Foundational => "statement of intent" from the language's own designers

Future Directions for MBSE with SysML v2

Sanford Friedenthal
SAF Consulting, LLC, U.S.A.

Keywords: OMG Systems Modeling Language, Systems Modeling Language, SysML v2, Model-Based Systems Engineering, MBSE.

Abstract: Model-based systems engineering (MBSE) has evolved over the last several years in response to the need to deal with growing system complexity and increased enterprise agility. MBSE is an approach to systems engineering where information about the system is captured in a system model. This approach can provide a more complete, consistent, and traceable system design than a more traditional document-based approach where information about the system is captured in a variety of documents, informal diagrams, and spreadsheets.
SysML v1 was adopted in 2006 and has been a key enabler of MBSE. Since that time, much has been learned about applying MBSE with SysML. The next generation of SysML (v2) is being developed by the SysML v2 Submission Team (SST) to provide capabilities that address the limitations of SysML v1 and enable the evolving practice of MBSE. This presentation summarizes the future directions for MBSE and how SysML v2 can support these needs.

1 SYSTEMS ENGINEERING OVERVIEW

Systems engineering aims to ensure the pieces of the system work together to achieve the objectives of the whole system. As an example, a car must be designed such that its powertrain, braking, steering, chassis, body, interior, electrical system, and infotainment work together to achieve the objectives of the car. The car's objective is to transport people and things safely and reliably while satisfying many other stakeholder expectations for performance, comfort, aesthetics, and cost. Systems engineering involves architecting solutions that balance competing requirements while managing the technical, cost, and schedule risk associated with development of complex systems.

Systems engineering practice began maturing as a discipline in the mid-twentieth century to address the growing challenges of the aerospace, defense, and telecommunications industries. Systems engineering has traditionally been defined as a top-down problem-solving methodology. The practice includes problem identification by eliciting and analyzing needs of the system stakeholders. In aerospace and defense industries, it is common to define one or more missions that the system is intended to support and to specify measures of effectiveness (moe's) that represent mission and stakeholder value.

The practice of systems engineering then systematically derives the system, subsystem, and component requirements to achieve the mission. This often involves determining how a system must interact with its external systems, users, and the natural environment to accomplish the mission, and then deriving the requirements for how the parts of the system must interact to satisfy the system requirements. The requirements are then used by design engineers to perform more detailed design and implementation. Systems engineering then verify that the parts satisfy their requirements. This practice of flowing requirements down from mission to component level, performing the detailed design and implementation, integrating the parts into the next higher level of assembly, and verifying each level of assembly satisfies its requirements is often referred to as a 'vee' process.

Systems engineering must specify requirements based on considerations from across the system lifecycle. This includes requirements on the system and its enabling systems to effectively manufacture, deploy, maintain, and dispose the system. For example, maintenance requirements may be imposed to ensure that critical parts of a car are monitored, accessed, and replaced in the event of a failure.

Friedenthal, S.
Future Directions for MBSE with SysML v2.
DOI: 10.5220/0011946300003402
In *Proceedings of the 11th International Conference on Model-Based Software and Systems Engineering (MODELSWARD 2023)*, pages 5-9
ISBN: 978-989-758-633-0; ISSN: 2184-4348

Systems engineering must determine a preferred solution that satisfies the stakeholder needs. This can involve extensive trade-off analysis and design optimization to evaluate alternatives and select a preferred solution. The preferred solution is the one that maximizes mission and stakeholder value within the constraints of the project.

The system development process rarely follows a strictly top-down process. Detailed design activities are often happening concurrent with the requirements flow down process. As a result, further iteration is needed to reconcile the requirements and detailed design solutions.

Systems engineering is an integrating discipline that must interact with other engineering disciplines, program management, and the customer and treat each of them as stakeholders of the systems development process. The customer is the acquirer of the system and often represents the users of the system. Program management is responsible for program success and the overall budget and schedule. The other engineering disciplines are responsible for the detailed design. As system stakeholders, systems engineering must address each of their concerns throughout the development process.

2 CHALLENGE WITH TRADITIONAL DOCUMENT-BASED SYSTEMS ENGINEERING

Systems engineering traditionally captures system, subsystem, and component requirements, design, analysis, and verification information in a variety of different artifacts including text documents, informal diagrams, and spreadsheets. The documents are authored by different members of the systems engineering team, and as the system design evolves, this documentation must be updated to reflect the changes. The effort required to create and maintain this documentation is substantial. In addition, it is a significant challenge to ensure consistency of the information contained within each document and from one document to another. An additional challenge is that the information is captured informally in text and diagrams and is subject to different interpretations.

This challenge is illustrated when there is a change to a system requirement. The requirement must be assessed to identify what parts of the system may be impacted, and then various analysis must be performed to determine the potential impact. Once the impact is determined, proposed changes to the design and derived requirements are made to satisfy the system requirement by iterating through the 'vee' process. This is often a very manual and time-consuming process where some information is often not well documented, thus making it difficult to coordinate changes to all the impacted documents.

3 MODEL-BASED SYSTEMS ENGINEERING (MBSE)

MBSE is a way to perform systems engineering where information about the system is captured in a system model. The model is a primary source of the information that is managed throughout the system lifecycle as part of the technical baseline. This contrasts with the more traditional document-based approach where the information is captured in a variety of documents, informal diagrams, and spreadsheets as described previously. MBSE is intended to provide a more complete, consistent, and traceable system design.

The roots of MBSE trace back many years, but this practice is becoming increasingly important to address the growing system complexity associated with increasing amounts of software, data, and interconnectivity. MBSE evolved with changes in computing technology and programming standards starting with the introduction of modeling and simulation as an engineering practice in the 1960's. Computer aided software engineering (CASE) and computer aided design (CAD) methods and tools were introduced in the 1980's. Computer aided systems engineering and early systems behavior modeling tools were introduced in the early 1990's. UML modeling tools were introduced in the late 1990's. SysML v1 tools were introduced in mid 2000's. SysML v1 provided a robust systems modeling capability to specify system behavior, structure, requirements, and parametrics.

4 FUTURE OF MBSE

According to the INCOSE Systems Engineering Vision 2035 (2023), '*the future of systems engineering is model-based'*. A model of the system is essential to manage the growing system complexity and to achieve the agility required to adapt to on-going changes in requirements, design, and technology. This direction is part of a broader trend towards digital engineering that is being enabled by

advances in computing technology and standards. Digital engineering emphasizes the need to capture engineering information in models and other structured data that can be semantically integrated and interpreted by computers.

When complemented with agile practices, MBSE provides a capability to provide more rapid response to changes in requirements and design. Agile practices for systems engineering include the use of automated workflows and configuration management of the digital thread, where the digital thread is the interconnected set of artifacts that constitute the evolving technical baseline.

The benefits for MBSE will continue to increase as the practice matures. This includes leveraging modeling patterns, reference architectures, and reuse libraries to more effectively support product-family engineering approaches and system design evolution. It is also anticipated that system models will enable more effective reasoning about the system to answer fundamental questions such as 'what is the impact of a change to a particular requirement'. The models should facilitate improved automation for system quality checks and status reporting. More generally, the models should enable shared understanding across the broad set of system stakeholders and improved overall capability to manage complexity and risk.

A primary goal for MBSE is to use the models to verify the system satisfies its requirements and surface issues early in the system lifecycle before building and testing the system. This significantly reduces the adverse impact on program cost, schedule, and risk of discovering these issues later in the system lifecycle.

To fully benefit from the application of MBSE, it must be applied across the system lifecycle from conceptual design through development, production, and support of the system, and at all levels of system design including system-of-systems, systems, subsystems, and components. Scaling to this increased scope of MBSE will require significant improvement in modeling capabilities in terms of the language, methods, tools, and the associated modeling skills.

5 SysML V2: THE NEXT GENERATION SYSTEMS MODELING LANGUAGE

SysML v1 was adopted in 2007. Much has been learned from the application of MBSE with SysML. SysML v2 is the next generation systems modeling language to addresses many of the limitations that were identified with SysML v1. SysML v2 (2023) is intended to provide much broader adoption and improve the effectiveness of MBSE to help realize the benefits highlighted in the previous section. The objectives for SysML v2 are to make significant improvements relative to SysML v1 in the following areas:

- Precision and expressiveness of the language
- Consistency and integration among language concepts
- Interoperability with other engineering models and tools
- Usability by model developers and consumers
- Extensibility to support domain specific applications
- Migration path for SysML v1 users and implementors

Some of the key elements of SysML v2 include:

- A new metamodel that is focused on system modeling. The metamodel is not constrained by UML but preserves most of UML modeling capabilities and is grounded in formal semantics.
- More robust visualizations based on a flexible view and viewpoint specification. The language includes both graphical and textual notations and supports tabular renderings of the model content.
- A standardized API to access the model from other applications

The SysML v2 language design is part of a multi-layered language architecture. The Kernel Modeling Language (KerML) includes the lower three layers to provide the foundation constructs and core semantics. SysML v2 extends the KerML to include the systems modeling concepts such as parts, actions, states, requirements, analysis case, verification case, and others. The language syntax and semantics can be further extended using model libraries without the need to extend the metamodel, which minimizes the need for additional tooling.

The textual notation is new to SysML v2. The graphical and textual notation provide complementary renderings of the same underlying model. Tool support for both notations will enable a change that is made in the textual notation to be rendered in the graphical notation and vice versa. The combination of both graphical and textual representations provides significant modeling flexibility.

The SysML v2 language capabilities enable the precise representation of many aspects of a system

including its structure, behavior, requirements, analysis cases, and verification cases. It also includes the ability to specify user defined views and viewpoints.

SysML v2 employs a regular pattern of definition and usage across virtually all language constructs. This regularity and associated consistent terminology should facilitate learning and use of the language and facilitate automation such as model checking.

A significant improvement over SysML v1 is referred to informally as 'usage focused modeling'. This enables one to define a system design configuration with parts that may or may not have explicit definitions. For example, a system decomposition in SysML v1 requires that blocks be decomposed into part properties, and that the part properties be typed by blocks which are further decomposed into the next level of decomposition of part properties. In SysML v2, a system decomposition can be represented as a parts decomposition directly. The zigzag decomposition pattern in SysML v1 (i.e., block to part to block to part) is replaced by part-to-part decomposition in SysML v2. This usage focused approach applies to all usage elements including actions, states, requirements, and others, and results in a significant simplification in how the language is used.

The usage focused modeling also provides the ability to readily adapt the usage to its context. For example, a vehicle may require different values for the tire pressure on its front and rear tires. The *frontTire* and the *rearTire* can be modeled as parts that are defined by their part definition *Tire*, which contains an attribute *pressure*. The *frontTire* and *rearTire* can redefine their *pressure* to have different default values. A part inherits features from its definition and can redefine or subset the inherited features or add new features, enabling the part to be adapted to its context. Similarly, a part can subset another part which is analogous to class inheritance. This enables the part to inherit features from the part it subsets and then redefine or subset the inherited features or add new features.

There are many other aspects of the language that illustrate its expressiveness, precision, and regularity, but discussion of this is beyond the scope of this paper.

The Systems Modeling Application Program Interface (API) and Services or SysML v2 API for short provides a standard way to access the SysML v2 model and perform various operations on the model. This is intended to greatly enhance the interoperability of the system model with other applications and tools. This capability has already been demonstrated by external applications that have interacted through the API. This includes model management applications that manage model versions and branching, visualization applications that render the model content in a graph, analysis applications that provide solvers to solve the equations specified by the analysis in the SysML v2 model, and an open- source CAD viewer application that renders the geometric information contained in the SysML v2 model.

The SysML v2 specification is being submitted to the OMG for approval as a beta specification in the first quarter of 2023. If approved, the specification enters its finalization phase which provides an opportunity for tool vendors that are implementing the specification to provide feedback and propose resolutions regarding their implementation issues. It is anticipated that the final adopted specification will be available in 2024.

6 SUMMARY

Model-based systems engineering builds on a history dating back to the 1960's with the advent of computer simulation as an engineering practice and further advancements in computer aided software and hardware design methods and tools. MBSE tools were introduced in the early 1990's. SysML v1 is based on UML and was adopted in 2007. Since that time, much has been learned about applying MBSE with SysML.

The future of systems engineering is model-based to deal with the increasing system complexity and the need to effectively respond to on-going changes in requirements, design, and technology. The model-based systems engineering approach is part of the digital transformation that leverages advances in computing technology and standards.

SysML v2 is the next generation systems modeling language that is intended to significantly improve MBSE adoption and effectiveness over SysML v1. In particular, it is intended to improve the precision, expressiveness, usability, interoperability, and extensibility of the language while retaining a transition path for SysML v1 users and tool implementors. SysML v2 includes a new metamodel that has been designed to address the needs for systems modeling from the onset, while retaining much of the legacy UML capabilities. It provides a textual syntax in addition to the graphical syntax and a robust visualization capability. SysML v2 also includes a standard API that will greatly enhance interoperability.

SysML v2 is anticipated to be adopted as a beta specification in early 2023 and become a formally adopted specification by the Object Management Group in 2024.

REFERENCES

(January 2022) INCOSE Systems Engineering Vision 2035, last accessed on January 03, 2023 at https://www.incose.org/about-systems-engineering/se-vision-2035

SysML v2 Submission Team, (December 2022) OMG Systems Modeling Language ™ (SysML®) Version 2.0, last accessed on January 03, 2023 at https://github.com/Systems-Modeling/SysML-v2-Release, *This draft specification has not been submitted to the OMG*