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

SpaceOps-2023, ID # 314

HOW MBSE CAN HELP AIV DOMAIN. A PRACTICAL APPROACH

Nieves Salor Moral [a], Pablo Beltrami [b], Alex Vorobiev [c], Sam Gerene [d]

[a] *RHEA Group, Madrid, Spain, n.salor@rheagroup.com*
[b] *RHEA Group, Robert-Bosch-Str 7, 64295 Darmstadt, Germany, p.beltrami@rheagroup.com.*
[c] *RHEA Group, Diegem, Belgium; a.vorobiev@rheagroup.com*
[d] *RHEA Group, Leiden, The Netherlands; s.gerene@rheagoup.com*

Abstract

The application of Model-Based Systems Engineering to manage the growing complexity in system design and development starts at the beginning of a project on the left side of the Systems Engineering V-model by conducting analysis of the problem and the potential concepts on functional and logical level all the way down to specifying the concrete technical solution on physical level. However, little application is currently seen in the space industry to apply MBSE as support for the activities on the rights side of the V-model, i.e. for assembly, integration and testing/verification and validation.

The objective of the Model-Based System Engineering for AIV (MBSE 4 AIV) system, subject of this paper, is to bring the same principles and related benefits to this later phase activities by creating a single-source of truth for AIV relevant-information capture and exchange between different tools for on/off-site AIT test campaigns to avoid manual work and inconsistencies.

The production of a tool suitable for the reporting of AIT and Verification activities using a language understood by all parties is, thus, the final goal of the project presented in this paper.

But how will all parties share the same understanding? And how can the documentation of such understanding be automated? The answer to those questions implies the digitalization of the engineering tasks throughout the lifecycle. Digitalization of the information will allow automation and ensure data persistence while conceptualization will provide interoperability. Both topics are involved within the MBSE domain, which is core to this development and roadmap.

In 2017 a collaborative effort between the Agency and European Large System Integrators (LSIs), took place to define an ontology focused on the reporting needs for AIT & AIV stages. During this effort, a comparison of the contents of real AIV reports created (mostly manually) and used by the LSIs was performed. It was clearly seen that while different reports shared some commonalities, the required contents of test reports differ significantly across departments and test activities, and none of the inspected test reports achieved a complete coverage of the required content as defined in ECSS-E-ST-10-02C [1] standard. Furthermore, the LSIs identified missing contents on manually created reports which were deemed useful and should be added.

From that analysis the following was drafted:

- requirements for a software reporting system,
- user stories representing the expected features and
- a data model named Test Report Standard Ontology (i.e. TRS) [2]

This paper presents the work performed on the creation of a conceptual data model able to capture all relevant AIV knowledge in integration with the Space System Ontology (SSO), an effort ESA is currently undertaking, and the implementation of a tool to exchange and generate reports compliant with those semantics. From that model, physical schemas have been automatically generated to be used for the digitalization of the involved data and for the definition of exchange interfaces used by the AIV reporting tool being implemented which will satisfy industrial selected AIV Use Cases.

Keywords: MBSE, AIV, ORM, Hub, Model, Interoperability, Exchange, Report, Displays, code generation

Acronyms/Abbreviations

MBSE (Model based System Engineering), AIT (Assembly, Integration and Testing), API (Application Programming Interface), LSI (Large System Integrator), TRS (Test Report Standard), SSO (Space System Ontology), UoD (Universe of Discourse)

1 Introduction

The envisioned way to implement digitalisation in the European space industry is in the form of the MBSE System Factory, an integrated software and hardware infrastructure to support the MBSE process. The objectives

of the System Factory are to streamline the space system development process across the entire life cycle, i.e., shorten the development time and reduce costs of projects, manage the increasing complexity of space missions, and offer process guidance and automation to support the development of ECSS compliant space systems.

There are many tools used in model-based activities in different engineering disciplines, but the exchange of information is typically done by manual processes, which has the disadvantages of work duplication, low automation, and lack of data normalisation.

The MBSE4AIV System, target of this paper and part of the system factory, aims to reduce the overhead of such issues including the document preparation, consistency checking and analysis of the validation activities during the generation of test reports applicable to the AIV domain. It will also improve the information exchange of data between AIT/AIV users to be used during the testing campaigns as required information comes from different sources (e.g. Excel, Word, SQL) which can create inconsistencies or duplication of information undetected until too late in the validation process.

The verification activities along the project lifecycle require reports, which in the case of OHB are based on information coming from DOORS as well as standard office tools where data is not interlinked. The risk of inconsistency due to the usage of different sources of information exists here as well. Moreover, the elaboration of reports based on the evaluation of sets of documents takes a lot of time. This was identified by the LSIs as one of the main problems to solve in [2].

In most cases, these cross-referenced and consolidated reports do not exist yet as currently the information is spread over different document and tools. The definition of the targeted reports has dedicated a huge effort to identifying nice-to-have/desired capabilities instead of replicating current mechanisms (which have not always proven to be efficient).

For such capability, MBSE4AIV will automatically generate these reports consistently with implicit cross-data verification to increase the overall productivity of the project team. Thus, the MBSE4AIV system will also help in the evaluation of the reports as support to decision-making of the validation campaign.

The Assembly, Integration and Verification workflow, as implemented at OHB, covers the activities starting with the definition of the verification within the Verification Plan to different branches:

- Verification Control Documents closing the Verification activities
- Integration and Test procedures and Test reports closing the Assembly, Integration and Test activities and providing also inputs to the Verification activities.

All the information to be handled by the MBSE4AIV system needs to be compliant with the conceptual model defined in the project describing the AIV semantics. The model aims to constitute the single source of truth for the AIV domain of expertise. Although the authoring processes of such data are outside the scope of the MBSE4AIV system; the resulting data shall be able to be exchanged through the system and/or used to generate automatically the required reports based on which, decisions will be taken.

2 Modelling Phase

Parallel to this project, a Space System Ontology (SSO) [2], led by ESA is being modelled by a consortium of the main LSI (including OHB), ESA and industry involved in MBSE domain (including RHEA). That ontology development is coordinated by the so-called "**O**verall **S**emantic **Mo**delling for **S**ystem **E**ngineering" Governance (OSMoSE) for ensuring semantic interoperability between all partners as described in [3]. As the Object-Role Modelling (ORM) [4] through the NORMA Tool [5] is being used for the Ontology development, the conceptualization of the AIV Reporting semantics has applied the same methodology and tools to facilitate the integration of the resulting conceptual data model with the Space System Ontology.

However, AIV is not a new domain and has been applied for decades. Thus, related expertise and knowledge has already been described into existing ECSS standards, handbooks (HB) and technical notes (TN), explaining what happens during the AIV phases. Furthermore, as already stated before, some years ago an exercise at European level was performed to try to harmonize the needs for AIV and how it has been used by Industry in the so called Test Report Standard (TSR) [2].

As a consequence the conceptual modelling exercise of AIV was performed in two separate ways to define and align; on one side, the knowledge already captured by the TRS data model on a top-down approach, and on the other, the updated practical needs of the industry which have to be demonstrated with real examples (in a bottom-up manner). These two opposite approaches were merged in a single conceptual model via agile meetings with ESA and AIV experts until an agreement was reached and no more concerns or problems were raised.

In the top-down way modelling, the TRS was transformed from the UML/OWL formats into the ORM conceptual methodology which has advantages over the restrictiveness of OWL/UML on the definition of relationships and expressiveness of the concepts and entities, among other things.

During this exercise, several issues and doubts (see Fig. 1) were raised due to the following reasons:

- Semantics were not covered. This means, definitions of certain terms (e.g. *TestResultSynthesis* in the left side of the Figure) and concepts were not further specified in the ontology.
- There were inconsistencies in the ontologies:
  - between the formats representations of UML and OWL for the same ontology.
  - between the TSR ontology and what other standards, handbooks or technical notes required. For example, the concept of *System Operation Baseline* on the top right part of the Fig. 1 below, used in TSR, is defined with different semantic in another ESA conceptual model.

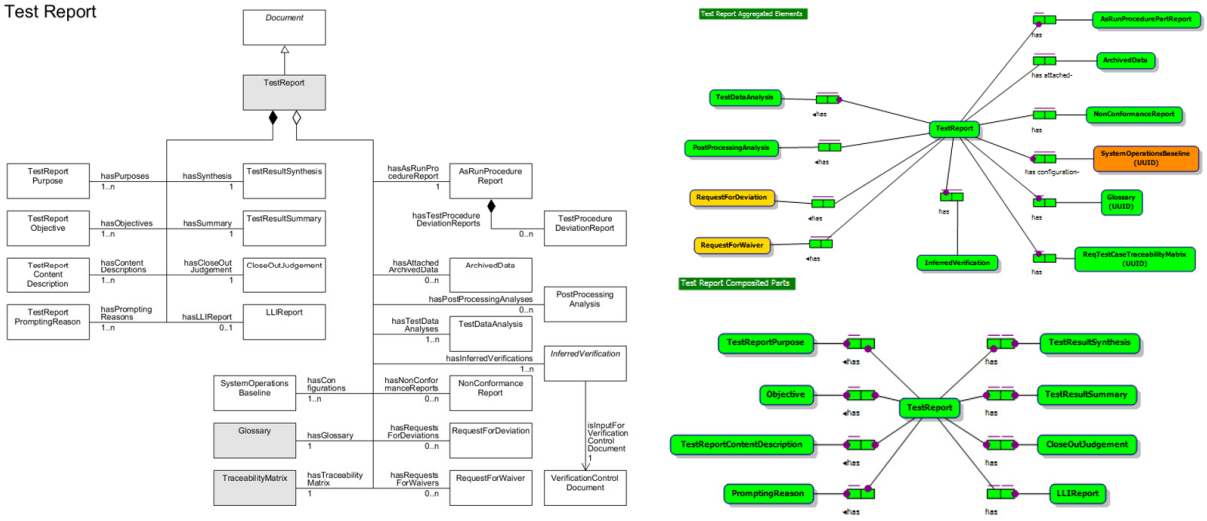

Fig. 1. AIV Model Part Re-engineering comparison between the TSR in UML and the achieved ORM model

These problems were mostly solved by holding discussions with experts and by looking for information on other standards or already modelled SSO Domains.

With this initial model, a representation of the status agreed at the time the TSR was defined, is achieved. However, in 5 years (time that has passed since the TSR was finished), AIV methods and needs have changed based on lessons-learned. Therefore, it is important to attain completeness by going the other way around. Starting with the created ORM model, we did the bottom-up approach in which we tried to populate the concepts from practical use cases OHB AIV experts require during AIV campaigns by:

- Mapping the information into already existing modelled concepts.
- When mapping was not possible, adding them into the NORMA model flagging them with specific group to define the source of information. One example of such needed enhancement can be seen in Fig. 2 below, where the information about Anomalies detected during Tests was basic on the TSR, while in real uses, the flow of the anomalies is of highest priority.

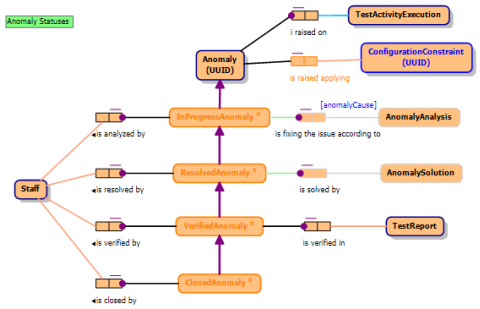

Fig. 2. Anomaly Addon to AIV Model

Once the model was stable, the mappings and differences between the concepts coming from AIV OHB Experts and information from the test report standard were discussed with the agency to see if old concepts, constraints, relationships could be removed, disabled or merged into a single model. The obtained model is composed by more than 250 concepts, 600 relationships and 100 validation constraints including minimum references to concepts/terms owned in other domains of expertise.

One of the biggest issues found during the modelling exercise was the change of paradigm in the target purpose of the information. When TSR was performed, most of the information was document centric. This means, most of the knowledge was not only persisted, but referenced as a whole document. That's why the internal semantic of the document was not defined, because users processed manually the referenced document. Nowadays, with MBSE and digitalization efforts, documents are just a possible view of the knowledge, but not the target in itself. As a consequence, all the semantics which want to be reflected in those documents need to be defined and also the relationships between those. A complete description of the modelling phase and the achieved conceptual model can be found in [6].

3 Designing Phase

Having a clear, complete and consistent data model which supports all AIV processes is good but it is not enough by itself. With it data can be persisted, but we do not live in an isolated world. So, in order to ensure the AIV information can be disseminated between the customer-prime-supplier chain with complete, understandable and comprehensive data, exchange interface(s) will need to be defined based on the conceptual data model explained before.

Levels of the exchange interfaces will also be impacted by security constraints and the level of granularity desired and agreed between parties. As a consequence, a generic interface with the minimum conceptually meaningful information (i.e. shallow exchange) has been defined. On top of it, another (i.e. the native one) has also been created to enable complete (i.e. deep) exchange including any referenced data which may belong to another Universe of Discourse (UoD) (e.g. a Test Activity references an Equipment defined as part of the Functional Architecture in the Thermal domain)

The provision of these interfaces is developed on top of a generic hub (i.e. MBSE Hub) also developed for ESA together with European Industry, as part of a separate activity, to exchange all space related knowledge as described in [7]. This hub has been designed following a microservice architecture.

The MBSE4AIV system will contribute to the Hub with dedicated microservices handling and processing the information according to the defined conceptual model using specific AIV Interfaces and generating the defined reports per identified Use Cases. Furthermore, it will also develop the Web User Interfaces for users to view and interact with the Hub in an extensible manner based on domain of expertise.

3.1 Model Based System Engineering Hub

The plan to overcome the challenges identified in the introduction of this paper is to develop a centralised Hub (at the centre of the System Factory concept) that provides interoperability between the different stakeholders and tools.

The Hub is foreseen to be used by different levels and disciplines of engineering, as well as throughout different phases of a space project, and it shall allow stakeholders to exchange information using a common vocabulary described in the SSO. In addition to providing interoperability, the hub needs to support auxiliary processes, such as authentication, authorisation, configuration control and enforcing correct workflows. The concise set of user requirements put forward by the Agency includes such complexities such as data version comparison, branching, federated deployments of connected Hubs and data validation against the SSO.

The exchange of engineering information from source authoring tools is achieved in several ways. The biggest flexibility is achieved by the creation of specific adapters for the particular tool. The adapters, on one hand, interface with the tool or the exchange file on the user's workstation; and on the other, with the MBSE Hub through its native API.

This hub-and-spokes model was successfully demonstrated to work in a related activity called Digital Engineering Hub Pathfinder (DEHP) [8], performed by RHEA, OHB, Astos and Open Engineering, in which the adapters were built for various industry standard 3rd party modeling and simulation tools, and used the server of the COMET Server [9] tool as the centralized repository to exchange with.

The power of independent adapters is such that they can provide segregated, maintainable capabilities independent from the developments of the MBSE Hub itself. DEHP adapters showed that they are able to operate on a common core architecture, have ability to do partial transfers, perform complex mapping, inspect impact on the target model and maintain exchange history.

The second way to exchange data is through uploading the information artifacts (complete files) directly to the MBSE Hub, together with some standardized mapping instructions. This would be appropriate for instance for

JSON, CSV, ReqIF or certain XML-based formats. The Hub architecture foresees a pluggable infrastructure for conversion plugins that would automatically detect the appropriate format and carry out the transformation to the native Hub model.

The deployment of the MBSE Hub would see it installed on a server alongside a message broker component. The message broker will facilitate the full exchange of engineering information with other federated instances of the Hub, potentially installed on other remote infrastructures in a push fashion. When a secure connection is established, a pipeline through a message queue is created that serves to create and update instances on the secondary site. Once any number of objects are shared, their ancestry is persisted and any updates on the master are instantly propagated to the share endpoints. An exchange loop is achieved by creating the connection in the same manner as the other Hub. To ensure the security of the federation, all message queues would be protected by certificates issued by a central authority.

This architecture design is a reengineering effort of the existing CIP tool, which is a software tool to support exchange of requirements, design and verification information, and even using some of the existing microservices. The Hub follows the same principles but with the application of newer technology stacks (e.g. Kubernetes, Keycloack, Blazor) and the change of the underlying engineering data model which now will be SSO-based.

3.2 AIV Extensions to the Hub

Using the Hub as the core framework, the MBSE 4 AIV System benefits from all the default capabilities to which those specific to the AIV related processes will be added.
The specific processes to be supported in the MBSE 4 AIV system focuses on two aspects:

- AIV Data Related Processes – These processes involve the management and persistence of the AIV data according to the data model (including Test Cases, Procedures, Verification Results, Requirements) and handling the cross references and consistency against the data owned in other domains of expertise (e.g. OPS Procedures, Functional Definition, etc.). As part of such processes, the definition and implementation of exchange interfaces providing CRUD and validation capabilities is also involved.
- AIV Report Generation and Display Processes – These processes involve the retrieval of all associated AIV and engineering data from the Hub data storage, the instantiation of the specific type of AIV report data model, the general implementation of Human-Machine Interfaces for the different types of reports and the capability of exporting the populated report data model.

The required processes are defined as stateless microservices exposing REST interfaces to be deployed in the same server as the hub ones. Furthermore, each microservice is configured in a docker image to make it easier to be deployed in a Kubernetes cluster, and thus scale the whole system.

Parallel to the pure data management and exchange, any application needs to involve the users in a graphical manner. However, the project developing the Hub will not provide display capabilities in its first version as it will be focused on an API/Command Line Interface (CLI) access. However, for the AIV experts, user interfaces are required to access the Hub features, to select the configuration criteria of the AIV reports and to see the results in plots, figures or traceability matrixes. So, although not strictly needed by AIV processes, the MBSE4AIV system has implemented default UIs for the overall management of the hub have also been created such as user access, project management, search displays, etc. These displays will become a shared contribution to the Hub. On one side, the MBSE Hub will have a minimum UI to start gauging users' involvement. On the other, users will have a reduced the learning curve of the overall framework and feedback will be received sooner than foreseen so modifications will be done in a more agile manner.

4 Implementation Phase

The implementation of a system based on model Base System Engineering has to demonstrate why MBSE is useful and more efficient than a normal development project. This means, implementation and deployment need to have a higher level of automatic code generation.

Both the Hub and the MBSE4AIV systems exploit this idea with the use of the ORM conceptual model of the engineering for the generation of the physical data model, the exchange format schema, the data transfer objects and the REST interface endpoints.

4.1 Automatic Interface Definition

The implementation phase needs to feed from the work performed during the modelling phase to achieve the acceptance of the AIV model in harmonization with the SSO. The tool used during modelling, NORMA, already allows to generate XML or SQL code automatically. However, the generated outputs are not completely practical (e.g. too many intermediate tables) and need manual modifications. Furthermore, as the system communication is designed through microservices communicating through REST, the exchange format is preferred to be JSON as it

is designed specifically for data interchange and its encoding is more compact than XML as summarized in [10]. Thus, a new code generator had to be created.

RHEA is developing the Kalliope [11] framework as an open-source library which allows to write and process ORM files (according to NORMA specification) and is used for the creation of the automatic generation of the corresponding DTOs, POJOS, GraphQL schema and the JSON representation. The ORM files are ingested by Kalliope, an in-memory representation is created and through target-specific template files, the output files are generated. This is shown in Fig. 3 below, for the generation of JSON code.

Fig. 3. Code Generation Process

In order to produce this output, we have made use of a NORMA mechanism called “custom-properties” which allow you to define extra metadata to concepts and facts (i.e. relationships).

Thus, the NORMA custom properties are used to target how we want these logical and physical levels of the model to be represented as follows:

- Identifying where a relationship implies aggregation or composition, This is important not only for the persistence and validation processes, but also for the exchange schema because composition will imply in JSON, all the composite information needs to be provided at the same time; while aggregation implies a consistency check needs to be performed.
- Using the definition of the concept playing the owner role of a relationship which is used to identify who host the reference to the other entity type,
- Explicit flagging the abstraction in a concept used for the instantiation of the generated classes.

This same mechanism is used in the code generated from the SSO, OPS and RAMs domains that the Hub Framework needs to support. Therefore, any reference to the concepts from AIV to any of the other domains can be ensured because, not only the ORM files are integrated at conceptual level; but all the information will be hosted in the same database and generated using the same mechanism for the logical (i.e. DTO, POJO) and physical levels (i.e. SQL and JSON)

However, for the moment, there are interfaces that have been created only from CRUD aspects of the data. One fundamental part of the project is the aggregation of the stored information and the derivation of AIV reports from which knowledge can be elicit and decisions made. The user requests for this type of complex request have still been defined manually. However, they use the JSON types generated with Kalliope. It is important to remark this is not due to technological issues but for practical reasons considering benefit vs. effort and cost ratio.

4.2 Report Engine

The generation of the AIV reports is, in essence, collecting all needed AIV data according to some configuration criteria (e.g. project, type of equipment, dates of the tests, etc) which are defined in detail in the use case scenarios to be covered and then populating the AIV model data creating instances of the specific type of reports (some of which are depicted in Fig. 4 below), so later it can be displayed to the user or downloaded in native format.

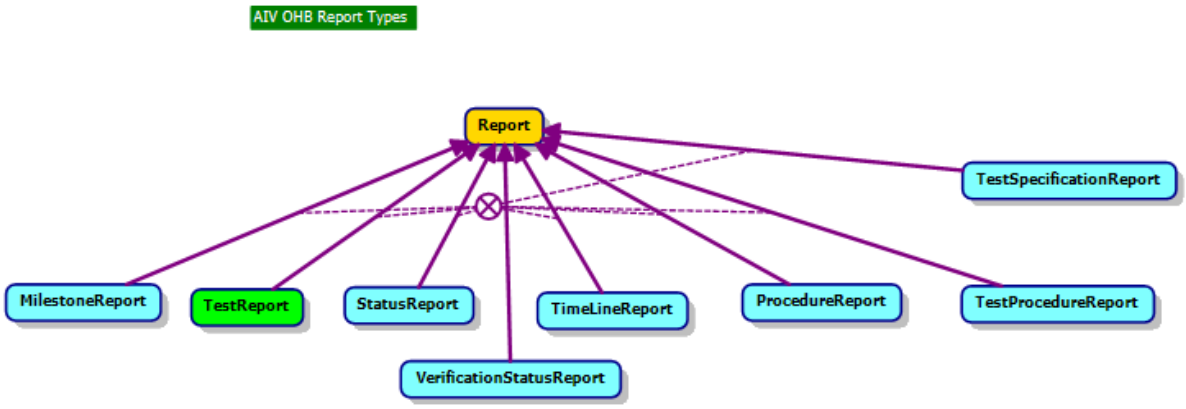

Fig. 4. Report Hierarchical Types

The report engine, outside of the Displays development, is purely specific to AIV. However, the development will be performed in a generic manner, so new types of reports (even if into AIV based) can be supported without code generation.

The report generation will be completely dynamic based on API calls instead of creating a custom mechanism only for the currently supported types. This decision has been taken because modifications are foreseen in the future, and new types of reports are known to have to be added in future updates.,

The technical solution for this problem is built around allowing the back end to request the metadata of the configuration criteria applicable for each type of report from the DTOs, the validation of the provided values against that same metadata already retrieved, and then performing the normal retrieval of the data from the underlying database for its aggregation. The aggregation is, in essence, the creation of the report instance which is composed of references to the retrieved elements and derived information (e.g. calculations, subsets, etc).

Having the instance of the desired report type, the last step of the report generation is the response provision to the user. Three possible options have been identified, and thus, will be implemented. On one side, if the results are going to be processed by another tool, the information will be provided in the system native format (i.e. JSON). On the other, when the report is going to be checked by a user, it will be either displayed in a dedicated User Interface or formatted as a PDF (containing the same graphics as in the UI) for offline review.

4.3 Displays

Convincing users to change their way of operating, making use of a new tool instead of what they are used to, is one of the most difficult tasks a new system has to overcome in order to be successful. MBSE4AIV is no different.

The design of the MBSE4AIV user displays is based on OHB user expectations and simple mock-ups. However, a minimalistic aesthetic has been implemented to offer an ergonomic, user-friendly and practical experience which does not confuse users following guidelines described in [12].

The displays are web-based and developed using Blazor [13] technology as most of the system is developed in .NET, and C#. However, it being a service-based system, there is no limitation for other displays/technologies to be created/used in the future.

The displays implemented for MBSE4AIV are structured as follows:

1. General Hub functionality → These displays are generic for any hub-based system and provide the access to the general features, including Access to the system, User Data Management, User Activity Tracking and Project Management. All these functions are accessible from a general sidebar menu as shown in screenshot below. The navigation of the system works through hyperlinking and will load, by default, the different information in the main area of the page.

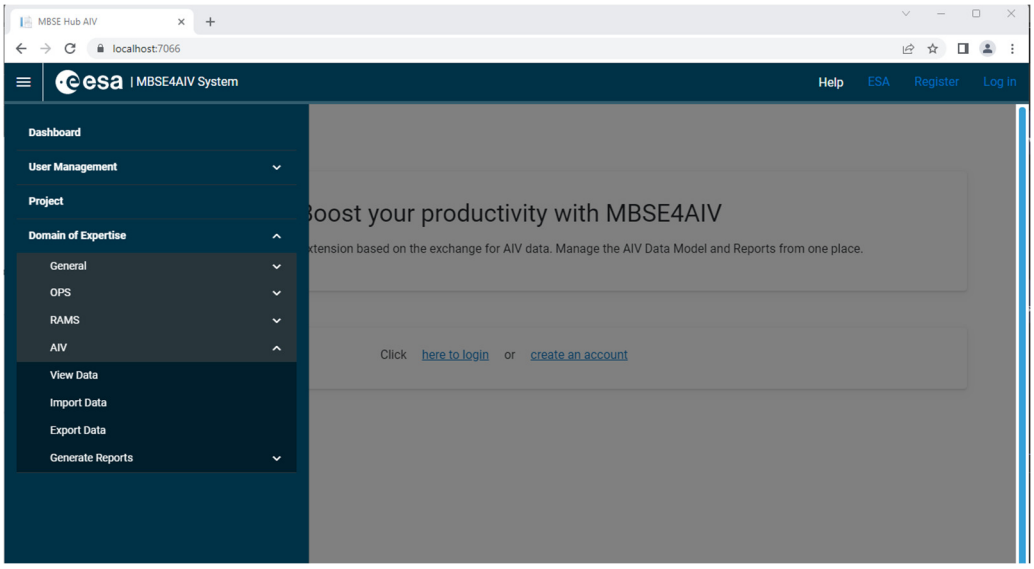

Fig. 5. MBSE4AIV General Side-bar menu on the Dashboard UI

2. MBSE44AIV Specific functionality → Those features which are only applicable to the specific domain of expertise. In the case of AIV these are the Visualization and search of AIV related data (e.g. Test Campaigns, Test Cases, Anomalies, Verification Analysis, etc), the AIV Report Requesting and Visualization and the Import and Export of AIV data in JSON format into the system.

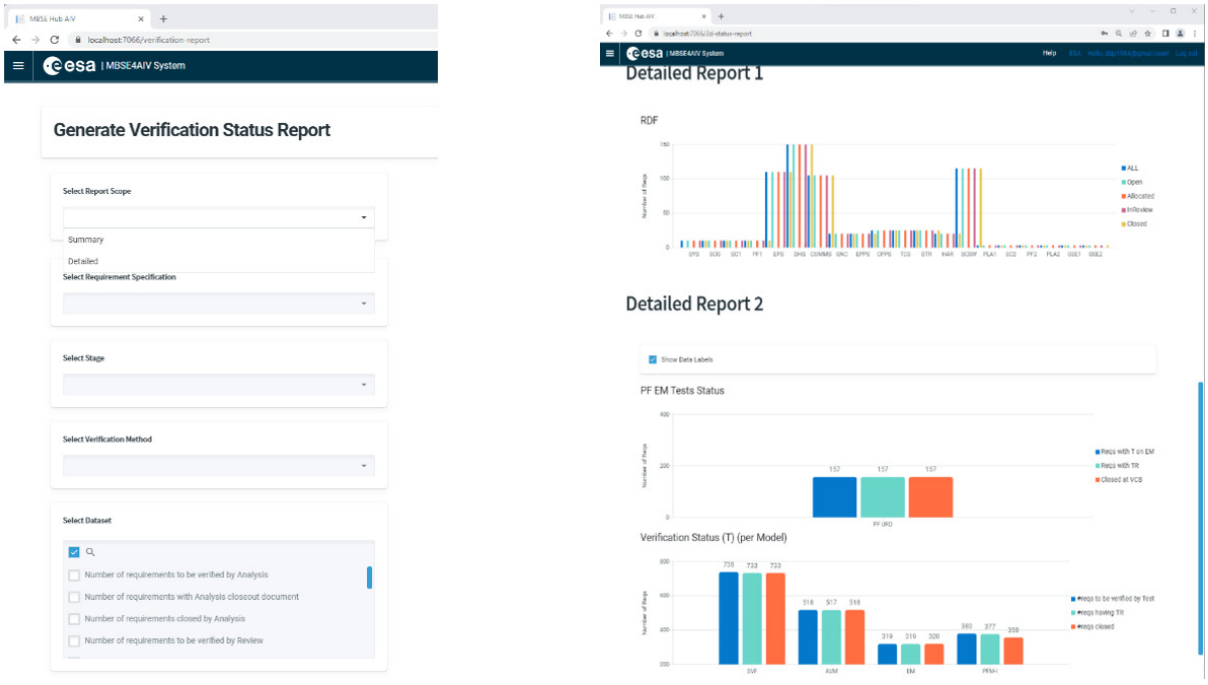

Fig. 6. MBSE4AIV Report Displays

In the left part of Fig. 6, the form for the specification of the criteria for generating a verification report is displayed. Each of the form fields are criteria fields of the underlying data model, upon user population/selection of those fields, the system requests the instantiation of the specific report and provides the resulted population in the JSON format that later is used to render the form. On the right-hand side, we can see the detailed report containing the verification status of different AIV models.

This development is still a work in progress because there is a need to add extra capabilities such as plots, rich text fields, export capabilities integrated. Furthermore, the display on other devices (e.g. tablets and mobiles) also needs to be verified.

Once a minimum meaningful part of the User Interface is stable (e.g. a complete end to end type of report is generated and displayed), the system will be released to OHB AIV Experts. The release will be done in a staging environment so they can start interacting with and feedback can be received as soon as possible and modifications can be implemented to increase the future acceptance and use of the system.

5 Conclusions

The development of the MBSE4IV system, which is foreseen to be completed in Q3 of 2023, has already managed to define a single source of truth for the AIV domain in the shape of a formal conceptual data model, while automating the generation of AIV reports integrating data from multiple sources which will be displayed and exchanged graphically.

Nevertheless, the development depends completely on three aspects: the underlying conceptual model, the integration and synergy with the Hub system and the development of engaging User Interfaces. These three aspects need to be continuously monitored and balanced out because the SSO is under design and governance, the Hub under development and the User Interfaces require the involvement of different user roles. This is not an easy task and creates schedule and dependencies issues.

Following a practical and proactive role is proving to be key. For the model integration with SSO and the different UoDs interacting with AIV, the involvement of the SSO governance team and transparency between different companies developing different UoD is of utmost importance. The first step for reducing risks and producing a meaningful and efficient knowledge model has been the organization of a Workshop between industry and agency to ensure model homogenization and decision making in the common concepts and relationship.

The use of Kalliope and template-based code generators from the model, has already proved to be really beneficial as changes in the model implies modification of the persistent database schema, JSON exchange format and in memory object representation. Changes are foreseen to be a constant in the close future while new UoD are integrated and SSO is officially approved. Thus, the investment on the development of code generators will be continued in the future.

Consequently, as more data types are going to have to be supported, and their definitions may change, the user interfaces also need to be generated dynamically.

These realizations seem to target higher effort and cost than expected at the start of the project. However, the better the models and the code generators, the less manual and maintenance work will be required later on. These benefits have already been seen, but the implications have to be understood by future projects. That's why the synergy with the Hub development is overarching because the development of code generators and model alignment effort is shared between the teams as results will benefit everybody.

Lastly, the end-user involvement since the start and the possibility of using real data for the validation is foreseen to provide beneficial for the acceptance of the new system into the AIV campaigns of OHB. However, it is still unknown and will have to develop trainings and marketing campaigns for users (not only OHB ones) to get to know the new system and include it in their toolset.

6 Bibliography

[1] ECSS, ECSS-E-ST-10-02C, 2018.

[2] ESA, "Test Report Standard," 2017 September. [Online]. Available: https://indico.esa.int/event/201/contributions/1864/attachments/1599/1846/1000b_Test_Report.pdf . [Accessed January 2023].

[3] S. Valera, Q. Wijnands, G. Garcia, J. Bernaudin and L. Laborde, "OSMOSE Workshop," in *Model Based System Engineering (MBSE) 2022*, Toulouse, 2022.

[4] T. Halpin, "Objet-Role Modelling," [Online]. Available: http://www.orm.net/. [Accessed Jan 2023].

[5] M. Curland, "NORMA - The software," ORM Solutions, [Online]. Available: https://github.com/ormsolutions/NORMA. [Accessed Jan 2023].

[6] N. Salor Moral and P. Beltrami, "Enhancing the MBSE-Hub for AIV Reporting Needs," in *Model Based Enginnering Sytem (MBSE)2022*, Toulouse, 2022.

[7] A. Vorobiev, K. Tiensuu, S. Gerene, S. Jahnke, L. Bitetti and H. De Koning, "Model Based System Engineering Hub," in *Model Based System Engineering (MBSE)2022*, Toulouse, 2022.

[8] M. Verhoef, S. Gerene , A. Vorobiev, N. Smiechowski, S. Jahnke, S. Weikert and S. Paquay, "Digital Engineering Hub Pathfinder,," in *CM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, Fukuoka, Japan, 2021.

[9] RHEA Group, "COMET Tool - rel9.3," RHEA, 2022 Sep. [Online]. Available: https://products.rheagroup.com/comet. [Accessed January 2023].

[10] Geekboots, "JSON with advantage and disadvantage," [Online]. Available: https://www.geekboots.com/story/json-with-advantage-and-disadvantage. [Accessed Jan 2023].

[11] RHEA Group, "Kalliope Library," November 2022. [Online]. Available: https://github.com/RHEAGROUP/Kalliope. [Accessed January 2023].

[12] R. R. D. C. a. C. N. A. Cooper, About Face 3: The Essentials of Interaction Design, JOHN WILEY & SONS INC, 2014.

[13] Microsoft Corporation, "Blazor Repository," 2018. [Online]. Available: https://github.com/dotnet/aspnetcore/tree/main/src/Components.