Lessons learned in applying MBSE to the development of autonomous and highly secure nanosatellites

Hazel Jeffrey
Craft Prospect Ltd, Glasgow, UK
hazel@craftprospect.com

1. Introduction and Objectives

The key objectives that this paper aims to address are to capture the lessons learned and experience gained from applying Model Based Systems Engineering (MBSE) to two separate complex nanosatellite mission designs and to present the enablers and obstacles encountered when introducing MBSE within the organisation and to project stakeholders, specifically this paper will address objectives O-1, O-5 and T-1.

Craft Prospect has recently utilised Model Based Systems Engineering in the development of two vastly different mission designs, and also at different lifecycle stages for each mission. Firstly, MBSE was used to design and develop an AI-Enabled Quantum Key Distribution (QKD) mission and nanosatellite payload, known as Responsive Operations for Key Services (ROKS), as well as the respective ground segment architecture and requirements. The ROKS mission is comprised of a 6U nanosatellite with a mission objective to provide Low Earth Orbit QKD services to augment larger scale QKD efforts with the additional onboarding of an Artificial Intelligence (AI) backed context imager to detect and avoid cloud cover. As of June 2022, the ROKS payload is closing out system functional and environmental testing of the first flight model, with expectations to integrate the satellite bus in late 2022 and launch in 2023. This was the first project that Craft Prospect had undertaken primarily using MBSE for the full system development, hence, a number of challenges were encountered and many lessons learned.

The second mission in which Craft prospect used MBSE was the Phase A development of an assured Machine Learning (ML) algorithm for fire detection deployed onboard a nanosatellite. For this mission if the ML algorithm detected an active fire, the satellite would be autonomously triggered to broadcast an alert including the fire location, severity and confidence of detection, which is received by on-ground emergency services. In this project MBSE was used initially to develop the full system architecture, to refine the assurance requirements for the mission and ML algorithm, and finally to analyse the functional behaviour of the system. This final task was arguably the most crucial as it would identify any operational aspects of the system which could introduce significant errors in the fire detection algorithm, which would have otherwise gone unnoticed.

Considering the systems engineering effort on these projects, not only does the systems designer have the burden of implementing an emerging technology, such as QKD, but the introduction of AI leads to significant levels of autonomy within the mission architectures which challenges the existing established principles of satellite operation. Additionally, for onboard autonomous fire detection the impact of reporting a false positive or false negative result to the user is significant. A false positive alert may result in a waste of fire response resources, while a false negative could result in loss of life. Due to this increased complexity and risk associated with both mission concepts, the systems development would be incredibly complex using traditional manual and document driven systems engineering processes.

2. MBSE process development

For the ROKS project an initial survey of available MBSE tools was conducted. Enterprise Architect was selected as the MBSE tool of choice for ROKS for two particular reasons, firstly it appeared to be

relatively flexible in terms of the design process and tool usage. As some early scoping work had been performed for ROKS it was believed that the flexibility it offered by Enterprise Architect would allow a smooth transition from capability level considerations through to system and subsystem design. Secondly, it was anticipated that the MBSE tool of choice would be used through the entire mission lifecycle, from a high-level systems definition, down to sub-system level design and through to manufacture and test. With built in requirements management, capability requirements modelling as well as architectural and behavioural modelling, Enterprise Architect provided all of the functions required to manage the full system life-cycle. Within Enterprise Architect, the Zachman framework was chosen as it offered a consistent, although potentially over flexible, approach to model development.

Alternatively, Capella was chosen to model the autonomous fire detection system and its development. This project was far more experimental than ROKS; the usage of MBSE in this project was experimental in itself. The technical development was also limited to an on ground demonstration of the fire detection algorithm, hence aspects such as requirements management were not as critical as the system was not due to be flown in the near future. The experimental nature of the MBSE development also opened up the possibility to investigate the use of a more rigid framework. With the Arcadia method firmly integrated within the Capella tool, in addition to being open source, the Capella MBSE tool was selected.

Having utilised both tools in early phase design and development, each has proven beneficial to the projects they were implemented on. Craft Prospect will continue to use these tools, however, it has been found that particular programs are suited to specific project types and lifecycle phases. An initial trade-off of Enterprise Architect and Capella is provided in Figure 1.

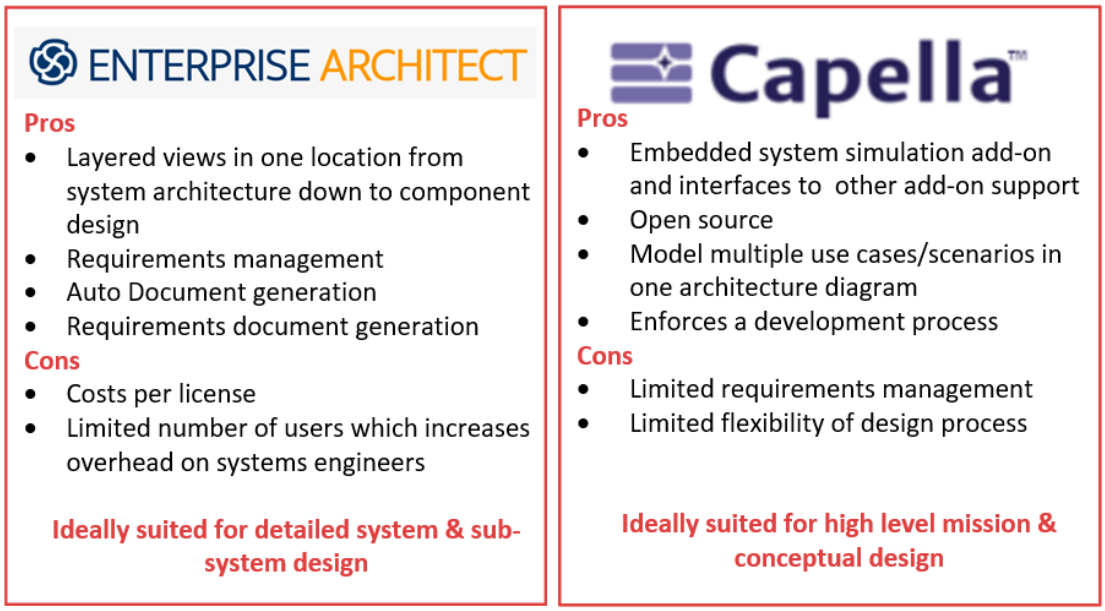

Figure 1: Trade-off between Enterprise Architect and Eclipse Capella

3. Implementation challenges and benefits

The typical benefits of MBSE are well known to the systems engineering community, namely it provides increased traceability, design consistency, quality improvements and, in time, reduces the overhead of document management.

However, Craft Prospect has found MBSE particularly beneficial in addressing the general challenges of nanosatellite design and the new space mission development approach.

Rapid life-cycle and agile development: Nanosatellites are known to have a much faster development lifecycle and tend to utilise an agile development approach in comparison to larger satellite developments. In addition, systems and discipline engineers in working in the new space sector are typically heavily involved in the system and sub-system development throughout the full life-cycle; design through to Assembly, Integration, Verification and Test (AIVT) and on-orbit commissioning. Therefore there are usually a smaller number of stakeholders involved in nanosatellite development, meaning design reviews and documentation requirements can be more informal. A document driven approach, or even using a specific requirements management package combined with document reviews, is not a cost effective approach for new space organisations. Combined with the rapid lifecycle, this can lead to more frequent document updates being required, and more overall time spent on document management. Holding all the relevant information in one place, with design changes in one area automatically translated throughout the rest of the design is a much more efficient and effective way of working to tight deadlines

Size, Weight and Power Constraints: There are significant physical and electrical constraints within nanosatellite design which leads to design dependencies being very tightly coupled to each other and small changes having a bigger impact on the satellite design elements. The traceability aspect of MBSE is very useful to identify the impact of changes made in one design area to another. MBSE also has the ability to model multiple use cases and functional chains in a single model, saving the need for multiple documents or design files to be managed simultaneously.

Usage of Commercial Off The Shelf (COTS) components: Usage of COTS components leads to less bespoke hardware design. Design work is more focused on mission level concept of operations and software architecture. Architectural visualisation is of higher value for design reviews and behavioural analysis is more beneficial than physical analysis for mission success.

The **challenges** faced in working with MBSE throughout both projects have been,

Overhead associated with selecting and learning the MBSE tools: This is challenging in many aspects, firstly knowing physically how to use the tool and adapt to the language used within MBSE takes time to learn and adjust to. Secondly the development or modelling method may also be very different to what a systems engineer may be used to following. Alternatively if the MBSE tool is too flexible in terms of the design process then the user loses some of the benefits of MBSE; it can end up being used simply as a tool for visual diagrams rather than enforcing functional analysis. Finally, in order to ensure traceability, or simulation, of different use cases and design levels the system needs to be modelled correctly. This is difficult to ensure when learning the intricacies of various tools and can only really be mastered through consistent usage and trial and error.

Adoption of MBSE by non-systems engineering disciplines: Typically an MBSE model is created and managed by a systems engineer, however the information stored within the model must be available for reference and review by project stakeholders and other engineering disciplines. Even with open source access or an exported HTML model it was found that other engineering disciplines were hesitant to access the released MBSE model and would instead rely on outdated documentation or verbal information. This leads to a lack of communication from system level down to sub-system level and introduces a risk that the design implementation does not match the design specifications. The approach within Craft Prospect taken to mitigate this risk was to frequently export the system information to a easily readable form (such as JPEG files of model views or an HTML export) and to

ensure the change control of these snapshots was well managed. Repeatedly referencing the location of the exported MBSE data and reviewing this with discipline engineers led to an improved adoption of MBSE to non-systems engineering disciplines, particularly in the manufacture and test phases of the ROKS flight model build.

Information Communication: Similar to the above challenge, it initially was quite challenging to communicate the visual information in the MBSE model to non-technical or high level project stakeholders. Without a document approach with detailed descriptions issues prior to a review or design meeting, some of the design or mission context was lost. Through multiple trials, Craft Prospect developed a set of model views to convey the key information to these stakeholders, and ensured that the documentation fields within the MBSE tool were well detailed to ensure transition of context in offline prior reviews.

4. Conclusions and future outlook

There were six key lessons learned from the usage of MBSE in both projects.

1. Experimentally introduce MBSE into the organisation alongside traditional methods initially, although this may have a larger overhead than committing to MBSE outright. Although at a point there should be a decision made to fully commit and transition to MBSE or to maintain a document driven approach. Mixing MBSE and a separate document approach negates a lot of the benefits MBSE provides and introduces the potential that the MBSE model is used as a diagram tool.
2. Experiment with the views and analysis that work for particular design phases and stakeholder communication. Generate model templates and 'rules' to reduce the overhead when re-using models, starting a new project or preparing for a review.
3. Account for the overhead of learning a new tool or invest in professional training. Training could have saved significant time in getting to grips with the tools.
4. In general, MBSE is a significantly better way to identify behavioural or functional architectures and interactions. It provides an excellent base to begin subsystem and software design.
5. A combination of tools or frameworks could be used on a single project or within an organisation based on the type of project or analysis that should be performed.
6. Spend more time on defining the methodology or process rather than tool selection.

In conclusion, Craft Prospect found that there was a significant return on investment when introducing MBSE to the organisation. Following the initial overhead of learning the process and particular tool, the effort associated with managing documentation and design information was drastically reduced. For example, when considering the fire detection project it took two dedicated days to work through the free online Capella tutorial and three days to generate the initial system model, including some iterative review. The amount of time required to further analyse and update the design information up to the first major design review is estimated to be 50% less than would have typically been spent.

The design improvement is difficult to quantify as both projects are very bespoke and are yet to be tested in orbit, however non-systems engineers and project level managers on both projects did report that they gained a much greater understanding of the system functionality and constraints than with previous similar projects. Looking forwards, Craft Prospect is moving towards continuing to implement and assess using MBSE for all design and development projects.