MBSE-2020 workshop abstract submission

Abstract : "Experience Report: History and State of the Practice of Model-Based Software Engineering in Thales Alenia Space in France"

Authors: Régis De Ferluc, Marco Panunzio

Thales Alenia Space in France

Introduction

Model-Based Software engineering methods and tools have been used in Thales Alenia Space in France for more than a decade, benefiting from active and efficient R&D efforts, including collaboration with space agencies, and accompanied by a pragmatic and incremental deployment. In this paper, we summarize the major steps of adoption of MBSE: 1) emergence of modeling; 2) consolidation and maturity; 3) link to other disciplines beyond SW development; 4) application to payload software development.

We highlight the main factors that have made the adoption possible. We describe the current state of the practice of Model-Based Software Engineering in Thales Alenia Space in France, and provide some insights about challenges ahead in the next future.

Emergence of modeling for on-board software

In the late 90's and early 2000s structured design methods were already adopted in the space domain to address challenges of hard real-time software development (HRT-HOOD[1] was finalised in 1994). The biggest merit of those methodology was to stimulate the emergence of an initial software reference architecture within the company, which would permit to support the development of real-time embedded software, while supporting domain-specific aspects (i.e., first adoption of PUS, specificities of the avionics). At the same time however, there was a large space for improvement in the design support of these methods.

[1] HRT-HOOD: A Structured Design Method for Hard Real-Time Ada Systems - de A. Burns, A. Wellings

The first emergence of satisfactory use of software modeling for on-board software has its origin in self-funded R&D activities in collaboration with Thales Group, which led to the definition of a model-based engineering environment for satellite platform software: Melody CCM.

Melody CCM was originally devised so as to target OMG's CORBA Component Model, yet it evolved and adapted to support space-specific considerations: support for PUS in the design space, support for precise data type modeling (mirroring the expressiveness of the Ada programming language), and use of a target run-time adapted for embedded space software. The software design environment was extended so as to support code generation for Ada, and targeting TAS' own reference architecture. The work capitalized on early advances in software modeling, such as those of the EU FP6 ASSERT research project, with ESA as project coordinator, or attempts to use UML as modeling language; yet the result was a decisive step forward, in particular thanks to the domain-specific nature of the modeling space, and the generated code, which was factorizing reference code patterns already in use.

The engineering environment (known as "CCM for Space") was operationally deployed for the first time in 2009 for the development of the Sentinel 3 platform software.

Consolidation and maturity

In the years 2012-2016, every new platform software was developed using more and more advanced evolutions of the MBSE methodology and toolset (Exomars TGO and EDM, Iridium Next, Spacebus NEO, and Earth Observation satellites, including SWOT). Evolution of the toolset was driven by two main factors: i) to quickly respond to specific program needs (adaptations and performance optimizations); (ii) to extend the capabilities of the toolset with additional capabilities.

Factor i) led to the decision of maintaining full control within the on-board software department of the development of the toolset, as only a dedicated team with knowledge both of modeling and toolset development and of the target software architecture could support this goal, and with the necessary reactivity.

Factor ii) was possible thanks to a synergy of self-funded R&D and participation to several R&D activities funded by ESA and CNES. Among all the topics, those that demonstrated good

potential and results in prototypal developments were progressively added to the operational toolsuite (e.g. generation of export files to the Satellite Data Base; auto-coding of configuration / missionisation software based on the content of the Satellite Data Base; generation of TM-TC ICD; MMU support in the modeling tool; Model-Based Test Campaign specification; Model-Based testing of OBSW missionization. Some other features were considered promising, but have not found (yet) a path towards operational deployment: support for Time and Space Partitioning; Model-based Schedulability analysis; Model-based test behavior specification).

Link to other disciplines beyond on-board development

The growth of the Software Factory within the software area (on both development and test sides) was resulting from a pragmatic and local improvement of engineering practices. At the same time, model-based techniques were adopted in related disciplines such as the Satellite Data Base (SDB-Next), the Operations Preparation Environment (SCOPE), the AOCS team (full GNC modeling with Matlab-Simulink), or Data Handling Teams (FDIR Design and Avionics Unit Specification in Capella). This new ecosystem has brought new opportunities (System to Software transition for equipment management SW, full auto-coding of GNC software, harmonization of test and operation environments, or digital continuity from design models to Satellite Data Base), but also raises new challenges considering the strong heritage of practices in the company. Just as a few examples: need to set-up co-engineering practices, need to align tools and technology, need to coordinate configuration management, …).

In this context, the Software Factory cannot been seen anymore as an independent asset, but needs to be considered together with many other external assets within the so-called concept of "Model-Based System-Software Factory". This interesting step in the evolution of the Model-Based adoption in TAS in France required a paradigm change in terms of organization and governance. Instead of local optimization, the Model-Based System-Software Factory seeks for global optimization throughout the whole process, where more effort is needed in the early stages of the V cycle in order to save significant amounts of time and money on the latter ones. It is particularly true when dealing with topics such as Electronic Data Sheets, Early Validation and Verification analysis, or System to Simulation transitions.

Application to Payload Software development

Model-Based practices have taken more and more importance for the development of avionics / platform on-board software. A recent trend foresees partial or full application of the same methods also for payload software. Payload software complexity has greatly increased in the past years, thanks also to the trend of moving function implementation from costly and often be-spoke ASIC implementations, to FPGAs, or SW executing on a general purpose space processors. Unfortunately, this ramp-up was not sustained by the formalism that comes with model-based practices, and the lowest reliability expectations associated to (part of) payload software have not permitted to justify early adoption of similar practices. However, it appears nowadays that Payload Software development can benefit substantially from the existing Model-Based System-Software factory, in particular for parts related to (Payload) Command and Data Handling, real-time behavior, communication protocols and resource management, i.e., aspects in common with platform software. In turn this requires adaptation to this new context (possibly different programming languages or software execution platforms, different underlying hardware, different performance/ predictability / reliability needs, …).

Initial deployments in this context were performed for the Payload Execution Platform of the MTG FCI and IRS payloads, and a sizeable telecom payload.

Challenges for the near future

As a continuation of the trend highlighted in previous sections, model-based software engineering requires to be used in context that show three new trends:

- A systematic search for a solution to the "digital continuity" challenge, i.e., the capability of *meaningfully* model a **full system** from the early design phases (i.e., 0, A/B1), and to transition modeling data in the design and implementation phases (B2, C, D) and later into operations, without loss of data, and maintaining flexibility of adjusting the abstraction of representation to the level meaningful to the actual phase of development
- The reconciliation of heterogeneity internal to the project, which derives from approaches, methodologies and technologies best suited for individual disciplines (i.e., AOCS, thermal, structures, avionics / SW), which should now be able to fit together,

breaking existing walls in the free, meaningful circulation of data between and beyond disciplines, throughout the whole development lifecycle

- The reconciliation of heterogeneity brought by the collaboration of several company (or several Agencies) within the same project, which may hinder effective communication and engineering work.

Recent work on methods and concepts such as Model-Based System Engineering, Ontologies, Engineering PDMs, Digital Twins, all reflect the desire of the engineering community to overcome those challenges.

These challenges will be all present in the Gateway project, a multi-agency endeavor lead by NASA, with contributions of ESA, JAXA and CSA for the development of the future human base in orbit around the moon. Thales Alenia Space in France is one of the partners selected for the realization of the Gateway I-HAB module. We will report on how we plan to deploy modeling approaches in such context, in particular for the areas of software and avionics, and what adaptations to our practices we foresee in this new development context.