MBSE in AIT and Operations of MOVE-II and its Impact on the MBSE2DL Activity and the MOVE-III Mission

Florian Schummer,
Maximillian Hyba
Chair of Astronautics
Technical University of Munich
Boltzmannstr. 15
85748 Garching, GERMANY
0049-89-289-16007
f.schummer@tum.de

Armin Müller,
Björn Beyreuther
ScopeSET GmbH
Im Rothmoos 5
83730 Fischbachau
amueller@scopeset.de

Andreas Lindner
3DSE Management Consultants GmbH
Seidlstr. 18a
80335 Munich, Germany
a.lindner@3dse.de

Abstract— **This paper covers the lessons learned on applying MBSE in the MOVE-II mission, with a focus on Assembly-Integration-Testing and Operations of the spacecraft. It shows the states of implementing MBSE across the various project phases. Furthermore, it shows the implementation of these lessons learned on the MOVE-III mission and what has been learned based on this application. MOVE-III was initiated in 2019. Taking the input from both missions and the results of ten interviews with ESA specialists in the fields of AIT and MBSE, we progress to the aims of the "MBSE to DataLakes project". This new activity heads the primary objective of increasing efficiency of spacecraft AIT by employing MBSE, matching ESA's overall MBSE strategy.**

Set up as a public funded project, with the aim of hands-on education for future spacecraft engineers, the MOVE-II project followed the typical design stages of any spacecraft development. Combined with the lack of inherited liabilities and the limited complexity compared to larger spacecraft programs, this makes it a good reference project for MBSE implementation. While the spacecraft is in orbit since December 2018 and is being operated successfully since then, the focus of this paper is set on lessons learned in the implementation of MBSE throughout the design and qualification of the spacecraft, up to its daily operations in orbit over the past five years.

1. Stages of MBSE Implementation on the MOVE-II Project and Lessons Learned Thereof

Compared to traditional spacecraft, the limited size and fixed envelope of a CubeSat severely limit the necessary budget. As with many other space missions, MOVE-II was initiated with the target of hands-on education for university students, mainly in the physics, computer sciences and engineering curriculae. This goal shaped the whole development setup as well; subsystems for which sufficient expertise and interest could be gathered among the students were developed in house from scratch, such as the Attitude Determination and Control System, the structure and shape memory alloy based deployment mechanisms, transceivers in the UHF/VHF and S-Band range and all on-board software. The goal of hands on student education also greatly impacted the development team's setup [1]. As scientific payload the CubeSat carries quadro-junction solar cells, of which the degradation in the space environment over time shall be measured [2], [3].

With only a single full-time employed PhD candidate until late 2017, systems engineering and most management tasks were also performed by students. The amount of weekly active developers ranged from around 50 to over 100, with peak team size being reached during software-implementation and integration of the subsystems with each other. Due to the fact, that students were in general not awarded credits for their participation in the project, most of the team could only engage in the development in part-time, ranging from around 8-10 hours a week to up to full-time for students writing theses on specific aspects of the spacecraft or taking a semester off to commit to the project fully. An additional challenge was the high turn-over rate. While some students accompanied the project from end to end, the majority had a retention time of roughly one year. Thus, having set the stage, we will progress to the different phases of MBSE implementation during the project.

In March 2016, a formal systems engineering team was established on the project, two months from the Preliminary Design Review. Before that, the coordination of interfaces, requirements, budgets and system configuration were not treated as a group effort, lacking knowledge of formal methods and a high overhead for coordination. Being new to the project and tasked with leading systems engineering, we had no prior knowledge of MBSE tools, despite a well-founded education in SysML. [4]

Over the course of three years, various stages of MBSE implementation have been achieved and are presented in the following. [5]

Stage One, SysML with Whiteboards, Pen and Paper

Hence during this first stage of our journey to MBSE, SysML was mainly employed to structure discussions and dialogues, with diagrams being drawn by hand during meetings and used to protocol the outcome of discussions. The model therefore consisted of a loose collection of diagrams, although semantically accurate and consistent to each other. While still being far from Model-Based Systems Engineering, we found the approach quite useful in the beginning. The overhead is minimal, the low complexity of diagrams allowed engineers across the field to understand them and participate in the discussions on specific features or the overall architecture. However we found the approach lacking in availability of information. As the diagrams were often stored as hand-drawings in distributed notebooks (Microsoft OneNote), it could become difficult to find a specific drawing. In case it was actually drawn on paper, the likelihood of losing the information increased. While enhanced with multiple other methods over the time, live-drawing of SysML diagrams to structure discussions was a method frequently employed

through the project and is still done for major troubleshootings on orbit in the form of activity diagrams for planning of the next analysis-steps.

Stage Two, A Collection of Diagrams Maintained in Microsoft Visio

For the documentation of the Critical Design Review in September 2016, the diagrams were condensed into a uniform view, which started the second stage in our journey to MBSE. Microsoft Visio was employed as tool, which can be attributed to the lack of availability of proper SysML tools such as MagicDraw. The same tooling was used to define the software-architecture of the spacecraft, where in a single diagram all data-exchange on a subsystem-level, as well as within the Command and Data Handling subsystem, were defined. While not providing enough detail to deduct the exact interface definitions from the diagram, the singular view of any software running on any hardware aboard the spacecraft was very popular among the developers, as it provided a good overview of the spacecraft. It was used for regular updates between developers, showing which features were already implemented, or whom to contact to coordinate a specific interface. This diagram was the basis for most of the implementation of the MOVE-II spacecraft software. While it lacked details, which were supplemented by a detailed software-specification, the main features of its success were its simplicity and the frequently published updates. Especially the regular updates caused a high confidence of all team members that the diagram shows the real state of the design. Our lessons learned from this second phase of our journey to MBSE were:

- Only add as much detail to your MBSE model as your team is able to maintain. A model that is that is only 95% up-to-date leads to frustrated developers and a starkly decreasing interest in the information contained in the model.
- Actively advertise the model and the information it contains. People new to the team often lacked overview of the whole spacecraft, and had a eye opening moment when they were shown the diagram of the software architecture.
- When in doubt, investigate. There is always a chance that decisions have been made you are not aware of, that make reality differ from the model. Keeping consistency is key however.

Summing up the second stage - a variety of aspects were modelled, mainly in block definition and internal block diagrams, but also employing activity diagrams and state machines to model specific software behavior.

Stage Three, MBSE in Assembly Integration Testing

The next stage of our implementation of MBSE involves a self-written tool. In the beginning of the integration campaign, information about probable anomalies often was shared using the messaging app Slack. Additionally, formal reports in a redmine ticketing system were required, once an anomaly was confirmed. Sharing information in two different systems on the same topic caused a lot of confusion. Therefore, a Slack application was developed, that created redmine-tickets out of threads once someone reacted with a certain emoji to the thread's original message. The app updates the redmine ticket with every new reply in the thread in real-time, enabling much faster communication, while at the same time ensuring the formality of a ticketing process, where issues are resolved and closed is upheld.

Though not being part of the formal specification of MBSE provided by INCOSE, which states that MBSE is "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases" [6, p.15], the increase in efficiency as perceived by both developers and management, later triggered us to investigate further into increasing efficiency during assembly integration and testing, via better tooling for communication and reducing the necessary effort to get a holistic view on a topic. A disadvantage of the approach is the necessity of connecting the discussion on a probable cause of failure with the system's architecture by hand. Expert knowledge was key to investigate on how an observed anomaly can be linked to its root-cause. Having a good understanding of the interfaces between various components enabled us to build efficient anomaly resolution boards, which would usually consist of a systems engineer, software engineer and experts from each potentially involved component.

Our perception here is that the majority of testing and implementing team members have a lack of knowledge with regards to the overall context of a problem they face. Without understanding the context of the observed anomaly, they were unable to self-sufficiently perform a root-cause analysis. They specifically lack the following knowledge:

- What are sensible next steps in the analysis of the perceived anomaly?
- How can I find possible root-causes for the perceived anomaly?
- How can I rule out unlikely root-causes, or those that were proven to be wrong?
- Where can I find additional information on the hardware/software under investigation?

This caused a high dependency on the duo of software- and systems engineer to help resolving those anomalies, making both crucial personnel for the mission. Their unique understanding of the overall architecture meant they were able to "connect the dots" and lead most anomaly resolution efforts. This is not acceptable from a (risk-)management perspective. Which provides the most important lesson learned of this paper regarding the employment of MBSE: A detailed MBSE model might be able to reduce dependency on key-personnel in later project phases such as Assembly Integration Testing or Operations. Such a model would have to include

- commands and data being exchanged aboard the spacecraft and between spacecraft and ground
- connections to the respective repositories to check up on the implementation of specific components (for software as well as electronics)
- connections to the spacecraft's telemetry database, showing which telemetry is taking which path to the operator's/tester's viewpoint.

We found the last part to be crucial: most of the work done by software- and systems engineer revolved around ruling out possible causes for a specific anomaly. This was most often achieved by investigating what other telemetry, apart form the reported anomaly seemed affected and which closely connected telemetry (i.e. processed by the same components) was unaffected, allowing to rule out possible causes for the anomaly.

Stage Four, Employing MBSE for Operations

Entering the next stage of our journey to MBSE, a model of the whole spacecraft's behavior and structure was built prior to the launch. Models of the ground station and operations backend were built additionally, creating a holistic view of the mission for operators and ground personnel. While the model did not contain any of the connections to other data sources mentioned above, commands and telemetry were modelled end-to-end from originating sensors on the spacecraft down to the operations database.

Especially the ground station's model proved to be very useful in the time leading up to the launch and early operations. As overpasses of the spacecraft usually come in pairs or triples, separated by roughly 80 minutes, errors observed on the first pass could often be methodically traced back to their root cause and fixed before the next overpass, employing logical deduction based on the model of the ground station.

Examples of the workflow and excerpts from the model can be found in the full paper.

Apart from the ground station's model, the spacecraft's model was often used as a focal point for discussions on specific observations and necessary next steps. Facing the same timeline of roughly 80 minutes between two overpasses, efficient analysis and deduction of next steps would often have to be done within 30-40 minutes, to leave sufficient time to prepare and test the next commands.

Having all relevant diagrams hung up in printed form in the mission control center proved a valuable addition and greatly helped in the analysis (similar to a "war room" used in task force teams).

Our lessons learned for stage four are that the approach worked very well. It saved us quite some time and often enabled quick but informed decision making during the early days of Mission Operations.

2. Impact of Lessons Learned from MOVE-II on the MOVE-III Project and the MBSE2DL Project

During stage four of the previous section, MagicDraw was employed for the first time as a full-scale modelling tool. Due to the success in the later stages of MOVE-II with MBSE, the decision was made to fully embrace MBSE for the MOVE-III mission. The model's focus should be on structural diagrams as well as requirement diagrams, to enable complete traceability of requirements and help to define interfaces by deciding on required telemetry and commands. Compared to MOVE-II, a whole team of students with modelling capability, builds and maintains the model, which right now covers the structure of the whole spacecraft but omits the ground segment of the mission as no decisions on the ground segment have been made yet. At the same time, the very positive experiences made with the model of MOVE-II depicting telemetry and commands end-to-end form the basis for the MBSE2DL project, where a dedicated software to connect telemetry from the spacecraft's database with the SysML Model in form of a Semantic DataLake is under development. Drawing on the experiences of stage three and four of the previous chapter, the system aims at enabling holistic analyses of the model and data by employing graph-querying techniques.

3. The MBSE2DL Project

Selected as a demonstrator implementation by ESA's Open Space Innovation campaign "Model-based System Engineering: from documents to models", the project officially started in January 2021. Beyond the scope of traditional MBSE tools, this study aims to combine various MBSE related data in a Semantic DataLake. Inspired by the Big Data Europe initiative (cf. [7]), its main objectives are:

- Provide a DataLake as source of a holistic view, which allows access to all relevant data in an un-opinionated fashion, i.e. the semantics are not hardcoded as it is in an MBSE tool typically, but rely on a set of extensible ontology definitions.
- The data defined and maintained in typical MBSE tools is usually limited by the underlying opinionated language or meta-model and the level of detail provided by the model. However, other relevant sources of information exist in a variety of other tools, e.g. Excel, a set of Analysis tools for specific use cases, simulation and test databases. In contrast to the state of the art, where such information is integrated with point-to-point solutions, the DataLake provides a holistic view and a general access layer to all of the accumulated data and its documentation.
- This in turn allows to perform activities such as
  - Import and map a variety of input data based on an extendable Ontology (eventually to be aligned with the OSMoSE Space Ontology, cf. [8])
  - Integrated querying of all accumulated data
  - The (semi-automated) creation of traces between different artifacts, in particular between Blocks/Modules/Parts of the product structure and the related telemetry database/table
  - Dependency analysis between such artifacts using graph analysis
  - Assist engineers with the logical deduction of likely causes for anomalies with a seamless integration of the measured telemetry of the spacecraft to the originating SysML model

Figure 1 outlines the main functions and data flow in the envisaged DataLake platform.

The scope of this activity is not to replace existing solutions and tools, but to provide a scalable analysis platform by applying both the lessons learned from the MOVE-II project as well as established frameworks and building blocks from BigData analysis. The extendibility aspects cover

- the adaption for additional data to be imported (e.g. other flavors of SysML, and telemetry data) as well as new artifacts such as anonymized data from a ticket system or other communication platform
- the development of new MachineLearning approaches based on established and scalable interfaces to the DataLake
- the integration of dedicated Web UIs for efficient access

The platform will be suited for domain experts, who will be able to setup different kinds of imports and analysis without having to worry about the technical infrastructure, as well as for end-users who will only browse and access the analysis result data being stored on the platform through dedicated web interfaces. In Phase 1 of the MBSE2DL project, a group of domain experts from ESA participated in a set of interviews where input on both the current status of MBSE as well as future needs was requested. The majority of the experts confirmed the given objectives of the project, in particular in regards to improving the efficiency of the AIT process in conjunction with MBSE.

In addition, most of the interviewees stated signigicant in-

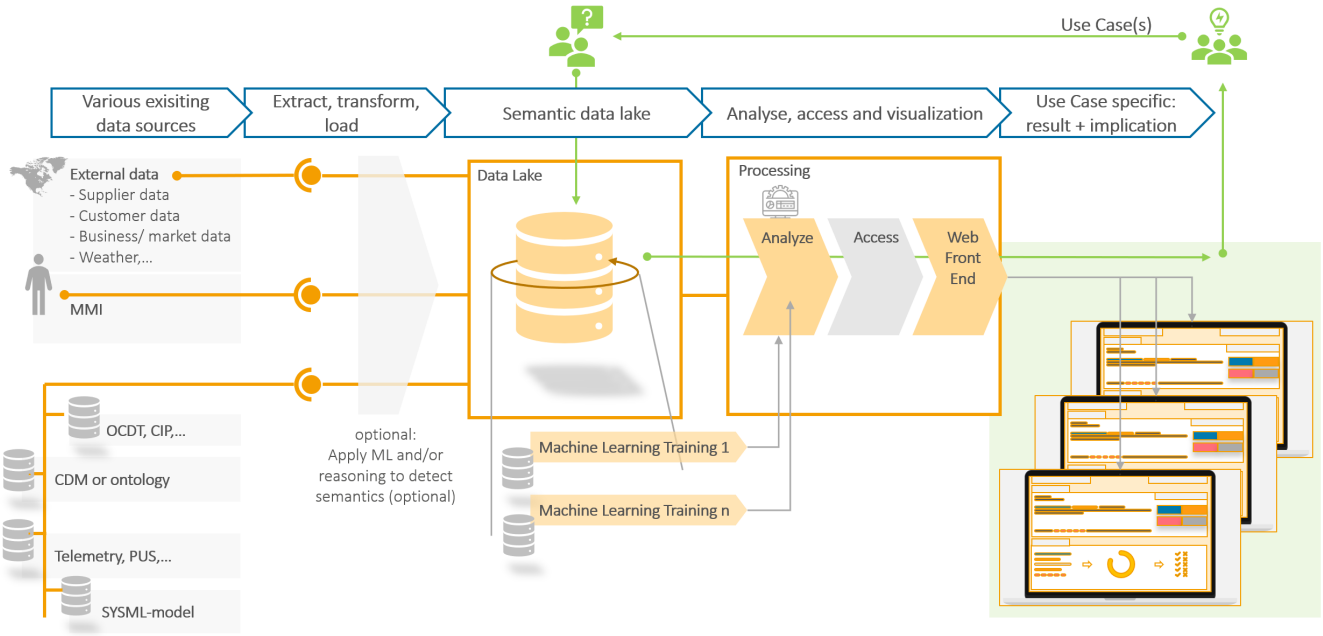

Figure 1. High Level Architecture of the MBSE2DL Project

terest in including aspects of requirements handling and tracing within the functional scope of the platform. Others highlighted some potential use cases in the fields of external collaboration / extended enterprise applications by including external customers and development partners, e.g. with extended sharing of models instead of documents. To put it bluntly: Today MBSE still has a clear acceptance problem, since advantages of a financial, technical, individual and organizational nature are not clearly visible and cannot be transferred into significant market advantages nor clear business cases. In order to establish MBSE successfully, an user centered and integrated system landscape must be created from the team level to the management, as well as from the first requirements to the retirement of a technical system. Technical feasibility of digital solutions has been demonstrated in limited fragments/ demonstrators in non-engineering areas. The presented Semantic DataLake platform will provide significant advantages in capabilities, time efficiency and labor cost by providing a single source and access point for information retrieval. It is the next necessary step to handle complex development projects and data structures like digital twins.

With further adaption and improvement of the concept, we expect new levels of collaboration and performance across engineering disciplines, business domains and customer involvement.

The MBSE2DL project will be finished in summer 2022, all results will be available under the ESA public license.

References

[1] M. Langer, N. Appel, M. Dziura, C. Fuchs, P. Günzel, J. Gutsmiedl, M. Losekamm, D. Meßmann, T. Pöschl, and C. Trinitis, *MOVE-II-der zweite Kleinsatellit der Technischen Universität München*. Deutsche Gesellschaft für Luft-und Raumfahrt-Lilienthal-Oberth eV, 2015.

[2] M. Rutzinger, L. Krempel, M. Salzberger, M. Buchner, A. Höhn, M. Kellner, K. Janzer, C. G. Zimmermann, and M. Langer, "On-orbit verification of space solar cells on the cubesat move-ii," in *2016 IEEE 43rd Photovoltaic Specialists Conference (PVSC)*, 2016, pp. 2605–2609.

[3] M. Langer, F. Schummer, N. Appel, T. Gruebler, K. Janzer, J. Kiesbye, L. Krempel, A. Lill, D. Messmann, S. Rueckerl *et al.*, "Move-ii-the munich orbital verification experiment ii," in *Proceedings of the 4th IAA Conference on University Satellite Missions & CubeSat Workshop, Rome, Italy*, 2017, pp. 4–7.

[4] Florian Schummer, Kim Steinkirchner, "Implementation of systems engineering methods in a running volunteer project at the example of move-ii," 2016.

[5] Faatz Niklas, "Investigation of the applicability of an artifact model in sysml using the cubesat move-ii," 2019.

[6] International Council on Systems Engineering, "Systems engineering vision 2020," 2007.

[7] BigDataEurope, "Big Data Europe Empowering Communities with Data Technologies." [Online]. Available: https://www.big-data-europe.eu/

[8] European Space Agency, "OSMoSE - Home ." [Online]. Available: https://mb4se.esa.int/OSMOSE_Main.html