Evolving SysML and the System Modeling Environment to Support MBSE
Draft (March 30, 2015)
S. Friedenthal/R. Burkhart

Introduction. The transition to a Model-Based Systems Engineering (MBSE) approach is generally viewed as essential for systems engineering to meet increasing demands of system complexity, design cycles, productivity, and quality. Many other engineering disciplines, such as mechanical, electrical, and controls engineering, utilize models as integral parts of their practice. Such models have long been important for systems engineering as well, but MBSE emphasizes the need to create a coherent model of the system that helps to integrate other aspects of the design, including electrical, mechanical, and software. This ***system model*** provides a shared view of the system that can drive communication and coordination across the system development lifecycle. This model represents the authoritative source of information that is maintained to ensure consistency and traceability across requirements, design, and verification. The model-based approach contrasts with the traditional document-based approach in which information is spread across many different documents created in common applications such as Word, Visio, Excel, and PowerPoint. To fully leverage a model-based approach, the system model must be maintained as a fundamental part of the technical baseline that is integrated with other engineering models and tools.

Background and Current State of SysML.The OMG Systems Modeling Language (OMG SysML™, www.omgsysml.org) was adopted by the OMG in 2006 and Version 1 was made available in 2007. The SysML specification resulted from a collaborative effort between INCOSE, the OMG, and the ISO STEP AP-233 Working Group to develop the requirements for the language, and then develop the system modeling language solution in response to these requirements.

SysML is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities. The language provides graphical representations with a semantic foundation for modeling system requirements, behavior, structure, and constraints.

Since its adoption, SysML has enabled broad recognition and increased adoption of model-based systems engineering practices across industry. Much has been learned from this experience, both in the strengths and weaknesses of SysML as a language, and the benefits and challenges in adopting and applying MBSE.

A standard system modeling language, such as SysML, is needed to express fundamental systems engineering concepts such as system composition, interconnections and interfaces, functional and state-based behavior, parametric aspects of a system, and relationships between requirements, design, analysis, and verification. Such a language is proving to be an essential capability to specify and architect ever more complex systems, and to overcome the informational “Tower of Babel” among the systems engineering community. The capability to express system concepts in the form of models results in quality improvements by reducing downstream design errors, and in productivity improvements through reuse of models across projects and throughout the lifecycle. Other benefits are also realized, such as the ability to automate tasks such as change impact analysis, and to auto-generate reports and documentation with increased confidence that the information is valid, complete, and consistent.

A major emphasis of the SysML community has been on the integration of SysML models and tools with other engineering tools. There has been significant progress integrating system-level models with software design, especially for software developed with UML models and tools, since SysML is based on UML. In addition, considerable progress has been made integrating SysML with engineering analysis and simulation using various integration methods and tools. SysML has been integrated extensively with requirements management tools. There has been progress integrating SysML with product lifecycle management and hardware design tools, but much more remains to be done.

A key challenge for MBSE is the learning curve to develop the modeling skills. Some of this challenge is inherent in the complexity of systems and systems engineering, such as the need to define and maintain multiple consistent views of a system. Other challenges are unique to the language and current modeling tools. The challenges of learning a more formal language, as well as the tools to build and maintain models, together with the MBSE methodologies for effective application, all require significant investment by the organization to build an MBSE capability. Applying these capabilities on a program requires a disciplined approach to establish and maintain the system model as an integral part of the technical baseline throughout the system lifecycle.

***Future Directions for SysML*.**The OMG Systems Engineering Domain Special Interest Group chartered the System Modeling Assessment and Roadmap Working Group to assess how well SysML is supporting MBSE, and to develop a roadmap for SysML as part of a System Modeling Environment. The Working Group is beginning to identify driving requirements for the next generation of SysML and the tools that implement the language. Some of the initial capabilities and requirements are summarized below. These are subject to further analysis, inputs, and review with the broader community.

A System Modeling Environment (SME) is used by system modelers to perform MBSE in the broader context of Model-Based Engineering (MBE). This environment must provide basic capabilities that impose requirements on both the modeling language and the tools. Some of the key capabilities for the SME include:

- model construction
- model visualization
- model analysis
- model management
- model exchange and integration
- support for MBSE collaboration and workflow

Some of the key effectiveness measures include:

- **Expressive:** Ability to express the system concepts
- **Precise:** Representation is unambiguous and concise
- **Presentation/communication**: Ability to effectively communicate with diverse stakeholders
- **Model construction:** Ability to efficiently and intuitively construct models
- **Interoperable:** Ability to exchange and transform data with other models and structured data
- **Manageable:** Ability to efficiently manage change to models
- **Usable:** Ability for stakeholders to efficiently and intuitively create, maintain, and use the model
- **Adaptable/Customizable:** Ability to extend models to support domain-specific concepts and terminology

Based on the above capabilities and effectiveness measures, some of the preliminary driving requirements for the next-generation system modeling language and tools have been identified as follows:

1. The next-generation modeling language must express the core systems engineering concepts. This requires definition of a robust data model that reflects these concepts. The requirements that drove SysML are heavily based on the original Systems Engineering Conceptual Model that was jointly developed by the INCOSE/OMG/AP233 requirements team. This model will be revisited in light of what has been learned over the last several years, and refined as necessary to express the core systems engineering concepts.

2. The next-generation modeling language must include precise semantics that avoid ambiguity and enable a concise representation of the concepts. SysML currently leverages the UML metamodel for much of its semantic foundations. The language must be based on a well-specified logical formalism that can leverage the model for a broad range of analysis and model checking. This includes the ability to validate that the model is logically consistent, and the ability to answer questions such as the impact of a requirement or design change, or the assessment of how a failure could propagate through a system. The language and tools must also integrate with a diverse range of equation solvers and execution environments that incorporate the capture of quantitative data.

3. The next-generation modeling language and tools must provide flexible and rich visualization and reporting capabilities to support a broad range of model users. SysML currently includes concepts for view and viewpoint. Tool vendors and end users have been able to apply this capability to query the model and provide flexible reporting capability. The next generation must extend this capability with advanced visualization techniques that include dynamic zoom, filtering, and traversal of model relationships, and visualizing the dynamic behavior of a system, such as provided by simulations. The modeling language must also support symbol libraries that extend well beyond the current SysML notations. It is also expected that the modeling environment will provide a simplified web interface to dynamically view the model from a diverse set of viewpoints.

4. The next-generation modeling language and tools must enable much more intuitive and efficient model construction. It often requires several clicks to capture a core concept in a model. More streamlined and efficient user interfaces could reduce the time and effort to build and maintain a model. The ability to repeat common modeling patterns with reduced user input (e.g., table-based entry) is another capability to increase modeling productivity and understanding.

5. The next-generation modeling language and tools must support MBSE in the broader context of Model-Based Engineering (MBE), where the models and tools are fully integrated across discipline-specific engineering tools, including hardware and software design, analysis and simulation, and verification. All these model-based tools working together establish an environment for engineering the total system.

6. The next-generation modeling language must provide a standard application programming interface (API) to provide dynamic access to the model, while providing appropriate access controls. It should also integrate with emerging platforms for managing and integrating model-based content, such as Open Services for Lifecycle Collaboration (OSLC), which is based on linked data and semantic web technology, and the Functional Mockup Interface (FMI), which provides model exchange and co-

simulation capability for executable behavior models. Model transformation is another core capability of the SME, by providing the ability to translate from one modeling language to another.

7. The next-generation modeling language must be capable of being managed in a heterogeneous and distributed modeling environment. The ability to manage change to the model, where multiple users are collaborating on a single model, is challenging enough. This basic capability requires extensive branch and merge capability that includes effective means for evaluating and integrating changes from multiple users, while maintaining a history of all changes. These challenges are magnified when multiple models and tools are all part of the collaboration. The ability to integrate with Product Lifecycle Management (PLM) environments, which enable versioning, configuration, and variant management, is a fundamental SME requirement.

8. The next-generation modeling language and tools must enable efficient and intuitive use by a broad range of users with diverse skills. This imposes requirements on model precision, model construction, model visualization, model management, and several other aspects of the language and tools. As noted previously, the learning curve for the SysML language and tools is quite steep. Usability must be a primary consideration for the next-generation modeling language and tools.

9. The next-generation modeling language and tools must be highly adaptable and customizable to multiple application domains. This implies that the modeling language must be extensible to address domain-specific concepts, and that the modeling tools provide flexible means for the user to enter, analyze, and visualize model data in ways that are meaningful to each domain. In addition, the SME must accommodate customization that is performed in a standard and rigorous way.

10. To protect investments made by organizations, the next-generation modeling languages must support the migration of existing models with minimum information loss. Models must also be capable of being stored in neutral formats that can be retained for future access.

11. The next-generation modeling language and tools must enable evolution of the above capabilities to take advantage of on-going advances in technologies, concepts, methods, and theories. Its architecture must be modular and extensible and support the definition of conformance levels that satisfy requirements of different applications.

Summary. During the seven-plus years since SysML 1.0 was released as a standard systems modeling language, the language and tools have matured considerably, as has the practice of MBSE. There is now much broader recognition that systems engineering must evolve to a model-based discipline. The transition to a model-based discipline has been identified as an INCOSE strategic initiative. It is also recognized as a key practice in the INCOSE Systems Engineering Vision 2025 (pg 38-39), which describes the projected state for MBSE in 2025 as*: "Formal systems modeling is standard practice for specifying, analyzing, designing, and verifying systems, and is fully integrated with other engineering models. System models are adapted to the application domain, and include a broad spectrum of models for representing all aspects of systems. The use of internet-driven knowledge representation and immersive technologies enable highly efficient and shared human understanding of systems in a virtual environment that span the full lifecycle from concept through development, manufacturing, operations, and support."*

The need for a standard systems modeling language that can be readily adapted to domain-specific applications is a key enabler of this MBSE capability. Much has been learned from the use of SysML and

MBSE applications. These insights can inform the development of the next-generation systems modeling language.

The path forward has begun through the OMG Systems Engineering Domain Special Interest Group (SE DSIG).This must be a collaborative effort that draws upon a diverse range of end users, tool vendors, academics, and others who can help provide a language that is broadly accepted by the engineering community, industry, and academia.