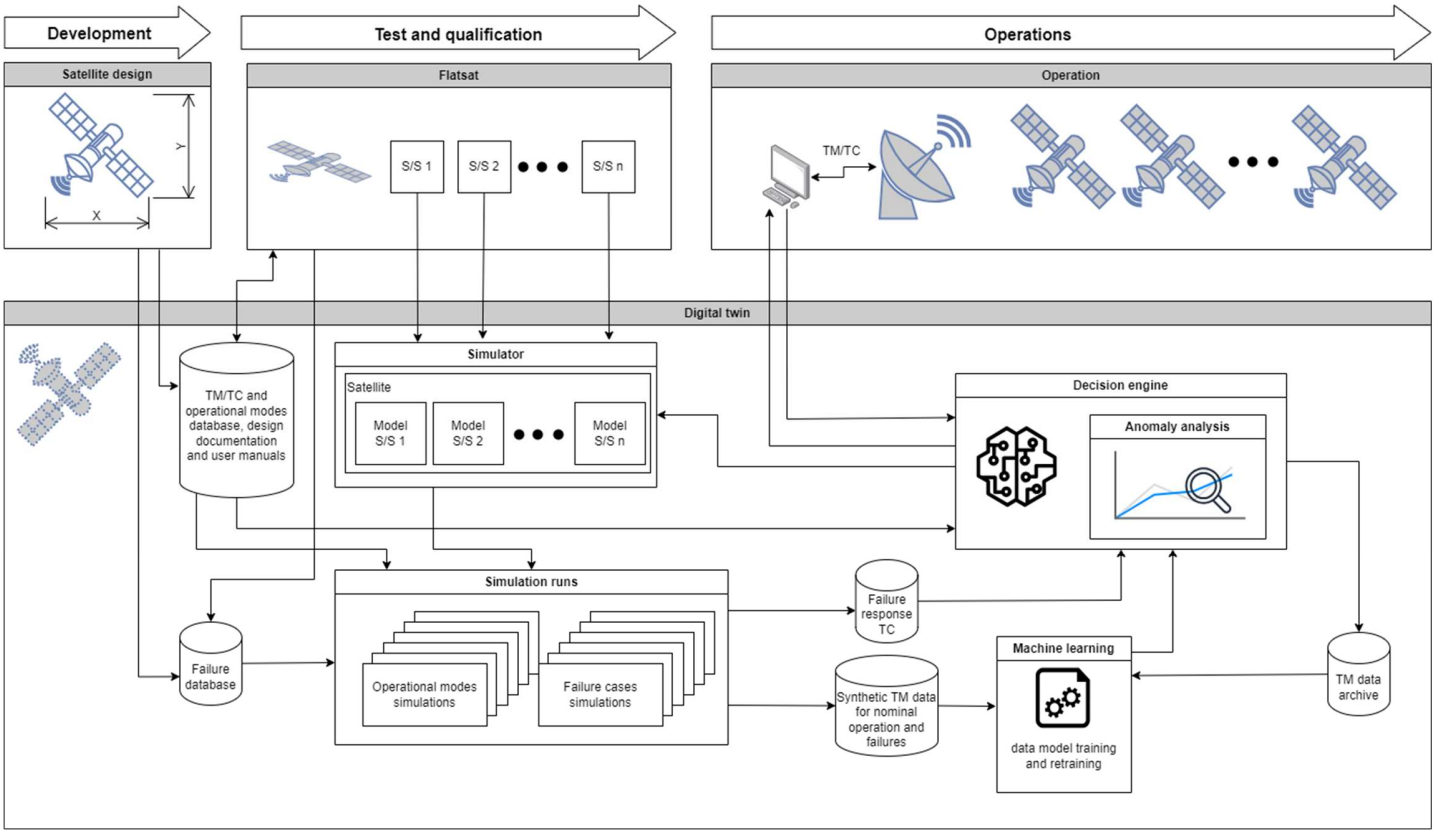

Fig. 11: Proposed design process including a digital twin

In the design process data about the satellite is collected, especially regarding operational modes, commands and planned telemetry data and also, information about possible failure cases. Furthermore, a satellite simulator has to be developed that is able to simulate the behaviour of the satellite with high accuracy and fidelity. The simulator consists of models for the various subsystems and payload of the satellite, the onboard software and a mission control system. The simulator is based on the develop flatsat of the integration and testing lifecycle phase. Since in this life cycle phase real hardware already exists and is tested, so especially the flight software is in a relatively mature state and also detailed information about the subsystems and payload is available. These are connected to environment simulation for orbit propagation and attitude of the satellite in space. Simulations can be executed using the simulator, the operational modes, telecommands and the possible failure cases, to investigate possible responses to the failure cases as well as to generate synthetic data to train machine learning models for anomaly detection.

Finally, in the operational phase, the real telemetry received from the satellites of the constellation can be analysed by the automated in the loop ML algorithm for detecting anomalies, which was trained with the synthetic data from the simulation runs. If no anomalies are found, the data is stored in a database and can be used to train the ML algorithms with real data gradually. If a anomaly is detected, and it can be assigned to a known failure using the methods in the description of the reasoning chain above, counter measures can be initiated.

The more identical satellites are used in the constellation, the more similar and consistent data can be collected to train the ML algorithms and to prepare for failures.

In this proposed process, the decision engine is the central element. In it all the inputs from the phases before, the real telemetry as well as the anomaly detection is brought together. To build such a decision engine, artificial intelligence can be the missing piece, especially with the emergence and development of Large Language Models (LLM). LLMs

offer a novel approach to decision-making within the satellite anomaly management process. By leveraging the full context provided by the preceding stages—detection, classification, disambiguation, and root cause identification—LLMs can synthesize rich, multi-layered inputs to inform downstream actions. These agents can be fine-tuned or prompted using extensive documentation, operator handbooks, and subsystem-specific procedures, giving them access to the same operational knowledge traditionally held by expert engineers. Once an anomaly has been contextualized, the LLM-based agent can reason over the telemetry profile, compare it with historical failures or documented mitigation strategies, and propose context-appropriate actions such as mode changes, reconfiguration commands, or escalation paths. In this setting, LLMs serve not only as interpreters of complex system behavior but as decision-support agents capable of bridging data-driven insights with operational expertise—greatly enhancing responsiveness and consistency in satellite operations.

5.3 Integration of Digital Twins

The above illustrated and described design process can be transferred into a DT or a set of DTs in the operational phase, see Fig. 12. Each satellite of a constellation is mirrored into its own DT on ground. These individual DT are connected to the other elements of the ground segment, such as the Mission Control System, Mission Planning System and Flight Dynamics System, via a microservice architecture. An additional DT can be introduced for the entire constellation of satellite. This DT can be used to monitor the overall status of the constellation and to control the Key Performance Indicators (KPI) of the constellation. One use case for the constellation DT is, if one of the satellites of the constellation has a problem and cannot contribute to the objective or service of the constellation, to perform simulations with current data of all satellites and to provide solutions how the constellation could still fulfil its KPIs.

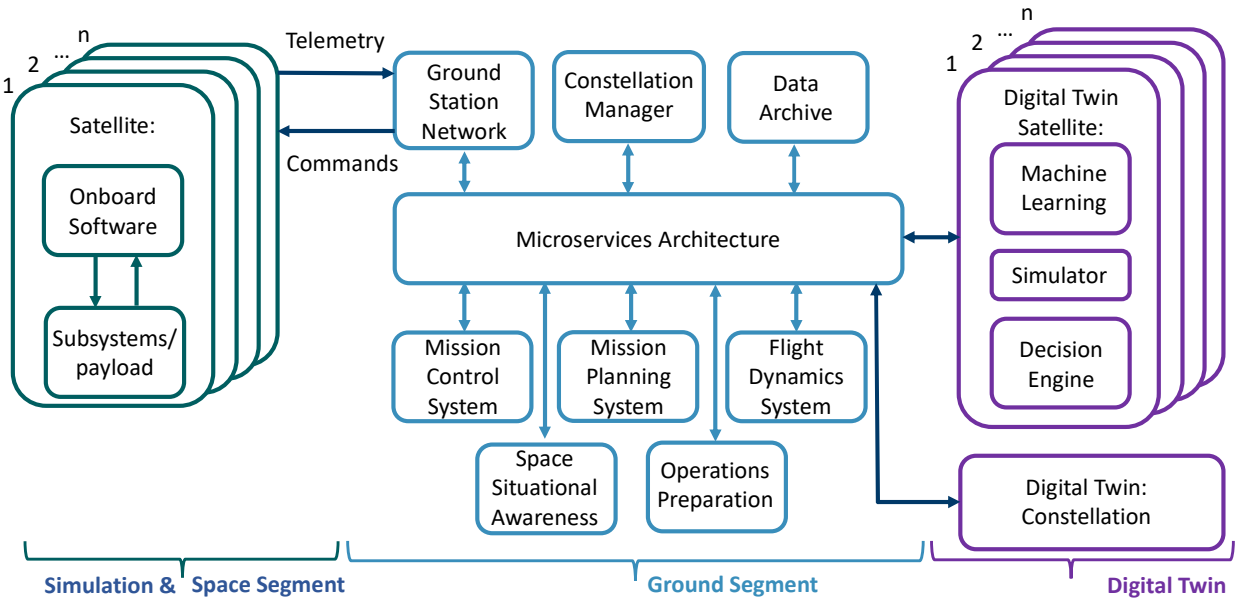

Fig. 12: Set-up of different digital twins in satellite operations for a constellation

6. How to test a Digital Twin for large Satellite Constellation?

To be able to test a DT within a satellite constellation a simulation environment is developed in which a single satellite simulator is scaled to a constellation of satellites [21]. The environment is based on a Kubernetes architecture offers a powerful platform for simulating and analyzing satellite constellations. Kubernetes is an open-source container orchestration system for automating software deployment, scaling, and management. The developed environment transforms a single entity of the NASA Operational Simulator for Small Satellite (NOS3) [22] into a flexible, scalable platform capable of modeling complex multi-satellite systems. The architecture is highly scalable, supporting simulations of large constellations, with the ability to model various satellite configurations, orbital parameters, and mission scenarios. It uses Kubernetes for efficient resource management and fault tolerance, enhancing the reliability and efficiency of simulations. The platform includes tools for performance monitoring and debugging, as well as a centralized interface for managing and monitoring satellite constellations.

Key features include modular, self-contained satellite representations enabling flexible scaling, a centralized control and monitoring interface that can be used with either OpenC3 COSMOS or YAMCS, and deployment

versatility across in-house private servers or cloud-based infrastructures. This system helps optimize the performance of satellite operations, supports operator training, and accelerates the development of new space systems. Additionally, it enables integration of advanced features like AI/ML-driven analysis and visualization techniques. Performance optimizations, such as a 40% reduction in CPU usage per simulated satellite compared to traditional VM-based methods, allow for a 65% increase in simultaneous satellite simulations on identical hardware. The system also supports adaptive fidelity, dynamically balancing simulation accuracy with computational demands. This scalable, adaptable platform not only streamlines pre-deployment testing and operational scenario development but also establishes a robust foundation for advanced capabilities, such as integrating DT technology to evaluate constellation-level performance metrics, which is the focus of our current research efforts. The architecture is designed for future cloud-based deployment, expanding beyond local resources and supporting distributed simulations. Future developments aim to integrate DT technology for modeling entire constellations and evaluating performance metrics, advancing the design, testing, and management of complex satellite constellations. the Integration of the DT concept into satellite constellation simulation aims to model an entire network as a unified system. The proposed constellation-level DT will simulate key performance indicators (KPIs) such as global coverage, network throughput, latency profiles, resilience, resource utilization, orbital stability, and collision avoidance maneuvers. This system will enable advanced capabilities like predictive performance optimization, what-if scenario analysis, capacity planning, anomaly detection, and enhanced operator training. Shifting from component-level to system-level simulation, it addresses the increasing complexity of satellite constellations and provides insights for strategic decision-making, long-term planning, and real-time optimization. Despite challenges in data aggregation, system modeling, and high-performance computing, this approach promises significant improvements in constellation management, performance, and reliability, making it a promising avenue for future development.

This Kubernetes setup provides us with a robust and flexible foundation for our constellation simulation, allowing us to efficiently manage and scale multiple satellite simulations within a single cluster. It offers improved resource utilization, easier management of complex configurations, and a pathway for future enhancements as our simulation needs grow.

We use Built-in Kubernetes objects for containerized applications to manage the main components of each simulated satellite, such as the NOS Engine, 42 simulator, and cFS instances. These Deployments allow us to easily scale the number of simulated satellites and manage updates seamlessly.

To facilitate communication between components and expose necessary endpoints, we utilize Kubernetes Services. We create a Service for each major component (NOS Engine, 42, cFS, and COSMOS) of every simulated satellite, allowing for standardized internal networking. Our modular design treats each satellite in the constellation as an independent unit, consisting of a set of core components. This approach offers both scalability and the ability to operate each simulated satellite autonomously. The modularity is especially beneficial when simulating constellations with diverse satellite configurations or when zooming in on specific subsets for in-depth analysis. By adopting Kubernetes, we are well-positioned for future growth: the core system components are already optimized for containerized deployment, allowing us to focus on refining orchestration and resource management. Additionally, our Kubernetes-based architecture provides substantial flexibility, making it easier to accommodate future expansions and changes.

Fig. 13 illustrate a schematic representation of the Scaling of NOS3 Instances. Scaling NOS3 to simulate multiple satellites introduces significant computational demands that require careful consideration. Our current architecture addresses these challenges through a focused approach to performance optimization, tailored to our single server environment.

Scaling satellite simulations presents several technical challenges that must be addressed for effective and efficient operation. Key challenges include port conflicts, where identical simulator instances may collide on default ports, and hostname conflicts, requiring unique identifiers for each container or network alias. Configuration management becomes increasingly difficult as the number of satellites grows, necessitating the use of parameterization and automated configuration generation rather than manual editing of XML files. Resource limitations, including CPU, RAM, and file descriptors, quickly arise on a single host, which can hamper scalability. Additionally, handling multi-target capability for the ground system (GSW), particularly when managing multiple spacecraft simultaneously, demands unique interface configurations**.** Time synchronization across independent satellite simulations also poses difficulties, especially when accurate time references are required for constellation analysis. Lastly, debugging and logging become exponentially more complex as the number of containers increases, making it challenging to manage and correlate logs effectively. Addressing these issues is crucial for developing a scalable and manageable simulation architecture.

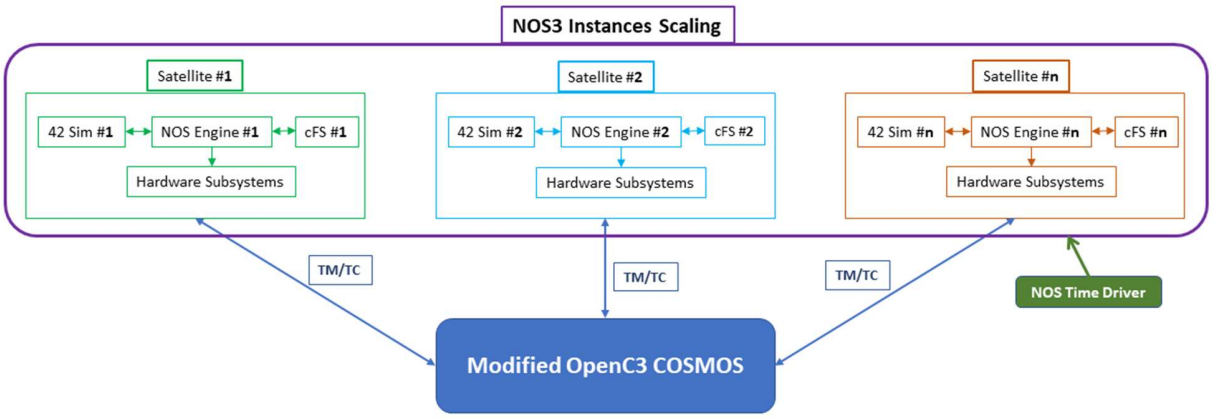

Fig. 13: Scaling of NOS3 within the Kubernetes-based simulation environment

7. Conclusion

In this paper we give an overview over certain aspects for the use of digital twins for operating large satellite constellations and how they may be used to operate such constellations. The influence of automation on the scaling of large satellite constellations is described as motivation to use digital twins. Derived from this a reasoning chain is introduced to show the different steps that need to be performed in order to make decisions when dealing with anomalies onboard a satellite. It is shown that this is a very complex task with various steps that requires a high degree of intelligence. This reasoning chain is the starting point for the definition of a design process of digital twins for satellite operations that is already used in the design process of a satellite (constellation) and during integration and testing phase to collect data about the satellite, possible failure cases, real telemetry data from flatsats and synthetic data from simulations. Finally, we present an architecture for testing the proposed digital twin set-up for large satellite constellations by scaling a satellite simulator using a state-of-the-art software approach.

References

[1] D. Shangguan, L. Chen and J. Ding, “A Digital Twin-Based Approach for the Fault Diagnosis and Health Monitoring of a Complex Satellite System”, Symmetry, Vol. 12, No. 8, 2020
[2] M. Grieves, Digital Twin: Manufacturing Excellence through Virtual Factory Replication, 2015
[3] A. Fuller, Z. Fan, C. Day and C. Barlow, Digital Twin: Enabling Technologies, Challenges and Open Research, in *IEEE Access*, vol. 8, pp. 108952-108971, 2020, doi: 10.1109/ACCESS.2020.2998358.
[4] M. M. Rathore, S. A. Shah, D. Shukla, E. Bentafat and S. Bakiras, "The Role of AI, Machine Learning, and Big Data in Digital Twinning: A Systematic Literature Review, Challenges, and Opportunities," in *IEEE Access*, vol. 9, pp. 32030-32052, 2021, doi: 10.1109/ACCESS.2021.3060863.]
[5] J.R. Cook; A. Johnson, G. Hautaluoma, NASA Readies Perseverance Mars Rover’s Earthly Twin. 4 September 2020, https://www.nasa.gov/feature/jpl/nasa-readies-perseverance-mars-rovers-earthly-twin (accessed 11.02.2025)
[6] B D. Allen, Digital Twins and Living Models at NASA, 3 November 2021, https://ntrs.nasa.gov/citations/20210023699 (accessed 17.03.2025)
[7] J.-R. Cook , D.C. Agle, A. Johnson, G. Hautaluoma, Why does the world (and NASA) need digital twins? 4 September 2020, https://www.nasa.gov/centers-and-facilities/jpl/nasa-readies-perseverance-mars-rovers-earthly-twin/ (accessed 31.01.2025)
[8] R. Minerva, N. Crespi, R. Farahbakhsh, F. M. Awan, Articial Intelligence and the Digital Twin: An Essential Combination,in: N. Crespi, A. T. Drobot, R. Minerva (Eds.), The Digital Twin, Springer Nature Switzerland AG, Cham, 2023, pp. 299-336
[9] L. Pizzuto and S. W. Ziegler, Impact of Launch Cadence on the Automation & Economics of Constellation Operations, 75th International Astronautical Congress (IAC), Milano, Italy, 14-18 October 2024
[10] K. Kotowski et al., “European Space Agency Benchmark for Anomaly Detection in Satellite Telemetry” arXiv: 2406.17826, 2024.
[11] Schmidl, S., Wenig, P., & Papenbrock, T. (2022). Anomaly detection in time series: A comprehensive evaluation. Proceedings of the VLDB Endowment, 15(9), 1779–1797.

[12] R. Chalapathy and S. Chawla, "Deep Learning for Anomaly Detection: A Survey," arXiv preprint arXiv:1901.03407, 2019.
[13] Z. Zamanzadeh Darban, G.I. Webb, S. Pan, C. Aggarwal, M. Salehi, Deep learning for time series anomaly detection: A survey, ACM Comput. Surv. 57 (1) (2024) http://dx.doi.org/10.1145/3691338.
[14] Y. Hundman et al., "Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding," KDD, 2018.
[15] H. Zhou et al., "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," AAAI, 2021.
[16] B. Wu et al., "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis," ICLR, 2023.
[17] D. An and S. Cho, "Variational Autoencoder based Anomaly Detection using Reconstruction Probability," Special Lecture on IE, 2015.
[18] L. Ruff et al., "Deep One-Class Classification," ICML, 2018.
[19] D. Li et al., "MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks," arXiv: 1901.04997, 2019
[20] Y. Jeong et al., "AnomalyBERT: Self-Supervised Transformer for Time Series Anomaly Detection using Data Degradation Scheme," arXiv preprint arXiv:2305.04468, 2023.
[21] M. Campanelli, Architecture of a Simulation Test Bench for Operating Large Satellite constellations, 75th International Astronautical Congress (IAC), Milano, Italy, 14-18 October 2024
[22] National Aeronautics and Space Administration, NASA Operational Simulation for Small Satellites, https://www.nasa.gov/nasa-operational-simulation-for-small-satellites/, (accessed 06.08.2024)