17th International Conference on Space Operations, Dubai, United Arab Emirates, 6 - 10 March 2023.

SpaceOps-2023, ID #278

A Concept to convert Operational Simulators to a Digital Twin by using AI Techniques

Pouya Haschemi[a]*, Joao Bernardes Guerreiro[a], Hamza Faiz[a], Callum S. Arnot[a]

[a] *Terma GmbH, Europaarkaden II, Bratustraße 7, 64293 Darmstadt, Germany*, poha@terma.com
* Corresponding Author

Abstract

Considerable effort has been invested in recent years in the area of artificial intelligence, as well as the concept of a digital twin (DT) for various areas within the space industry. For the area of space operations, this has included among others the development of DTs of all subsystems, visualising the holistic picture and eliminating the need for deep knowledge about each subsystem. However, DTs can only be applied in use case specific circumstances and must evolve over time to mature for other specific use cases. One such use case is the application of DTs in conjunction with operational simulators to enhance mission operations. The concept would be to establish a Spacecraft DT retrieving real-time telemetry (TM) and applying artificial intelligence (AI) and machine learning (ML) techniques in connection with operational simulation models to improve predictions, detect anomalies, and increase trust in simulation models with the long-term objective of fully eliminating the need for physical engineering models. In the long term this concept would allow the introduction of an Artificial Operator automatically performing different operational activities such as collision avoidance, manoeuvre calculation, budget calculation, scheduling, etc. The artificial operator would be driven by the AI enabled DT which is in turn being fed by the operational simulator and seamlessly supervised by the human operator.

This paper presents a concept for a modular digital twin framework with a spacecraft operator as the central user. This includes an EGS-CC architecture being connected to both a real spacecraft (S/C) as well as its SIMULUS based Operational Simulator including all required simulation components. The concept presents the method with which real and simulated TM packets are being collected, stored, and further processed for an AI/ML enabled DT post-processing. The AI/ML models shall enable the detection of possible anomalies and failures by making automated use of the operational simulator and by this also allow a better on-board budget prediction. Based on this, a GUI indicates the actual state of the S/C which in turn is a digital representation of the physical S/C in real time. In the long run, it is foreseen to apply the AI/ML models for correction of dedicated simulator models by changing model configurations to enhance the fidelity and therefore realism of the simulator models.

Keywords: Digital Twin, Artificial Intelligence, Operational Simulator, Mission Control System

Acronyms/Abbreviations

Artificial Intelligence (AI), Assembly Integration and Testing (AIT), Application Process Interface (API), Archive (ARC), Analysis and Reporting System (ARES), Consultative Committee for Space Data Systems (CCSDS), Command Link Transfer Unit (CLTU), Command Packet Handling (CPH), Data Generation Module (DGM), Digital Twin (DT), EGOS Data Distribution System (EDDS), European Ground Operation System – Common Core (EGOS-CC), European Ground Systems - Common Core (EGS-CC), Fault-Detection -Isolation and -Recovery Techniques (FDIR), Forward Space Packets (FSP), Graphical User Interface (GUI), Ground Station (G/S), Human-Machine-Interface (HMI), Machine Learning (ML), Monitoring and Control (M&C), Monitoring and Control Model (MCM), Mission Control System (MCS), Mission Control System – Common Core (MCS-CC), Mission Planning System (MPS), Natural Language Processing (NLP), On-board Software (OBSW), Processed Data Archive (PDA), Perception Module (PerM), Parameter Extraction (PEX), Processing Module (ProM), Packet Utilization Standard (PUS), Return All Frames (RAF), Return Channel Frames (RCF), Return Operational Control Fields (ROCF), Spacecraft (S/C), Spacecraft Control and Operation System 2000 (SCOS-2000) (S2K), Source Data Archive (SDA), Space Link Extension (SLE), TC Data Link Protocol (TCD), Transmission Control Protocol / Internet Protocol (TCP/IP), Telecommand (TC), Telemetry (TM)

1 Introduction

This paper defines a concept for an infrastructure which connects different building blocks, specifically the Mission Monitoring & Control infrastructure based on EGS-CC (M&C), ESA's Mission Planning System (MPS), the ARES-based mission lifetime data archive (connected via EDDS with the EGS-CC infrastructure), a SIMULUS based Satellite Simulator (for a current mission), in combination with artificial intelligence/machine learning models which are themselves another core of this paper.

The infrastructure aims to ensure the archiving of operational data, data retrieval (e.g. operational data, simulated data, test data) as well as a logical data exchange between the building blocks. The infrastructure will then form the structure of a spacecraft digital twin (DT), including several interfaces to allow smooth data flow between systems, which can be applied to customise on various DT use cases (e.g. anomaly detection, failure diagnosis, etc.).

As previously mentioned, the last building block (AI/ML Models) is another central aspect of this paper. The notion of AI is a broad concept encompassing the idea of machines being able to perform tasks which require human-like intelligence. One specific approach to achieving AI is called machine learning (ML), which focuses on training machines with a data-driven approach to learn parameters – often referred to as model training.

Depending on its training data and end goal, these models can perform certain tasks such as predicting future numerical values or classifying data. In the present case this data-driven approach shall leverage TC/TM data from the real asset (spacecraft in orbit) and a parallel-running operational simulator. To use this for the ML model, there shall be a data pre-processing step, typical of ML pipelines. This step will look to clean and transform both the real and simulated data. From here a machine learning model is built and shall consume the necessary data for training, testing, and validation. After this stage, the model shall be ready for deployment in a digital twin scenario.

In the following sections the specific architecture elements such as the Digital Twin, Artificial Intelligence, and the relevant Ground Segment Systems are described.

1.1 Digital Twin

The term "digital twin" embodies a new paradigm. DTs are connected to a real-world counterpart and use various technologies and other solutions such as simulation, artificial intelligence, and augmented reality to optimise the different processes with respect to their counterparts. DTs are being addressed at an accelerating pace in both industry and academia, and have therefore reached multiple degrees of acknowledgement. The term "digital twin" has evolved since its first use in 2003 by the University of Michigan as well as later by Grieves who proposed that a digital twin consists of three parts: physical product, virtual product, and their connections [1]. In 2012, NASA defined it as "a multi-physics, multiscale, probabilistic, ultra-fidelity simulation that reflects, in a timely manner, the state of a corresponding twin based on the historical data, real-time sensor data, and physical model"[2].

Digital twin solutions are distributed among different sectors. Most solutions currently appear within the Architecture, Engineering, and Construction (AEC) industry, followed by the manufacturing industry and the urban sector. The Aerospace & Defense sector is currently covered at a very basic level. Especially within the space sector, there is still no existing solution or solution provider active.

To be able to build an interoperable digital twin, a ground laying "digital thread" first needs to be established (see Figure 1). The digital thread is a method to generate a central access point for data, a so-called single source of truth. For example, when rolled out at an enterprise level, it forms consistency and raises collaboration by streamlining different functionalities and features around a set of data. It connects data from different tools and software such as CAD, PLM, ERP, CRM, and so on, allowing users to create, maintain, and exchange data while deploying and communicating processes. It also allows analysis and verification of increasingly complex systems across the customer/supplier chain, across disciplines, and throughout phases of development more effectively. In short, the digital thread is intended to improve the interoperability of all used state of the art and future engineering tools. The concept is therefore highly applicable to the creation of a spacecraft digital twin.

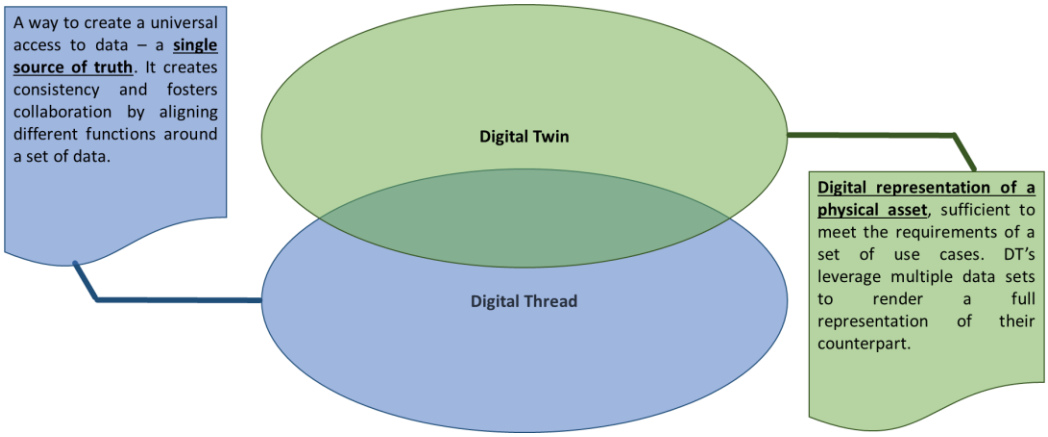

Figure 1 From Digital Thread to Digital Twin

1.2 Artificial Intelligence

AI is a broad field of study involving the development of systems for performing intelligent tasks such as understanding natural language, recognising images and objects, and making decisions. This is achieved using various techniques, such as rule-based systems, optimisers, decision trees, and neural networks. AI is currently being used in a wide range of applications, such as self-driving cars, image and speech recognition, and natural language processing. The following subsections discuss two key approaches to the development of AI systems: Machine Learning and Deep Learning.

1.2.1 Machine Learning

Within the field of AI, Machine Learning (ML) is a particularly interesting application to explore for the work proposed in this paper, as it can leverage the multiple data sources envisaged for the digital twin infrastructure. ML models can be used to analyse data and make predictions or decisions based on that data which allows computers to improve their performance on a task over time, without the need for human intervention.

The generic ML pipeline usually consists of the blocks illustrated in Figure 2.

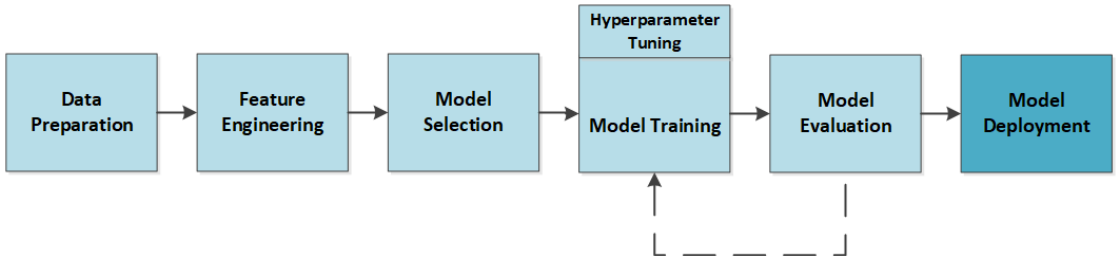

Figure 2 ML pipeline

The key blocks are described as follows.

- **Data collection and preparation**: here the necessary data for the project is gathered, cleaned, and pre-processed to make it usable for training ML models. Depending on the size of the data, it may be necessary to use specially selected processing and storage systems.

- **Feature Engineering**: this step is where data scientists extract useful features from the data to be used as inputs for the model. This is a very iterative process and a large fraction of the AI model development cycle.

- **Model Selection**: here an appropriate model architecture for the task is chosen. Again, depending on the dataset size and the envisaged application and its deployment environment, it may be necessary to use specific deep learning frameworks.

- **Model Training**: this typically involves feeding the data through the model, adjusting the model's hyperparameters to minimise errors, and repeating the process until the model is able to perform according to a standard set by the engineers.

- **Model Evaluation**: here a separate set of data (called the test set) is used to evaluate the model's performance over a particular set of metrics relevant to the domain of the application. This step is important to ensure that the model is performing appropriately on an unseen dataset and is not overfitting to the training data.

- **Model Deployment**: this is the deployment of the AI model in a production environment where it will be monitored and potentially retrained to deal with data drifts.

1.2.2 Deep Learning

Deep learning (DL) is itself a subfield of ML involving the use of Artificial Neural Networks (ANN) with several layers (hence “deep” learning). ANNs are simply a type of machine learning model inspired by the structure and function of the human brain. It consists of layers of interconnected "neurons” which process and transmit information. Each neuron in a layer receives input from the previous layer, processes it, and sends it to the next layer. Adding more layers allows the model to learn more complex representations of the data which can have applications which were previously difficult or impossible for traditional machine learning. Figure 3 shows a schematic of a simple ANN.

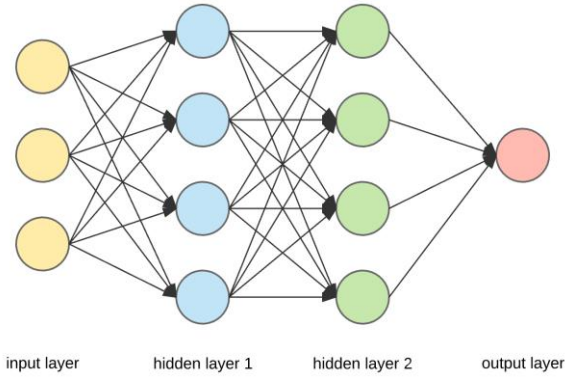

Figure 3 Deep Neural Network diagram

In simple terms, AI is a broad concept encompassing the idea of machines being able to perform tasks that require human-like intelligence, while ML is a specific approach to achieving AI, which focuses on training machines to learn from data. DL is a subfield of ML which alludes to the usage of multi-layered ANNs (also often called Deep Neural Networks) to perform highly complex tasks.

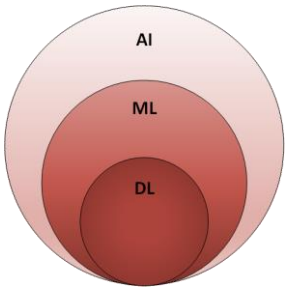

Figure 4 Illustration of AI, ML and DL relational concepts

1.3 Ground Systems

The DT concept proposed by this paper relies on the use of two main ground segment systems, specifically a spacecraft operational simulator and an EGOS-CC based mission control system. This section will describe these systems in a general sense, while the approach for their inclusion in the DT will be described in later sections.

1.3.1 Operational Simulator

Spacecraft operational simulators are used by the mission's flight control team to create, refine, and practice real flight operations procedures both preceding and during the mission lifetime. An operational simulator is a computer program which models the behaviour of all spacecraft subsystems for a specific spacecraft or mission. It emulates the processor of the on-board computer (OBC) and internally runs a copy of the real spacecraft's on-board software (OBSW). It accepts all telecommands (TC) and generates all relevant telemetry (TM) used for and produced by the real spacecraft. It models the physical environment around the spacecraft to accurately reproduce the behaviour of the affected subsystems, simulates the ground stations which track the spacecraft, and allows for the user-injected failure of spacecraft subsystems to simulate real component failures. In addition, it interfaces with the mission control system in the same way as the real ground stations, so spacecraft operators can interact with it in the same manner as for the real spacecraft.

In this paper, operational simulators are primarily discussed assuming their development using ESA's SIMULUS infrastructure, which consists of a run-time framework (SIMSAT), software emulators, dynamics models (environmental and body), reusable generic models, and ground station models. SIMULUS provides an extensive toolkit for the development of spacecraft simulators, allowing for shortened development times and greater reuse of existing models when compared to writing a simulator from scratch.

As mentioned in Section 1, an operational simulator will be used as a key component in the digital twin structure proposed in this paper and will be discussed in more detail in later sections.

1.3.2 EGOS-CC based Mission Control System

MCS-CC is a Mission Control System which extends the EGS-CC with additional EGOS-CC components supporting mission operations functions needed by all ESOC missions but not covered within the scope of EGS-CC.

The objective for the adoption of EGS-CC is to enable the development of a new generation of ESOC's ground data systems infrastructure supporting all applications related to mission and network operations preparation and execution. The ambition is to deploy EGS-CC based solutions for all types of monitoring and control applications, for all categories of missions and for all types of controlled systems, including the space segment as well as ground equipment, in particular the network of ground stations. This is necessary to tackle the obsolescence of the current infrastructure and offers an opportunity to re-think some of the current processes to pursue important strategic objectives, including:

- Leveraging on the EGS-CC as a European level initiative to minimise Cost of Ownership of the next generation M&C Ground Data Systems Infrastructure.
- Enabling long term reduction of development and maintenance costs of the ground data systems infrastructure and of the dedicated systems relying on it.
- Providing the users communities with an efficient environment to prepare and execute operations using modern technologies.
- Rationalising the organisation/architectures of the target systems to enable a clean split of responsibilities throughout the lifecycle.
- Promoting/enabling cross-fertilisation of concepts/solutions with other European EGS-CC stakeholders.
- Promoting/enabling cross-fertilisation of concepts/solutions between the Missions and G/S Network operations domains. The new generation of EGS-CC based infrastructure products is expected to provide the development and user communities with ancillary opportunities and benefits which go beyond the capabilities of the current implementations, which include:
    - Support of heterogeneous controlled systems: the S2K infrastructure has been designed assuming a given category of controlled systems, namely spacecraft exchanging TM/TC data with the monitoring and control systems on the basis of the ECSS Packet Utilisation Standard. The EGS-CC based infrastructure will provide an abstraction layer which will enable the design and execution of operations to act on any type of systems, including in particular the space and the ground segment systems involved in mission operations.

- Simplified integration of multiple controlled systems: this applies to the complete process to tailor, validate and deploy monitoring and control applications for 'system of systems', such as a space segment composite consisting of several units assembled together, a network of ground stations, a control centre consisting of mission dedicated or shared software applications, a family of missions managed by the same operations team, a constellation consisting of multiple spacecraft.
- Enable off-the-shelf re-use of ground data systems infrastructure products: the EGS-CC has been designed to respect a component and interface-based approach, enabling the development and seamless integration of functional extensions. For the new generation infrastructure, the same principles and architectural layering of the EGS-CC itself will be adopted, so that a clear separation between the infrastructure and the application specific implementations is enforced. Full solutions can be designed, implemented and deployed without any detailed knowledge and, most important, need of modification of the underlying implementation (binary compatibility).
- Simplification of operations concepts: current implementations heavily rely on the definition and execution of operations at a very low level (e.g. individual telecommands, individual telemetry parameters). The EGS-CC provides an abstraction layer which enables a radical simplification of the interactions with the controlled system and promotes wherever possible the design of operations which can be relatively easily automated, thus leading to operations execution costs reductions.
- Cross-fertilisation across heterogeneous teams: currently largely different approaches are followed by the space segment and ground stations network operations teams, for example different tools are used in order to prepare and validate the necessary artefacts (e.g. M&C data, procedures, displays) but also substantially different lifecycles are followed to implement the operations planning and execution processes. The new generation EGS-CC based infrastructure will promote and somehow enforce the rationalisation and harmonisation of the associated processes e.g. through the adoption of a common data model to manage the applications tailoring for a given controlled system [10].

2 Methodology

This section describes the overall approach to achieve the objectives of the concepts addressed within the paper. It includes aspects related to the EGS-CC infrastructure, AI and ML techniques, as well as the approach behind relevant use case selection.

2.1 High-Level System Structure

The general concept presented within this paper is to combine existing solutions to create a digital twin of a real spacecraft. This includes various Mission Operation software and tools as well as AI/ML techniques, with an operational simulator and EGS-CC based MCS included in the loop. The combination of these systems is a novel approach and requires further analysis as well as addressing the challenges coming with each of these systems, including their integration. The following sections will highlight and further discuss these different components.

2.2 Deploying the EGS-CC Infrastructure

The AI needs to access data from the real and simulated spacecraft, which implies that the MCS-CC shall manage different types of TC/TM data flows. This could be achieved by proper deployment of the EGS-CC infrastructure, given the solutions offered by the concept of EGS-CC sessions. This problem is further elaborated with possible solutions in the following sections.

2.2.1 Problem

For demonstrating the concept of an operational simulator digital twin using AI, the spacecraft monitoring and control capabilities would be provided by MCS-CC as an extension of the EGS-CC infrastructure. MCS-CC would need to establish TCP/IP communications via the CCSDS SLE interface to the real and simulated G/S, then exchange TC/TM data with the real and simulated S/C simultaneously. On the uplink, the same TC shall be released to both real and simulated G/S and S/C, then archived into two distinct dataspaces so that TC verification stages get updated upon receiving corresponding real or simulated TM. On the downlink, real or simulated shall also be archived in separate data spaces TM to allow the digital AI twin to query for the data of interest. This requires an adequate MCS-CC configuration and deployment.

Part of the overall mission ground segment, the MCS could expose several interfaces to external systems for exchanging TC/TM data, such as Automation, Flight Dynamics, Data Dissemination, Mission Planning, Ground Station, and Simulator. Part of this research paper, for direct M&C performed by MCS-CC we will elaborate on the interfaces with the Ground Station and the Simulator. For external systems with no direct access the MCS-CC and with a need for remote M&C, we will elaborate on the example of interface between MCS-CC and MPS for scheduled commanding and will elaborate on the example of interface between MCS-CC and EDDS/ARES for monitoring. For M&C a human would typically operates those systems software using a GUI. The AI digital twin would substitute the human operator and use exposed APIs.

2.2.2 Real operations and simulated system sessions

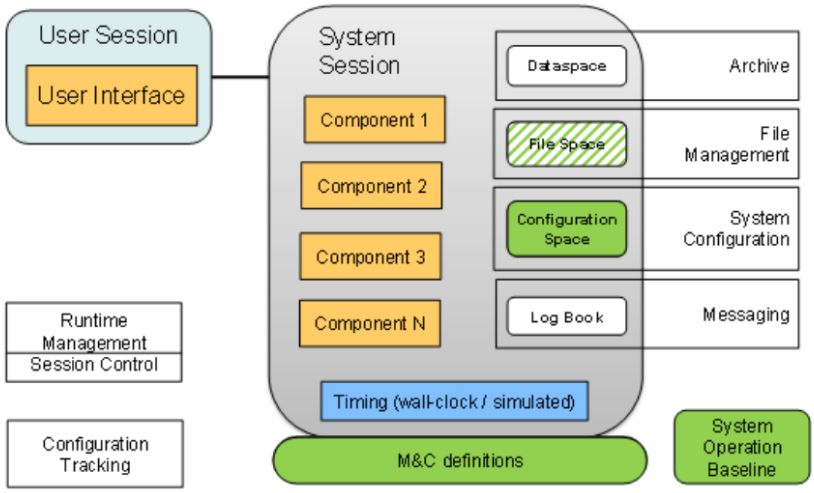

Figure 5 – EGS-CC System Session [11]

An MCS-CC System Instance is a collection of EGS-CC components across one or more physical and/or virtual machines to make up a context for the system. A system session is defined as a logical grouping of EG(O)S-CC components belonging to a given running system instance which use a given configuration and tailoring, are dedicated to the processing of the data of a monitored system in a common processing context and generate a separate set of outputs (including archive dataspaces). The selection of the components is achieved by defining application instances. Most EGS-CC applications run within the context of a system session. Some applications in a system instance run outside a system session.

A system session allows a collection of components to be run in a self-contained unit within a system instance that will have no impact or will not be impacted by other system sessions. System sessions can be started, terminated, and restarted (resumed) within a running system instance. A system session can be seen as a self-contained run of the system where all data being processed and persisted is held completely separately from that used by other system sessions. System sessions can run in parallel and use different simulated times from one another.

There are different types of system sessions, e.g., operational, simulations, tests, preparation, evaluation, and administration. They might correspond to a test or test campaign in AIT, or to a period or phase of operations. Separate system sessions might be created for preparation or configuration activities [11].

As later shown in Figure 8, to support TC/TM data exchange with real or simulated G/S and S/C, the deployment of two distinct System Sessions within the same MCS-CC system instance is foreseen, a Real Operations System Session and a Simulations System Session, each using the CCSDS SLE interface and using specific TC/TM data flows, as described in Section 4.2.3

2.3 AI component

By providing an infrastructure for both real and simulated data pipelines, the digital twin infrastructure opens a broad range of possibilities for data-driven applications. The following section shall focus on establishing the foundations of this building block which will be common across many of the envisaged applications.

2.3.1 Data preparation

The first step in building AI/ML models, as mentioned in Section 1.1is to prepare the data. The envisaged AI/ML models will be able to access different data sources:

- Real TM data from the spacecraft
- Simulated TM data from the simulator
- The Telecommands (TC) history
- Data archives from AIT campaigns

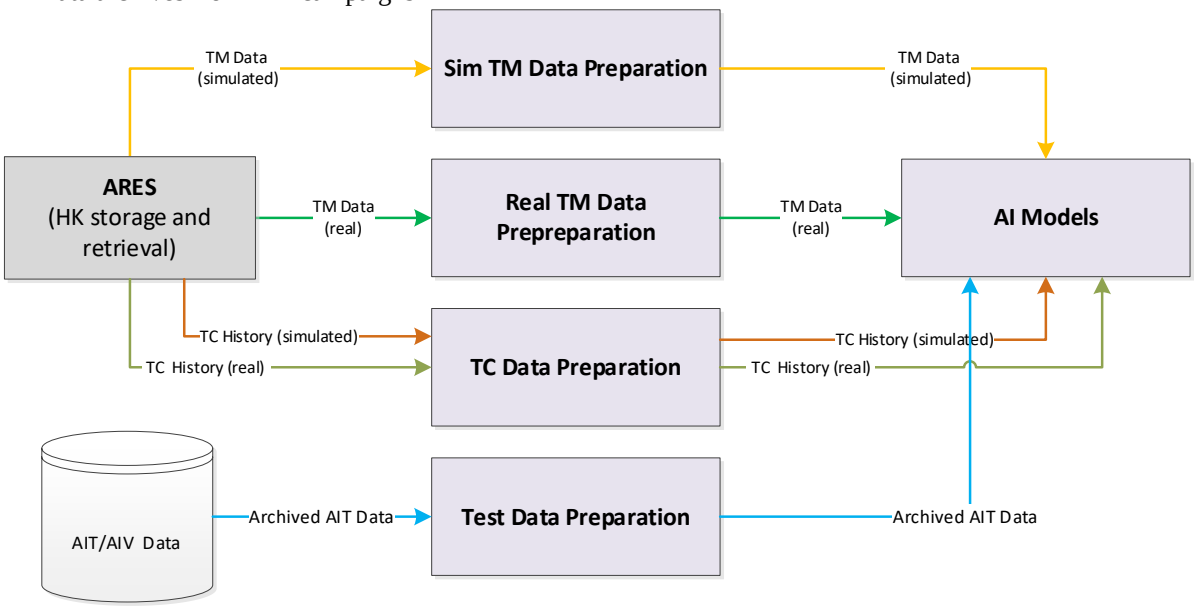

Figure 6 Data preparation

This step is heavily reliant on the use case and data utilised, as different approaches require different methods to handle the data. Domain knowledge can play a significant role in the data cleaning aspect of pre-processing. Some relevant domain factors could be:

- Out of limits (OOL) checks and ranges to consider for TM values.
- The limitations on the simulator and its TM values.
- The fact that the TM data is time sensitive, meaning that values within a certain time window may not be representative of the use case.
- That some TCs or TC sequences may not be relevant for the use case.
- That satellite TM data can contain missing values because of the difference in sampling periods and timings among their variables.
- That TM data can correspond to different spacecraft operational modes and therefore not be relevant for the use case at hand.
- That often TM outliers are caused by glitches in data conversion or transmission and (for an anomaly detection use case) are not of interest to the operator [3].

It is important to mention that with more domain knowledge in the pre-processing step and a more curated the dataset, higher quality data can be fed to the AI models leading to better performance. This also means that simpler AI models can be used which will contribute to more explicable results. This approach is typically more granular and often considers one parameter at a time - a univariate approach.

In contrast, a multivariate approach accounts for several parameters at once. This results in a much more complex and resource-heavy model requiring more computational resources during training. Having said that, by accounting for multiple variables at once these models are able to extract contextual insights between parameters and potentially find relationships which were otherwise undetectable. A potential use case could be to raise contextual anomalies and therefore increase operator awareness ([4]-[6]).

Finally, with the addition of the digital twin infrastructure and the access to the simulated TM data as well as the TC history, multivariate approaches could potentially include more significant features and insights and continuously test them in a simulator, as we shall elaborate on the following sections.

2.3.2 AI/ML Models

As in the previous section, the AI models are highly dependent on the use case scenario of the envisaged application. In principle, the digital twin is model agnostic, meaning it can use any kind of AI model. An additional case would be to use an ensemble of different AI models all contributing to the same use case. With this in mind, we can elaborate on some models:

Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) networks are a type of Recurrent Neural Network (RNN) architecture that is capable of remembering previous inputs for a longer period of time. They use a combination of gates (input, forget, and output gates) to control the flow of information and preserve the memory of previous inputs over multiple time steps. LSTMs can learn the normal patterns in the time series data by processing the sequential data over multiple time steps and maintaining a hidden state that encodes the information about the past inputs. This allows the LSTM to capture long-term dependencies in the data, which is useful for time series data such as TM.

LSTMs can be fed on nominal satellite data and then predict TM values using either univariate or multivariate approaches [7].

Auto Encoders

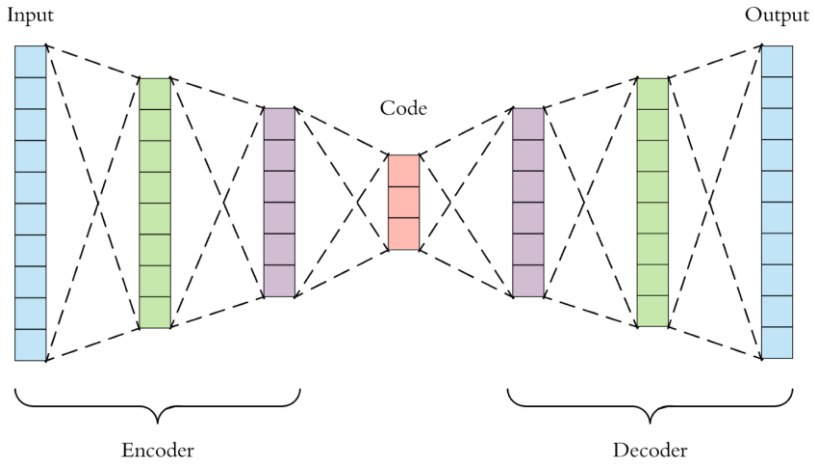

Figure 7 Autoencoder architecture [8]

TM data suffers from the “curse of dimensionality” where essentially its large number of features makes the amount of data required to accurately model the problem increase exponentially. The Autoencoder architecture consists of two main components: an encoder network which takes input and compresses it into a bottleneck to find latent (“hidden”) dimensions, and a decoder network which reconstructs it back to its original dimensions. By minimising the difference between its input and its reconstructed output the network can find a lower dimensional space which is representative of the data.

In practical terms, this means that the autoencoder should be capable of reconstruct nominal instances of satellite data with minimal error, but struggle to reconstruct instances that are unusual or anomalous. This can be used to identify potential anomalies similarly to the work performed in [4].

2.4 Use Case Selection

The use case identification will be considered from two angles:

- From the use case point of view: this aims at finding use cases first and asking afterwards what data would be required for the envisaged ideas.

- From the data availability point of view: this aims at first identifying the available data sets and then coming up with use cases that can be approached with the available data.

Therefore, use cases and data sets must coincide. A use case is only useful when data required is available for approaching it and a data set is only valuable when there is a useful use case associated to it.

The start will be with a broad but shallow perspective to acquire a birds-eye overview. Subsequently, use cases will be down selected and the level of detail will be increased. We apply this approach to achieve a broad coverage while still acquiring details as necessary. To maintain a good overview throughout these tasks and to improve the selection process, we apply a loosely systematic process. For this, we use an informal taxonomy for loosely judging the use cases and data sets. The aim is to provide light-weight support during these tasks without applying a too-rigid framework nor adding excessive overhead. Since the taxonomy follows a pragmatic solution-oriented approach and as we present preliminary results here, this taxonomy must be further adjusted before an actual usage. Below an overview of the preliminary taxonomy is given.

- Use Cases
  - Value
    - Reduce Manual Labor
    - Extend Mission Lifetime
    - Increase Science/Payload Operation Duration (e.g. through better margins refinement)
    - Prevent Failures
    - Improve Accuracy
    - …
  - Novelty
  - Application Fields
  - Perceived Maturity
- Data Sets
  - Access (IPR, Technical Data Access and Storage)
  - Quality
    - Missing/Duplicate Values
    - Gaps
    - Resolution
    - Time, Spatial
    - Accuracy
  - Suitability for AI/ML models
    - Data Set Size
    - Number of Relevant Occurrences

3 Use Case Scenarios

This section discusses some of the potential use cases for the proposed AI-based DT. This list is not extensive and shall only give some first ideas on possible application areas for the DT solution.

3.1 Novelty Detection

Novelty detection, also known as anomaly detection, allows the machine learning model to detect early either known or unknown types of anomalies, depending on type of model trained, supervised or unsupervised, respectively.

Within operations, the obvious case would be the detection of unusual behavior within telemetry. In the case of unsupervised trained models, unusual behavior seen in telemetry, representing behavior outside of the norm of, e.g., a satellite, could be a sign of an anomaly about to occur. Whereas in the case of supervised trained models, a system is taught to look out for specific types of anomalies, e.g., if there is a specific type of critical event you want to be notified of or want to have an automatic reaction from the system.

As a use case, novelty detection was one of the first applications of machine learning to be applied to the domain of space operations. However, although there are existing approaches, there have been many recent developments in the area over the last few years. Specially to aid the operators, focus on handling of false detections has a very high interest.

In addition to novelty detection as discussed, there is also another class of anomaly detection, based in contextual information. Some behaviors can only be considered anomalous in relation to other behaviors, which either means that multiple types of input are required, e.g., from multiple systems, or certain additional information which can support the logic of the anomaly detection need to be applied.

A typical example consists of a light/switch pair. The light being either on or off is nominal, the same goes for the switch, but having the switch on and the light off shall be considered anomalous. Especially these types of anomalies can be easily detected by comparing real and simulated TM. Some specific examples for contextual novelty detection for operations can be mentioned:

- General pruning of the output of the content anomaly detectors. Filtering the identified anomalies for real anomalies based on context, to avoid high number of false positives. Example: anomaly detection of performance for electrical components on a spacecraft, in context of space weather (e.g. solar flares)
- Detection of contextual novelty in remotely sensed data, using multiple types of data (e.g. overlapping measurements but performed with different light spectra)
- Critical changes in pressure within the propulsion system network, in the context of status of valves.

3.2 Anomaly Diagnosis

Beyond just the detection of the anomaly, the next step would be to support the operations teams to understand possible causes of an anomaly. Anomaly diagnosis can be seen as an extension by adding context to the anomaly. This can be achieved by the concept of a digital twin, where a digital version of the system would allow to quickly link anomalies detected with their location in the system. Likewise, a look-up failure list, which could be made based on concepts of active learning for example, could also be a means of allowing a system to look up potential sources for anomalies. Therefore, the objectives within this use case can also be linked to the topic of knowledge management. In case a novelty detection application has the understanding about the subsystem (e.g. based on a look-up table) where the anomaly appeared, the anomaly detection AI/ML models which the novelty detection algorithm is part of, could use Natural Language Processing (NLP) techniques to search for additional information related to the failure in supplementary sources (e.g. user manual).

Another application that has less focus on building hybrid AI solutions, as mentioned above, could be teaching a system on cause and effect, and therefore how to react to dedicated FDIR alarms. This would likely be tied to the concept of reinforcement learning and would be subject to direct interaction with the simulator.

3.3 Predictions

Prediction within operations is highly correlated with forecasted TM values (e.g. future thermal power consumption), environment conditions (e.g. when is the next cross of the radiation belts), the risk of collision with space debris, etc.

In such cases, a machine learning approach may learn from already existing data to make predictions of the future. These predictions will be operationally useful in the sense that they can be considered by the mission planning system or, alternatively, used to take better decisions.

The types of prediction can generally be split up into different types:

- Spacecraft internal predictions: highly based on combining real and simulated TM, to predict future values
- Lifetime prediction: A specific use-case is lifetime estimation. Here it might be valuable to have several datasets from different satellites, or data about impact (e.g. from radiation/space-weather), to understand the condition change of on-board equipment (e.g. solar panels or batteries) over time. Simulated TM in the loop plays a central role so the AI/ML models can automatically test and validate different scenarios.,

4 Conceptual Theory

As described above, different aspects of the infrastructure require different building blocks. The infrastructure conceptualised can be the basis for further usage beyond the aspects covered in this paper and is therefore intended to be extensible and flexible with respect to the systems with which it will be used. The preliminary architecture is described in section 4.1. The infrastructure setup needs to be followed by the AI/ML modelling phase including several preparatory actions such as data access, data pre-processing, and data separation followed by the actual model building and model training. Data access can be granted with the aforementioned infrastructure. The acquired data needs to be pre-processed, including data cleaning and editing, data reduction, and feature extraction.

4.1 Overall Idea

The first iteration of the architecture (digital twin framework) is provided in the figure below. The data types and interfaces are also provided. To support extendibility and re-use for potential projects based on this concept, the architecture is designed to be modular, allowing building blocks to be adapted and new applications added.

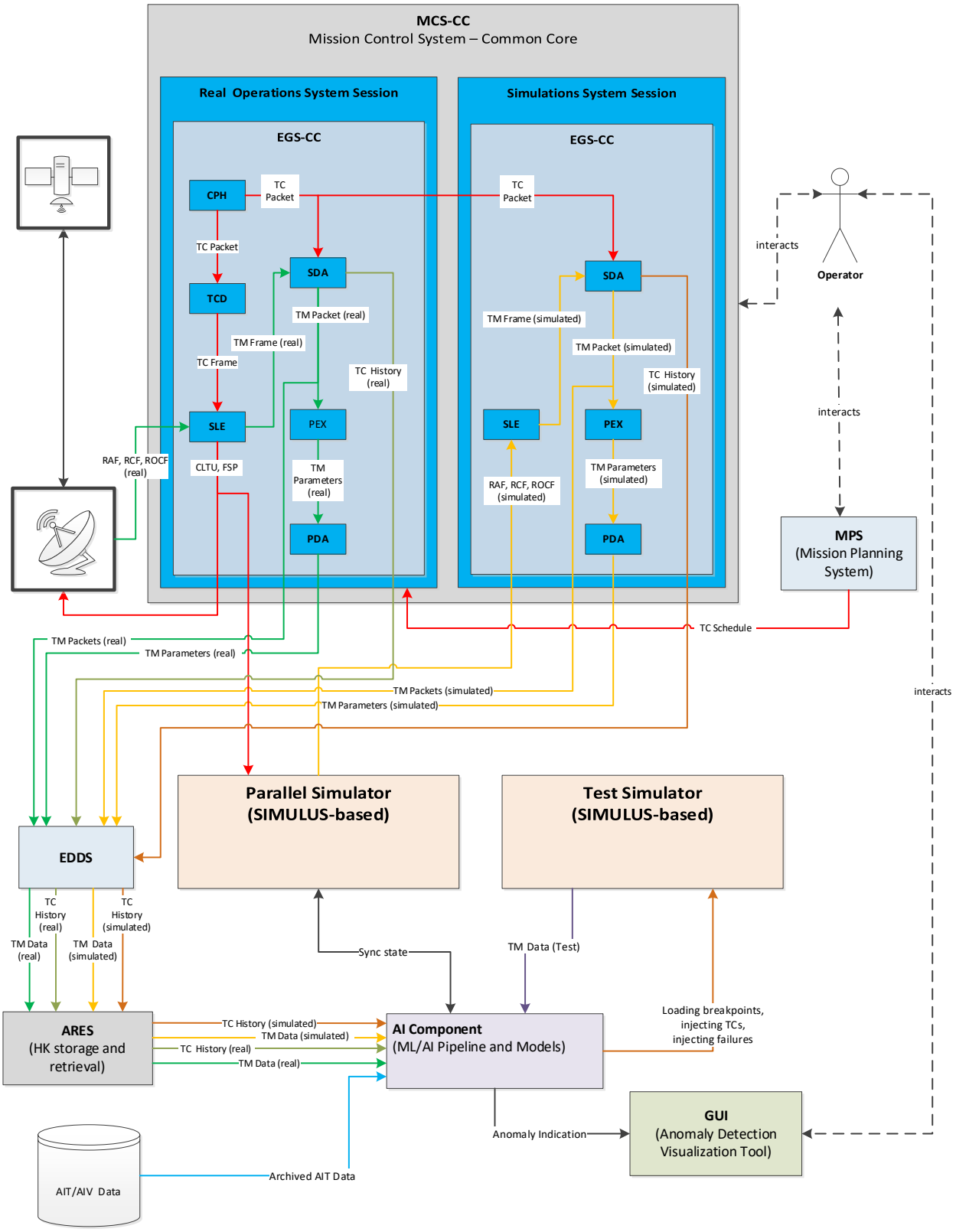

Figure 8 Architectural concept of the building block infrastructure

The following sections describe the data flow and the interaction of the different systems shown in the architecture.

4.1.1 Assisted Spacecraft Operation

The central user within this concept is the spacecraft operator being responsible for commanding and monitoring the health of the spacecraft. For the actual envisaged solution, the system should ideally augment the already existing spacecraft monitoring and control setup, to allow the operator to monitor diagnostics data in a Human-Machine-Interface (HMI) to which the operator is already accustomed to. However, the actual health monitoring of the spacecraft itself could alternatively be displayed and monitored on a newly developed Graphical User Interface (GUI) which is connected to the system.

4.2 System Interfaces and General Architecture

This section describes the interfaces contained in the architecture solution shown in Figure 8.

4.2.1 Spacecraft and Simulator in the Loop

One of the core elements of the architecture is the SIMULUS based spacecraft simulator containing the ground models, the spacecraft models as well as the environmental models. Once the operator generates and sends TCs to the real spacecraft in orbit, the same TC packets will also be sent to the simulator, forcing the generation of specific simulated TM frames. In case the usage of a fully developed spacecraft simulator is not foreseen, the TM generation can also be based on several relatively realistic engineering models already used during the spacecraft design phase or alternatively on a dedicated number of spacecraft simulator components. These reduced models are not meant to simulate the entire spacecraft but to allow the focus on specific spacecraft components with a certain level of TM generation accuracy to have some confidence in the later diagnostics processing.

In the further chain of the architecture, the real spacecraft sends back the actual TM data to ground which will be then post-processed within the EGS-CC components and made available to ARES via EDDS. The simulated TM frames will also be passed to the EGS-CC SDA via SLE to be archived within the SDA. This data will then be post-processed in the same way as the real TM and will also be made available to ARES via EDDS. With this, an MCS/SIM in the loop approach is proposed.

4.2.2 Connecting the MCS with Spacecraft and Simulator

CCSDS SLE interface between G/S and MCS

As per the CCSDS standard, the Space Link Extension (SLE) Reference Model identifies a set of SLE transfer services that enable missions to send forward space link data units to a spacecraft and to receive return space link data units from a spacecraft. However, the CCSDS SLE standard deliberately do not specify the methods or technologies required for communications [13]. But, typically for ESA missions, the SLE communications protocol between MCS and G/S is TCP/IP.

The SLE component of EGS-CC implements the Generic TM/TC Data Interface specification. It encapsulates all SLE services in a single component, offering all means to use them from within components of the M&C Adaptation subsystem. The SLE services include all SLE protocols currently defined for transfer of telemetry and telecommand data between ground stations and mission control systems: for telemetry it includes RAF (Return All Frames) service, RCF (Return Channel Frames) service and ROCF (Return Operational Control Fields) service, for telecommand it includes CLTU (Forward CLTU) service, but FSP (Forward Space Packets) service has been excluded due to being out of projects scope at this time [12].

TC data flow

As shown in Figure 8, the same TC frame (CLTU or FSP) would be sent by MCS-CC Real Operations System Session via the SLE interface to distinct TCP/IP addresses and ports corresponding to real or simulated G/S. The problem is that, within an MCS-CC System Session, by default MCS-CC can connect via SLE to either the real or simulated G/S. Modification of MCS-CC deployments scripts and eventually EGS-CC SLE component code might be needed to get MCS-CC connected via SLE simultaneously to real and simulated G/S**.**

After sending the TC, MCS-CC Real Operations System Session would archive the TC into SDA and display it in the TC History with TC verification stages updated upon reception of corresponding real TM (e.g., after MCS-CC

receives TM PUS(1,7), the TC verification stages are updated showing that the S/C on-board execution of the TC was successful).

On the other hand, MCS-CC Simulations System Session shall also provide a TC History in which TC verification stages would be updated upon receiving simulated TM. Therefore, a mechanism shall be implemented to get the TC archived in MCS-CC Real Operations System session synchronously archived in MCS-CC Simulations Systems Session.

TM data flow

As shown in Figure 8, MCS-CC would simultaneously receive TM frames (RAF, RCF or ROCF) from real and simulated S/C in separate SLE TCP/IP addresses and ports. TM packets are decoded and stored into distinct SDA per MCS-CC Real Operations or Simulations System Session. TM packets are then retrieved from SDA and processed to extract out of them TM parameters which are then stored into distinct PDA per session.

Data retrieval from SDA and PDA is described in 4.2.3.

4.2.3 Data Pipelines and Data Storage

MCS-CC archive

Archive (ARC) is part of the EGS-CC Kernel, its purpose is to persistently store:

- Processed data archived in PDA - state of MCM objects (such as state of a parameter, event occurrence or activity occurrence).
- MCM object definitions.
- MCM input data – source parameters (not from telemetry), activity invocations, event triggers for use within playback or reprocessing.
- Raw and source data archived in SDA (such as TM packets, TC packets, M&C information sent to and received from other controlled systems).

Archive provides interfaces for both storing and retrieving the data from the Archive. The interfaces for data retrieval provide extensive options for filtering the data within the Archive increasing this way the performance of the retrieval and decreasing the amount of data to be transferred. The retrieval interfaces of Archive are synchronous and do not provide subscriber type of continuous data delivery. The subscription based continuous delivery is provided by M&C Access API for processed data and Source Data Access for source data [14]. The exposed APIs for Archive data access can be used directly by the digital AI twin to retrieved data in case of direct access to MCS-CC, otherwise it can retrieve data via the API exposed by Data Dissemination Systems.

Data Dissemination Systems

EDDS is the EGOS Data Distribution System provide controlled and secure access to mission data for users who do not have access to the mission control system (MCS) monitoring and control facilities. Adapters provide access to different types of data sources. EDDS makes use of the JEEL library to access the EGS-CC Archive service. ARES is the Analysis and Reporting System responsible for collecting and storing housekeeping telemetry for offline analysis and reporting purposes. ARES is populated with telemetry files provided by EDDS with data retrieved from the EGS-CC Archive. The digital AI twin would use ARES API to retrieve monitoring data from MCS-CC, as raw data (e.g., TM Packets, TC History) or processed data (e.g., TM Parameters) [10].

Mission Planning System

MPS is a fundamental component of all spacecraft missions. Its task is to balance the needs of spacecraft users, with the constraints and resources of the spacecraft and its ground segment. A human operator is expected to use MPS to send scheduled TCs from MPS to MCS-CC. The digital AI twin is not expected to use MPS API.

4.2.4 AI Component

By providing an infrastructure for both real and simulated data pipelines, the digital twin infrastructure opens a broad range of possibilities for data processing applications. The section hereby presented shall focus on integrating the building blocks mentioned in Section 2.3 to build a representative use case that leverages the concept of the digital twin for anomaly detection and root cause analysis.

For this, we shall consider a scenario where a new batch of satellite TM and TC data is coming and the operator would like to understand whether this data is anomalous and if so, what the root cause of said anomaly was.

To carry out this task, a digital twin can leverage an ensemble of AI models. Considering Section 2.3.2, LSTMs and AE are two models typically used for anomaly detection tasks as they are reasonably generalisable and capable of handling multivariate datasets. As such, these models should not rely as much on domain knowledge nor heavy pre-processing. However, they would need to be trained with a vast amount of TM and TC data which would require significant computational resources and/or time.

After training to a satisfactory level of performance, the models will infer on the upcoming TM and TC data batches to detect anomalies. The LSTM model will predict the next batch of TM values and compare those with the new data. Concurrently, the AE will reconstruct the input data based on the parameters learned on the nominal data.

Although no discussion for the anomaly scoring method is envisaged for the scope of this work, we can foresee that for the present use case, it is preferable that that the model ensemble identifies as many anomalies as possible rather than missing potentially critical issues. In practice, this means the anomaly scoring method should mitigate false negatives first.

Another consideration is the weight attributed to each anomaly score from the ensemble. From a preliminary analysis it would seem that said weight should be proportional to the performance of the models on the new dataset (i.e. using mean squared error loss function). In simple terms, this means that whichever model was better able to fit the new batch of satellite data should have more authority in understanding whether a value is anomalous.

If the ensemble identifies an anomaly, the operator is alerted through the GUI as illustrated in Figure 9.

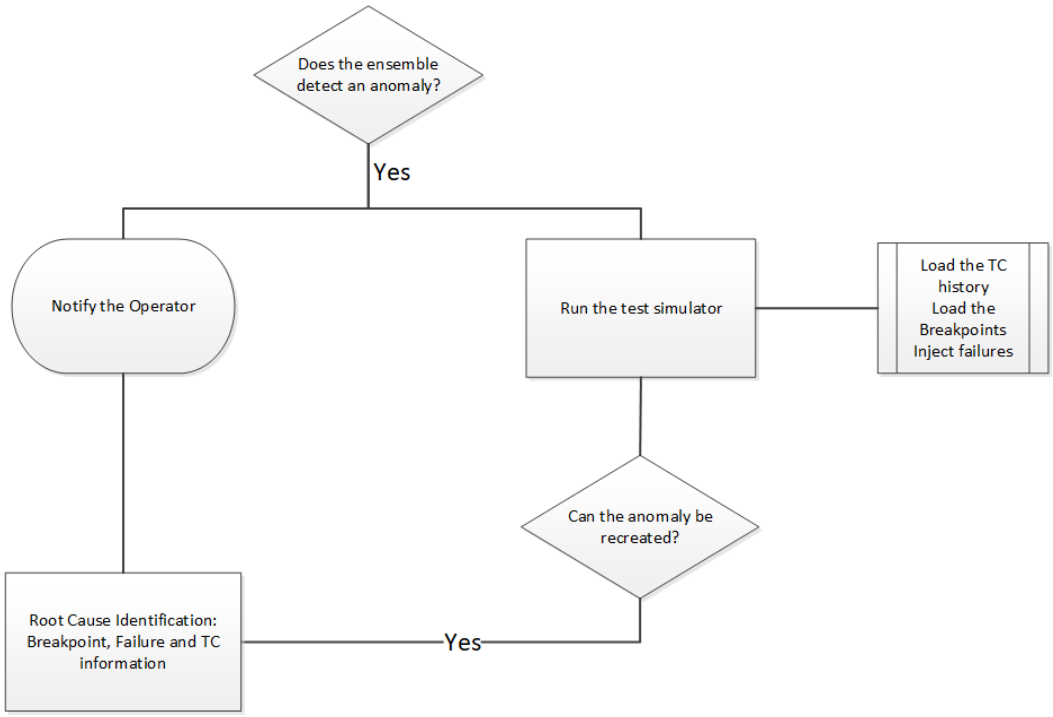

Figure 9 Flowchart for anomaly detection in the digital twin infrastructure

The test simulator then loads the breakpoints and the TCs sequences to replicate the spacecraft state (this assumes the simulator is of high-fidelity). From here, several datasets shall be generated and compared to the original anomalous data. If the anomaly can be replicated (which can be assessed by many potential metrics, one of which being Euclidean Distance) then operators can gather information about the root cause of the issue such as the spacecraft state, the failure type, and the TC sequences involved in the issue.

5 Discussion

This section presents a discussion of the implications of the digital twin concept, including the requirements thereby imposed on the internal operational simulator, the potential benefits of such a system, as well as the limitations and foreseeable problems such a system may be subject to.

5.1 Requirements on the Operational Simulator

To produce a DT using an AI-based approach, the operational simulator at the core of the DT must possess a certain level of accuracy, or fidelity, to the real spacecraft. While it is understood that the simulator must provide a solid baseline for the comparison of its telemetry output with that of the real spacecraft, the specific requirements will depend on the use case (or cases) envisaged by the mission's flight control team and simulation officers, as well as on the specific AI algorithm selected. Due to their complexity, operational simulators usually require considerable development time, with frequent interaction and feedback from the flight control team, to be considered mature enough for their use in simulator campaigns prior to and during mission operations. It can therefore be expected that a simulator will only be usefully integrated into a full-spacecraft DT when it has reached this mature stage.

It is critical that the software requirements specified for the operational simulator thoroughly foresee and capture the mission-critical functionality expected from it, while remaining testable, in advance of the bulk of the simulator's development. Additionally, the accuracy of an operational simulator relies heavily upon good quality and detailed documentation on all the real spacecraft's subsystems to ensure that the software's design and implementation is a good approximation of the hardware.

Figure 10 illustrates the relevant core components and interfaces of a SIMULUS-based simulator instance.

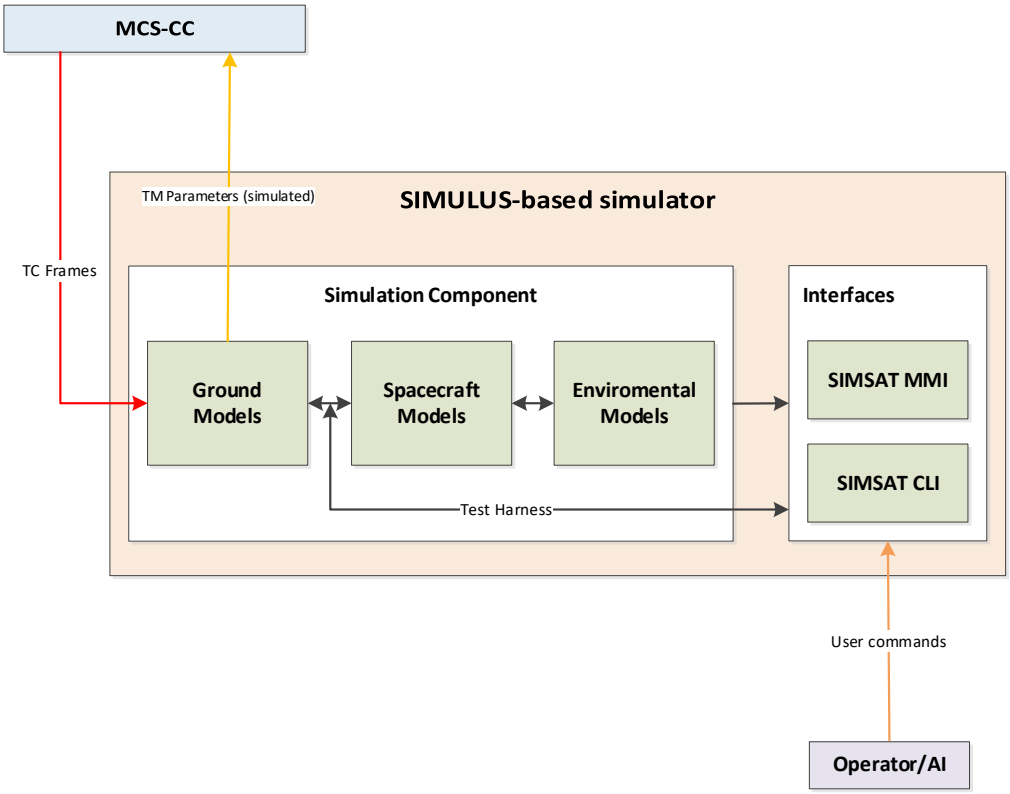

Figure 10 Schematic of a running operational simulator

5.1.1 Components of the Operational Simulator

As can be seen in Figure 8, two instances of the simulator are portrayed: one running in parallel to, and receiving the same commands as, the real spacecraft, with another instance purely for testing in a closed loop with the AI algorithm. Both simulator instances are identical except for variations applied by the AI algorithm to the scenario and configuration of the test simulator. This paper suggests that a SIMULUS-based simulator is used at the core of the DT. The SIMULUS infrastructure offers many advantages for the simulation of complex space systems, including a large toolbox of classes and models which can be used as a basis for development of subsystem models, a built-in runtime infrastructure, and a graphical user interface (SIMSAT MMI). However, regardless of the baseline infrastructure used, a full-spacecraft operational simulator will consist of three main categories of models (see Figure 10):

1. Spacecraft models, which simulate the various subsystems onboard the spacecraft and their interfaces. Each real spacecraft subsystem must be represented by a model which provides the necessary functionality specified in the software requirements alongside that which is needed for other connected subsystems to function. These models must also respond to commands with telemetry accurate to that which the real spacecraft subsystem provides.
2. Environment models, which simulate the orbit, body dynamics, and external environment of the spacecraft. These models must be sufficiently accurate to permit the various sensors, actuators, solar arrays, and antennae models to produce realistic measurements and effects. Again, this ensures that the generated telemetry is a close facsimile to that which the real spacecraft would generate.
3. Ground models, which represent the ground stations which will track and communicate with the spacecraft. These simulate the link from ground to space and must be configured with the locations and parameters (e.g. terrain masking) of the real ground stations. They also provide the interface to the mission control system using the SLE protocol, such that TCs and TM may be transmitted to and from the simulator in the same manner as for the real ground stations.

5.1.2 Emulating the On-board Software

The most critical component of an operational simulator is the on-board computer (OBC) model, which contains a processor emulator so that the real on-board software (OBSW) can be run in the simulator. This model runs an exact copy of the OBSW so that the responses to TCs, generated TM, and all internal processes match those of the real OBC. This in turn imposes fidelity requirements on the connected subsystems of the spacecraft: they must be of sufficient accuracy to allow the OBC and OBSW to operate without errors. This extends to the internal fault detection, isolation, and recovery (FDIR) processes of the OBSW, which can be considered a key indicator of the maturity of the simulator: typically, a simulator can be considered mature and accurate when, under nominal conditions, when running nominal operations procedures and with a mature OBSW version, it triggers no FDIR events (unless the real spacecraft would also trigger those events). Therefore, the simulator models must also be sufficiently accurate to cause the OBSW to detect errors and produce the appropriate FDIR response, where such errors are the result of either user-injected built-in failures of the models or because of commanded scenarios which exceed normal FDIR ranges. The comparison of FDIR events raised by the simulator and those raised by the real spacecraft represent a key mode by which the DT could detect discrepancies or anomalies in either source.

5.1.3 Simulator Scheduling

It is essential that timing aspects of the hardware are taken into consideration when developing simulator models so that model responses occur with the correct delays relative to their hardware counterparts. Multi-input, multi-output (MIMO) operations must have defined wait states to realistically model execution time, transactions on buses must be subject to delays based on bit rates of the bus or connection, and events must be scheduled using the simulator scheduler. SIMULUS currently provides an SMP2 scheduler, but more accurate results can be attained by using the Terma emulator (TEMU) and its scheduler. At the time of writing, the SIMULUS-provided emulator is an instruction-set-interpreter emulator which has performance limitations and can cause non-deterministic timing behaviour, while TEMU offers a higher performance factor and deterministic timing. However, the emulator (and therefore scheduler) selection will depend on support for OBC processor architecture. It is noted that TEMU is highly flexible and can be configured for many different processor models [9].

Generally, simulator schedulers can be run at either a fixed speed factor (i.e. a multiple of real time), or in a free-running mode where the simulation executes as quickly as permitted by the computer hardware which is running the simulator. The achievable free-running speed factor depends on the specifications of the computer hardware, so the AI algorithm could run scenarios in the test simulator at speeds several times faster than real time, while the parallel simulator would generally be run at a speed factor of 1.0 and with epoch matching the real mission.

5.1.4 Non-critical Functionality

Inevitably, there will be elements of the functionality of the real hardware which are not considered critical for simulation and are therefore only representatively modelled in the simulator. The specific telemetry related to these representatively modelled functions will have to be filtered by the AI algorithm so that false positives are avoided when detecting discrepancies between the real telemetry and that provided by the simulator. An example of this could be a star tracker model which does not use a star catalogue or image processing to produce an attitude measurement for the spacecraft – instead, it takes its attitude directly from the dynamics and environment models

and only provides representative or fixed values for telemetry regarding the measured star coordinates and magnitudes, and so on. Direct comparison of the telemetry produced by the real star tracker with that generated by the simulated star tracker will obviously show discrepancies for these values, and the AI algorithm must be configured to ignore these.

5.1.5 User Interface

For user oversight of the running simulator, a graphical user interface (GUI) is a useful feature. SIMULUS-based simulators offer a built-in GUI, called the MMI. Among other features, this provides access to all published properties and states of the running simulator, the simulator log, and a suite of controls for intervention in the simulator. Through the MMI, the user can directly inject Javascript commands and scripts containing sequences of commands into the simulator, such as for the saving and loading of breakpoints or the injection of built-in failures. However, SIMULUS also provides the SIMSAT Command Line Interface (CLI) for running and commanding the simulator without the use of the MMI. Such an interface would be used by the AI algorithm to start and stop the simulation, save, and load breakpoints, edit the simulator's configuration, and inject user commands and failures.

5.1.6 Test Interface

To facilitate the interaction of the AI algorithm with the test simulator (as shown in Figure 8), an additional requirement is the provision of a testing interface through which TCs can be directly injected and TM checked without transmission via the simulated ground stations to the MCS. For a SIMULUS-based simulator, this is provided by the UMF Test Harness, which permits injection of TCs to the OBC and contains a pool of TM parameters which are decoded according to the installed spacecraft database (SDB). In essence, the Test Harness bypasses the ground station models and permits a user (or the AI algorithm) to inject TCs and check the effects on the reported TM without involving the MCS. In this way, the test simulator can be operated by the AI algorithm in a closed loop separate from the simulator running in parallel to the real spacecraft. The Test Harness can be accessed either via the MMI, or, as is more likely for the AI algorithm, via the CLI, avoiding the computational overhead of the graphical user interface.

5.1.7 Breakpoint Generation

An additional crucial feature the operational simulator must possess is the ability for the user, and therefore AI algorithm, to save and load breakpoints. These breakpoints must save the complete state of the simulator at a fixed instant of simulator time and can be loaded by the user to return the simulation to that exact state. These breakpoints can then be used by the AI algorithm in closed loop with the test simulator to quickly reach and replicate desired simulation states for testing. Breakpoints can also be used for re-synchronising the parallel simulator as will be described in the following section.

5.1.8 Synchronisation of the Parallel Simulator

Operational simulators are very complex and highly computationally demanding, and situations may arise where the simulator becomes unresponsive or crashes. This presents an obvious problem for the simulator running in parallel to the real spacecraft, but this paper proposes a mitigation strategy in this section.

During normal operation, it is proposed that breakpoints are saved at regular intervals (e.g. 30 minutes) and are either stored singly or in a cycle of multiple breakpoints. Since breakpoint files are usually quite large (on the order of gigabytes), the number of breakpoints to be stored can become a driver for simulator memory requirements. Breakpoints also require some time to save, and the simulator scheduler must be stopped for the duration. This will inevitably cause the simulator to fall behind the real mission epoch. However, as described in Section 5.1.3, the simulator scheduler can be set to free-running mode until it has caught up with the true epoch, at which point the scheduler speed factor can be returned to 1.0, thereby re-synchronising the simulator with the real spacecraft.

In the case of the simulator becoming unresponsive or crashing, the simulator can be restarted, and the most recent breakpoint loaded, whereupon the simulator can again be set to free-running mode until it returns to the correct epoch. Since the AI platform will have a record of the TCs sent to the spacecraft in the intervening time between the crash and the reload, it can inject those TCs directly via the Test Harness (as described in Section 5.1.6) to ensure that the simulator reaches the correct state at synchronisation.

This procedure could be automated as a script, with intervention from the AI only when injecting the missed TCs.

5.2 Enhancing Simulator Capabilities

Prior to their use in simulation campaigns by spacecraft operators, simulators must be configured with accurate values for all configurable parameters e.g., thermal responses and built-in sensor/actuator errors. These values are selected to closely match the performance and calibration of the real hardware, and some simulator subsystems e.g., the simulated thermal network, can take multiple iterations of configuration to become fully tuned. The configuration of the simulator will then naturally diverge from the real spacecraft as the mission progresses, with degradation and failures of hardware components needing to be manually accounted for and applied to the simulator by the operator.

It is therefore proposed that the DT can be used to improve the operational simulator's accuracy over the mission lifetime. When the AI detects a recurring discrepancy between the TM reported by the real spacecraft and that generated by the parallel simulator, it can test the effects of changes to the configuration of the test simulator to identify the source of the error. For example, if the parallel simulator reports a measured voltage which is consistently different to the value of the same TM parameter received from the spacecraft, it can change the error (scale factor, noise, or bias) applied to that specific parameter inside the test simulator and check the reported effect. If the AI thereby finds that a minor configuration change reliably solves the discrepancy between the two TM parameter values, it can then make a recommendation to the operator that the configuration of the parallel simulator be changed. In this way, the operator can be provided with valuable insight into the source of variations between the spacecraft and simulator which are otherwise too small to trigger OBSW FDIR actions. The detection and diagnosis of these errors would either indicate simulator configuration errors or, in the case of an accurately tuned simulator, degradation of spacecraft components e.g. in advance of hardware failures.

While it can be argued that such investigations could also be performed by a human operator, the use of AI to continually monitor the low-level TM data from the real spacecraft and compare it with simulated results in real time offers both a reduction in workload for operators and faster detection and diagnosis of small discrepancies than a human operator could achieve. Note from Figure 8 that the AI cannot directly change the configuration of the parallel running simulator (it can only synchronise the simulator state), while it can freely edit the configuraton of the test simulator. This is to prevent the AI from bypassing the operator and making unauthorised changes to the simulator baseline; however, such a feature could be implemented in the same manner as for the AI's interface to the test simulator.

5.3 Increased Situation Awareness

Thousands of packets from a given satellite arrive at the mission control center every day. In fact, today's systems already help tremendously in handling this flood of data, but in the event of an FDIR alarm or even an anomaly, the space operators can quickly reach their (load) limits. In addition, it is also possible that several problems overlap in a short period of time, which makes the workload increase even more in the short period of time. In the worst case, this can even lead to the operators losing their situational awareness, at least for a short time, as they are busy processing the messages marked as urgent.

A clever assistance system which relieves the space operator of unnecessary preparation work, reduces the information to be shown to the essentials, and calculates possible causes of errors in advance and displays them, can significantly reduce the workload of the space operator in critical phases. AI-based anomaly detection and root cause identification would not only reduce the workload, but also allow the operator to focus on recovery and action planning while continuing situational awareness. Those systems could even support the operator with suggested recovery action recommendations, based on previous data, simulated results, or any other AI experienced activity.

Moreover, such systems could detect anomalies and possible existing or impending faults and problems even when there is no FDIR alarm, thus providing preventive insights that lead to further system and situational awareness of the operators.

5.4 Maturity of EGOS-CC Products

In recent years, EGOS-CC products have gained substantial maturity accomplishing major developments. MCS-CC is on the verge of deprecating Mission Control Systems based on SCOS-2000.

MCS-CC underwent some real S/C operations testing. On 26 June 2021, ESA's OPS-SAT space lab became the first spacecraft to be monitored and controlled using the EGS-CC – proving that this software of the future is ready to be extended across current and future missions flown from Europe [15]. On 19 January 2023, an important milestone was achieved: ESA's new Ground Operation System Common Core (EGOS-CC) commanded a major scientific mission in Earth's orbit for the first time. It successfully interacted with one of the three Swarm satellites (Swarm-B) during a routine ground station pass. Mission controllers checked the satellite's status, uplinked

commands and received data from the spacecraft. The successfully executed commands mark an important step on the journey towards ESA-wide adoption of this new, first-of-its-kind multi-mission control system by 2025 [16].

5.5 Potential time delay problems for TC Packets transfer from Real Operations to Simulations System Session

It could be that transferring TC Packets from Real Operations System Session into SDA in Simulations System Session takes substantial amount of time to complete, in the meantime MCS-CC could have received corresponding simulated TM and didn't process it to properly to update the TC verification stages.

Once the foreseen setup is established, such time delays shall be measured, and if evaluated too long and breaking some functionalities, the root cause shall be identified, then mitigation actions shall be taken accordingly, e.g., possible solutions could be:

- Optimise the performances by re-engineering the deployment (e.g., deploy applications in a distributed environment, with archive applications running on dedicated machines)
- Introduce similar time delays on the TM processing chain.
- Store received TM in files and only start ingesting them into MCS-CC at regular time periods.
- Etc.

5.6 Limitations on the AI Models and Possible Issues During Runtime

Working with AI can cause several challenges and issues which must be addressed accordingly. The initial challenge begins with the training of models itself. The AI models described in this work are simple in order to provide an illustrative example for the potential of the digital twin. Having said that, the multidimensional TM data takes up large amounts of computational resources during training. One potential solution could be to cut down on parameters or on the time window considered for the training dataset. Finally, another solution could be to consider individual parameters or group of parameters rather than the multivariate approach.

Another important aspect relevant for the selected use case is the fact that the simulator itself is not be a perfect machine and thus simulating the anomalous behavior of the spacecraft might not be perfectly achievable. However, some of the limitations of the simulator are known and as such corrective measures can be embedded in the pre-processing step. In a similar fashion, iterating on an appropriate anomaly scoring criteria is also a fundamental task which will require some research and experimentation.

During runtime there is always a risk of poor performance from the models leading to a lack of robust and reliable results. Operators can either miss anomalies or lose trust in the models if it outputs too many anomalies. This can be mitigated by a thorough training and testing process which ensures that the models are adaptable and able to perform (hence the particular choice for multivariate models). However, it is also important to consider that the models here selected are not state of the art and were considered on a trade-off between simplicity and envisaged performance for a representative hypothesis.

As for any AI application, a lack of suitable data or a lack of quality in the data needed for training the AI algorithms might restrict potential use cases. ARES is being seen as a suitable technical source for data, especially for missions with large datasets. However, dedicated use cases may lack sufficient volume for both training and validation and therefore, the use case selection requires a careful assessment.

6 Conclusions

In this paper a subset of different concepts were shown with which a collective approach can be taken in order to combine existing technologies and concepts of space operations and artificial intelligence in order to create a first version of a spacecraft digital twin. The paper discussed all possibilities but also limitations with such an approach. The discussed digital twin is still dependent on the TM-based communication with the spacecraft and cannot indicate an in-situ live status of the spacecraft without the usage of the TM-based communication. However, with the potentials shown in this paper, the rapid developments in artificial intelligence as well as further potential which can be uncovered using simulators, efforts can be taken in order to build a digital twin of the real spacecraft which has the capability to be a digital in-situ and live representation of the real spacecraft and its status even without a direct communication line.

At some point in the future this kind of digital twin would allow to replacement of the current human-centric mission operation with a so called "Artificial Operator" that is capable to automatically perform multiple activities that are performed today manually by humans. The role of the mission operator would then switch to a supervisor of the automated artificial operator which is driven in its core by the described digital twin.

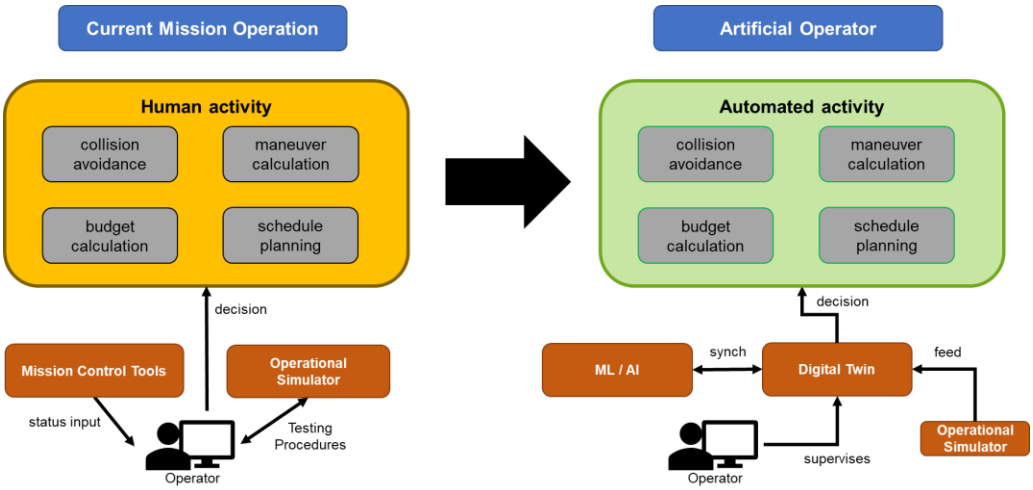

Figure 11 Transition of Mission Operation

References

[1] M. Grieves, "Digital Twin: Manufacturing Excellence through Virtual Factory Replication", 2015

[2] E. Glaessgen, D. Stargel, "The Digital Twin Paradigm for Future NASA and U.S. Air Force Vehicles", in 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Honolulu, 2012.

[3] Y. Wang, J. Gong, J. Zhang and X. Han, "A Deep Learning Anomaly Detection Freamework for Satellite Telemetry with Fake Anomalies," *International Journal of Aerospace Engineering,* 2022.

[4] J. A. Martinez-Heras and A. Donati, "Novelty Detection with Deep Learning," in *2018 Space Ops Conference*, Marseille, 2018.

[5] B. Pilastre, L. Boussouf, S. D'Escrivan and J.-Y. Tourneret, "Anomaly Detection in Mixed Telemetry Data Using a Sparse Representation and Dictionary Learning," *Signal Processing,* vol. 168, 2020.

[6] T. Yairi, N. Takeishi, T. Oda, Y. Nakajima, N. Nishimura and N. Takata, "A Data Driven Health Monitoring Method for Satellite Housekeeping Data Based on Probablistic Clustering and Dimensionality Reduction," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 53, no. 3, pp. 1384-1401, 2017.

[7] A. Dertat, "Applied Deep Learning - Part 1: Artificial Neural Networks," Towards Data Science, *https://towardsdatascience.com/applied-deep-learning-part-1-artificial-neural-networks-d7834f67a4f6,* (accessed January 27 2023).

[8] A. Dertat, "Applied Deep Learning - Part 3: Autoencoders," Towards Data Science, *https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798,* (accessed January 27 2023).

[9] M. Holm, "TEMU 3 and 5: High Performance Virtual GR740 Platform," in *GR740 User Day*, Noordwijk, 2022.

[10] EGOS-CC Products Overview. Issue Date 24/10/2018. Ref ESA-EGOS-CC-TN-100.

[11] EGS-CC Sessions, *https://csde.esa.int/confluence/display/EGSCCKB/.Sessions+v1.0*

[12] EGSCC-SYS-SSDD-3080 Space Link Extension Services, *https://csde.esa.int/confluence/display/EGSCCKB/.EGSCC-SYS-SSDD-3080+Space+Link+Extension+Services+v1.0*

[13] CCSDS Space link extension: Internet Protocol for Transfer Services. CCSDS 913.1-B-2.

[14] EGS-CC Archive, https://csde.esa.int/confluence/display/EGSCCKB/.EGSCC-SYS-SSDD-2050+Archive+v1.6

[15] First test of Europe's new space brain, https://www.esa.int/Enabling_Support/Operations/Ground_Systems_Engineering/First_test_of_Europe_s_new_space_brain

[16] EGOS-CC: commanding satellites from the shadows, https://esoc.esa.int/content/egos-cc-commanding-satellites-shadows