Realizing the importance of systems engineering and its critical role in the early phase of systems development, the Object Management Group (OMG) has now standardized the Systems Modeling Language (SysML). Based on a subset of the Unified Modeling Language (UML), OMG SysML enables systems engineers to analyze, specify, design, and verify complex systems. It is intended to enhance systems quality, improve the interchange of systems engineering information, and help bridge the semantic gap between systems, software, and other engineering disciplines. In this article, Matthew examines the four pillars of OMG SysML, including structure, behavior, requirements, and parametrics.

Since its adoption in 1997, UML has proven to be immensely popular with software engineers to the point where it is now the only widely used visual modeling language for software engineering. Unfortunately, UML’s software focus has discouraged many systems engineers from adopting it in earnest.

Systems engineers have used various approaches to address UML’s shortcomings for systems engineering purposes, but none have been ideal. Some systems engineers have made use of the stereotypes provided in UML to create libraries or profiles of entities in their application domain to express nonsoftware concepts. Others have used additional tools such as Microsoft Visio to model their systems engineering concepts in conjunction with their UML models, but this merely created two separate models that they could not integrate or cross-reference. Some simply ignored the problem or used words to fill the gap. Some tool manufacturers such as ARTiSAN Software Tools extended UML to facilitate the integration of hardware, software, and systems engineering concepts into a single model, but this left them open to the charge of being nonstandard.

As a consequence, the OMG decided to pursue the development of a new version of UML specifically tailored for systems engineering. The OMG’s goal was to create a “standard modeling language for systems engineering to analyze, specify, design, and verify complex systems … to enhance system quality, improve the ability to exchange systems engineering information amongst tools, and help bridge the semantic gap between systems, software, and other engineering disciplines.” However, the UML modifications systems engineers needed required more than just adding stereotypes. As a result, many of the concepts tool developers created to extend UML for systems engineering purposes have now been integrated into the OMG SysML.

SysML diagrams
OMG SysML reuses a subset of UML 2.0, adding new diagrams and modifying others. It includes diagrams that can be used to specify system requirements, behavior, structure, and parametric relationships, known as the four pillars of OMG SysML. Only two of these pillars are entirely new – the requirements and parametric diagrams. One of the design goals was to avoid making changes unless absolutely necessary.
Figure 1 makes use of cross-cutting constructs that apply to both structure and behavior. Cross-cutting constructs support concerns that cut across the different views and may be addressed by all or disparate parts of the model.

System structure is represented by block definition diagrams and internal block diagrams. A block definition diagram describes the system hierarchy and system/component classifications. The internal block diagram describes the internal structure of a system in terms of its parts, ports, and connectors. The package diagram is used to organize the model.

The behavior diagrams include the use case diagram, activity diagram, sequence diagram, and state machine diagram. A use case diagram provides a high-level description of the system functionality. The activity diagram represents the flow of data and control between activities. A sequence diagram represents the interaction between collaborating parts of a system. The state machine diagram describes the state transitions and actions that a system or its parts perform in response to events.

The requirement diagram captures requirements hierarchies and the derivation, satisfaction, verification, and refinement relationships. These relationships provide the capability to relate requirements to one another and to system design models and test cases. The requirement diagram provides a bridge between typical requirements management tools and system models.

The parametric diagram represents constraints on system parameter values such as performance, reliability, and mass properties to support engineering analysis. OMG SysML includes an allocation relationship to represent various types of allocation including allocation of functions to components, logical to physical components, and software to hardware. Figure 2 illustrates the SysML profile in ARTiSAN Studio 6.2, showing blocks, constraints, port flows, and requirements.

SysML extensions

Additionally, OMG SysML provides extensions that can be very loosely grouped under the term continuous, but are generally applicable to any sort of distributed flow of information and physical items through a system. These include:

- Restrictions on the rate at which entities flow along edges in an activity or in and out of parameters of behavior. This includes both discrete and continuous flows, either of material, energy, or information. Discrete and continuous flows are unified under rate of flow, as is traditionally done in mathematical models of continuous change where the discrete increment of time approaches zero.

- Extension of object nodes including pins with the option for newly arriving values to replace values that are already in the object nodes. OMG SysML also extends object nodes with the option to discard values if they do not immediately flow downstream. These two extensions are useful for ensuring that the most recent information is available to actions by indicating when old values should not be kept in object nodes and for preventing fast or continuously flowing values from collecting in an object node, as well as modeling transient values such as electrical signals.
OMG SysML introduces probability into activities through extending:

- Edges with probabilities for the likelihood that a value leaving the decision node or object node will traverse an edge.
- Output parameter sets with probabilities for the likelihood that values will be outputted on a parameter set.

It also includes an allocation relationship to allocate one model element to another. Allocation is the term used by systems engineers to denote the organized cross-association (mapping) of elements within the various structures or hierarchies of a user model. Often, this is the allocation of function to form, such as the deployment of software on a hardware platform, or a use case to an organization or system entity. From a systems engineering perspective, this is applicable to abstract system specifications as opposed to a particular constrained method of system or software design. Allocations can be used early in a design as a precursor to more detailed, rigorous specifications and implementations.

The allocation relationship can provide an effective means for navigating the model by establishing cross relationships and ensuring the various parts of the model are properly integrated. The OMG SysML specification includes some specific subclasses of allocation for allocating behavior, structure, and flows, but these serve as examples rather than an exhaustive list. A typical example is the allocation of activities to blocks (for example, functions to components).

**Supporting the whole life cycle**

The OMG SysML specification is intended to be compatible with the evolving ISO AP233 standard, a data exchange protocol for systems engineering data based on ISO 10303. STEP (ISO 10303) is a standard to describe, represent, and exchange industrial data in a computer interpretable format. For data exchange among different application domains, a neutral (that is, tool independent) data model must be defined. This is called Application Protocol (AP). ISO 10303 provides APs for representing this data exchange. Currently, about 40 different APs are defined.

The intent of AP233 is to support the whole system development life cycle, ranging from requirements definition to system verification and validation. Different application areas include engineering analysis, algorithm design, planning tools, testing tools, software design, mechanical CAD, and electrical CAE. This goal is consistent with that of systems engineering in general, which is to tie together the different domain engineering disciplines into one consistent view of the system. Figure 3 shows how OMG SysML is intended to share data with other engineering tools using AP233 as the neutral data exchange format.

Extensions made to UML 2.0 in the OMG SysML profile will provide systems engineers with a robust language for modeling systems that include models of requirements, behavior, structure, and parametrics. At the same time, the profile’s extensive reuse of UML will facilitate a much smoother flow down from systems engineering to software engineering than otherwise possible.

For more information on the complete SysML notation, contact info.us@artisansw.com.

Matthew Hause is principal consultant at ARTiSAN Software Tools in Cheltenham, United Kingdom, where he handles mentoring, sales presentations, and training courses. He has been developing real-time systems for almost 30 years with experience in power systems, process control, communications, supervisory control and data acquisition, distributed control, and other areas of real-time systems. He has written a series of white papers on project management, systems engineering, and systems development with UML and has been a regular presenter at INCOSE, the IEE, and other conferences. Matthew studied Electrical Engineering at the University of New Mexico and Computer Science at the University of Houston, Texas.