Integrating Supporting Aspects into System Models

Integrating Supporting Aspects into System Models

Lead Author: *Sanford Friedenthal,* ***Contributing Authors:*** *Dov Dori, Yaniv Mordecai*

This article discusses the integrated modeling of systems and supporting aspects using Model-Based Systems Engineering methodologies and frameworks. Supporting aspects of systems engineering include:

- Engineering Management
- Project Management
- Requirements Engineering and Management
- Risk Modeling, Analysis, and Management
- Quality Assurance, Testing, Verification, and Validation
- System Integration and Employment
- Analysis of "-ilities" (e.g., Reliability, Availability, Maintainability, Safety, And Security (RAMSS), Manufacturability, Extensibility, Robustness, Resilience, Flexibility, and Evolvability)

These aspects can pertain to physical facets, as well as to functional, structural, behavioral, social, and environmental facets of the core system model. The article focuses on three main aspects:

1. Project and Engineering Management
2. Risk Modeling, Analysis, and Management
3. Requirements Definition and Management

□

Contents

Background

Integrated Modeling of Systems and Projects

Integrated Modeling of Systems and Requirements

- SysML-Based Requirements Engineering
- OPM-Based Requirements Engineering

Integrating Risk into System Models

References

- Works Cited
- Primary References
- Additional References

Background

The model-based approach to systems engineering considers the system model as much more than a plain description of the system; the model is the central common basis for capturing, representing, and integrating the various system aspects listed above. The model is essential to the design and understanding of the system as well as to managing its life cycle and evolution. Modeling languages constitute the basis for standardized, formal descriptions of systems, just like natural languages form the basis for human communication.

As systems progressively become more complex and multidisciplinary, the conceptual modeling of systems needs to evolve and will become more critical for understanding complex design (Dori 2002). In addition to facilitating communication among clients, designers, and developers, conceptual modeling languages also assist in clearly describing and documenting various domains, systems, and problems, and define requirements and constraints for the design and development phases (Wand and Weber 2002). The importance of model-based analysis is demonstrated by the variety of conceptual modeling methodologies and frameworks, although *de facto* standards are slow to emerge. While certain disciplines of engineering design, such as structural analysis or circuit design, have established modeling semantics and notation, the conceptual modeling of complex systems and processes has not yet converged on a unified, consolidated modeling framework (Estefan 2007). The challenge is not only to integrate multiple aspects and support the various phases of the system’s life cycle, but also to capture the multidisciplinary nature of the system, which has led to the creation of various frameworks. Nevertheless, the information systems analysis paradigm is currently the most widely used, perhaps due to the need to integrate complex systems via information-intensive applications and interactions.

Integrated Modeling of Systems and Projects

This section discusses the integrated modeling of systems and projects and of the project-system relationship (often called Project-Product Integration). The fields of project management and systems engineering have been advancing hand-in-hand for the last two decades, due to the understanding that successful projects create successful systems. Many of the main systems engineering resources pay considerable attention to project management and consider it to be a critical process and enabler of systems engineering (INCOSE 2012; NASA 2007; Sage and Rouse 2011). The integration of system-related aspects and concepts into project plans is more common than the integration of project-related aspects and concepts into system models. Because the project is a means to an end, it is the process that is expected to deliver the system. Indeed, project activities are often named after or in accord with the deliverables that they are aimed at facilitating (e.g., “console design,” “software development,” “hardware acquisition,” or “vehicle assembly”). Each is a function name, consisting of an object (noun), or the system to be attained, and a process (verb), being the project or part of the project aimed at attaining the end system.

The specific process associated with each of these examples refers to different stages or phases of the project and to different maturity levels of the system or sub-system to which it applies. Moreover, the mere inclusion of system and part names in activity names does not truly associate system model artifacts with these activities. Overall, it is not truly possible to derive the set of activities associated with a particular part or functionality of the system that will be delivered by the project. Project-Product integration is not straightforward, as project models and system models are traditionally disparate and hardly interface. A model-based approach to project-system integration follows a system-centric paradigm and focuses on incorporating project-related aspects and concepts into the core system model, as opposed to the project-centric approach described in the previous paragraph. Such aspects and concepts include schedule, budget and resources, deliverables, work-packages, constraints and previous relations. The integrated system-project model should provide useful information on the mutual effects of project activities and system components and capabilities. Some examples of integrated system-project modeling include the following:

- The set of project activities associated with a particular system component, feature, or capability.
- The set of resources required for performing a task of designing or developing a particular component of the system.
- The team or subcontractor responsible for delivering each system component.
- The preexisting dependencies between activities of system components deployment.
- The cost associated with each system component, feature, or capability.
- The parts of the system negotiated for each delivery, deployment, build, release, or version.

The Work Breakdown Structure (WBS) is designed to support the division of the project scope (work content) amongst the individuals and organizations participating in the project (Golany and Shtub 2001). The WBS is traditionally organization or activity-oriented; however, one of its main cornerstones focuses on the deliverable, which corresponds to the system, sub-system, component, or a capability or feature of one or more of these. A deliverable-oriented WBS, in which the high-level elements correspond to primary sub-systems, is advocated, as it is likely to allow the WBS to be more product-oriented (Rad 1999). An integrated approach to project planning and system modeling (Sharon and Dori 2009) merges the system model with the project's WBS using Object-Process Methodology (OPM) (Dori 2002). The unified OPM model captures both the project activities and the system components and functionalities.

The Design Structure Matrix (DSM) is a common method for enhancing and analyzing the design of products and systems. DSMs can be component-based, task-based, parameter-based, or team-based (Browning 2001). A DSM for an OPM-based project-product model derives a hybrid DSM of project activities and system building blocks from the unified OPM model, accounting for dependencies between project activities and system components, as well as replacing the two monolithic and separate component-based and task-based DSM views (Sharon, De-Weck, and Dori 2012). The underlying OPM model assures model consistency and traceability. The integrated project-product OPM model includes both a diagram and an equivalent auto-generated textual description. The DSM derived from this model visualizes a dependency loop comprising both system components and project activities.

Another model-based approach (Demoly et al. 2010) employs System Modeling Language (SysML) in order to create various views that meet the needs of various system stakeholders, such as the project/process manager. The approach includes both product-oriented and process-oriented views.

Integrated Modeling of Systems and Requirements

Requirements are statements that describe operational, functional, or design-related aspects of a system. Requirements definition and management is an important SE process, as it both initiates and facilitates the entire SE effort by defining the expected functions and performance of the engineered system. Several challenges associated with requirements include:

- Defining the requirements in a structured, controlled manner.
- Tracing these requirements to system components, aspects, and decisions.
- Testing and verifying compliance of the system with these requirements.

The extension of conceptual system models to include requirements has several significant benefits:

1. Requirements provide the rationale for the system's architecture and design by making and justifying architectural and design decisions based on specific requirements.
2. Modeling the internal logic and the hierarchy and dependency relations among requirements enables identification and elimination of redundant and contradictory requirements.
3. Teams and persons responsible for delivering various

system components can often take responsibility for satisfying specific requirements. While the advantages of having good requirements engineering is clear, it is often a challenge to directly trace requirements to specific system artifacts, especially when the requirements are defined in a holistic, solution-independent manner.

There are several methods to incorporate requirements into system models, including SysML Requirements Engineering and Object-Process Methodology (OPM)-based Requirements Engineering and Authoring.

SysML-Based Requirements Engineering

The SysML requirements diagram makes it possible to capture the requirements and the relations among them in a visual manner, which is more intuitive than the textual manner in which requirements are traditionally edited and managed. The diagram was added to the basic set of UML diagrams that formed the basis for SysML (Friedenthal, Moore, and Steiner 2006), and is not a native UML diagram. Tracing requirements to the system blocks and artifacts satisfying them can be captured in the SysML Block Definition Diagram, which is primarily designated to capture the relations among types of system elements and components. The < > link between the block and the requirement captures the trace.

OPM-Based Requirements Engineering

Object-Process Methodology (OPM) is a methodology and language for conceptual modeling of complex systems and processes with a bimodal textual and graphical representation (Dori 2002). OPM’s textual representation is coordinated with the graphical representation; additionally, each visual model construct in the Object-Process Diagram (OPD) is described by a formal structured textual statement in Object-Process Language (OPL), which is a subset of natural English. OPM facilitates model-based requirements engineering, authoring, and specification, in three possible modes:

1. OPM can be used to generate conceptual models which initially focus on the requirements level—the problem domain, rather than the design level or the solution domain, which facilitates automated model-based requirements generation (Blekhman and Dori

2011). The requirements model is solution-neutral, and it can be the basis for one or more architectural solutions for achieving the functions specified in the requirements.

2. OPM can be utilized in order to generate requirement-oriented OPDs in a manner similar to how the SysML Requirements Diagram enables an engineer to capture the requirements specification as the skeleton for the system model. User-defined tagged structural relations, such as "is realized by" or is allocated to," provide for associating requirements with system model functions (objects and processes that transform them). This approach is similar to the SysML requirements diagram; however, instead of using a unique notation in a separate diagram type, the requirements are seamlessly incorporated into the single system model.
3. OPM can be used for the purpose of generating visual system models from formally specified requirements by tracing the textually authored requirements to system model inserts and artifacts (Dori et al. 2004).

Integrating Risk into System Models

Risk is an expression and a measure of the negative or adverse impact of uncertainty. Risk exists whenever uncertainty can lead to several results, of which some may be negative (adverse) and some positive. A system faces risks from other systems or from the environment, and it can also pose risks to other systems or to the environment. Systems are characterized by such attributes, such as: goals, objectives, inputs, outputs, variables, parameters, processes, events, states, subsystems, interfaces, mechanisms, and methods. System vulnerability is the system's total potential to be harmed or negatively affected in any one of these attributes. Analogously, system harmfulness is the system's total potential to harm others or to generate negative effects, which can be manifested in one or more of these attributes (Haimes 2009). Model-based risk analysis (MBRA) enables structured analysis and risk-related process control. Several model-based risk analysis approaches are available in the literature. MBRA is presently more common in the information technology and information security domains than in the systems engineering domain; however, some of the methods are generally applicable to complex systems as well. The ISO-IEC-IEEE collaborative software development and operation lifecycle standard (ISO and IEC 2004) proposes a concurrent approach to IT Risk Management. This approach consists of six main activities:

- Plan and Implement Risk Management
- Manage the Project Risk Profile
- Perform Risk Analysis
- Perform Risk Monitoring
- Perform Risk Treatment
- Evaluate the Risk Management Process

These activities are executed concurrently, affect and provide feedback to each other, and interact with other software life cycle processes, such as the technical management and the design processes (ISO and IEC 2004).

The CORAS approach (Fredriksen et al. 2002; den Braber et al. 2006; Lund, Solhaug, and Stølen 2011) is a UML-derivative for IT security risk modeling and assessment. This framework consists mostly of the UML use case (UC) diagram, extended for misuse cases. Additional notation was added to the UC notation in order to capture risk sources, effects, and results (e.g., the “bad actor” icon, moneybag for asset-in-risk). A misuse diagram can include, for example, the risk of loss of legal protection of proprietary know-how due to information theft and distribution by an unfaithful employee. The treatment for the risk source of insufficient security policy, which contributes to the above risk, is illustrated in a separate treatment diagram.

A quantitative risk assessment method for component-based systems (Grunske and Joyce 2008) supports component vulnerability analysis and specification using modular attack trees. In addition, it provides attacker profiling, which enables supporting econometric approaches to risk response. The methodology utilizes SysML as its underpinning language, especially the SysML block definition diagram and parametric diagram, in order to capture parametric relations and constraints as a means to defining risk profiles.

System-Theoretic Accident Model and Processes (STAMP) is a method for system and component design for safety (Leveson 2011). STAMP reformulates the safety problem as a control problem as opposed to a reliability problem. STAMP is optimized for safetyoriented systems engineering and design and for hazard avoidance and mitigation, specifically in complex socio-technical systems. A model-based adaptation of STAMP was also proposed (Leveson 2004) and was implemented in various safety-critical and mission-critical systems, including aircraft collision avoidance systems (CAS) (Leveson 2004) and ballistic missile defense systems (Pereira, Lee, and Howard 2006). Risk-Oriented Systems Engineering (ROSE) (Mordecai and Dori 2013) is a method based on Object-Process Methodology (OPM) for integrating risk into system models. Being system-centric, ROSE is responsible for capturing risk layers and aspects on top of and in sync with the core system model, while improving and immunizing it against captured risks, as well as for generating system robustness and resilience by design in response to various risk-posing scenarios. The risk handling meta-model includes risk mitigation during the design phase and risk response during the operational phase.

References

Works Cited

Blekhman, A. and D. Dori. 2011. “Model-Based requirements authoring - Creating explicit specifications with OPM,” in 6th International Conference on Systems Engineering. Herzeliyya, Israel.

Browning, T.R. 2001. “Applying the design structure matrix to system decomposition and integration problems: A review and new directions,” *IEEE Transactions on Engineering Management,* vol. 48, no 3, pp. 292–306. Available at: IEEE http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=946528, and doi:10.1109/17.946528. Accessed December 4, 2014.

Demoly, F., D. Monticolo, B. Eynard, L. Rivest, and S. Gomes. 2010. “Multiple viewpoint modelling framework enabling integrated product–process design,” *International Journal on Interactive Design and Manufacturing (IJIDeM),* vol. 4, no. 4, October 12, pp. 269–280. Available at: Springer http://link.springer.com/10.1007/s12008-010-0107-3, and doi:10.1007/s12008-010-0107-3. Accessed December 4, 2014.

Den Braber, F., G. Brændeland, H.E.I. Dahl, I. Engan, I. Hogganvik, M.S. Lund, B. Solhaug, K. Stølen, and F. Vraalsen. 2006. *The CORAS Model-based Method for*

Security Risk Analysis. Oslo, Norway: SINTEF.

Dori, D. 2002. *Object-Process Methodology – A Holistic Systems Paradigm*. New York, NY, USA: Springer-Verlag.

Dori, D., N. Korda, A. Soffer, and S. Cohen. 2004. "SMART: System model acquisition from requirements text," Lecture Notes in *Computer Science: Business Process Management,* vol. 3080, pp. 179–194. Available at: Springer http://link.springer.com/chapter/10.1007/978-3-540-25970-1_12. Accessed December 4, 2014.

Estefan, J. 2008. *A Survey of Model-Based Systems Engineering (MBSE) Methodologies,* Rev. B. San Diego, CA, USA: International Council on Systems Engineering. INCOSE-TD-2007-003-02.

Friedenthal, S., A. Moore, and R. Steiner. 2006. "OMG Systems Modeling Language (OMG SysML™) Tutorial" (July).

Golany, B. and A. Shtub. 2001. "Work breakdown structure," in Salvendy, G. Ed. 2001. *Handbook of Industrial Engineering, Technology and Operations Management,* 3rd ed. Hoboken, NJ, USA: John Wiley & Sons, Inc., pp. 1263–1280.

Grunske, L. and D. Joyce. 2008. "Quantitative risk-based security prediction for component-based systems with explicitly modeled attack profiles," *Journal of Systems and Software,* vol. 81, no. 8, pp. 1327–1345.

Haimes, Y. 2009. "On the complex definition of risk: A systems-based approach," *Risk Analysis,* vol. 29, no. 12, pp. 1647–1654. Available at: Wiley http://onlinelibrary.wiley.com/doi/10.1111/j.1539-6924.2009.01310.x/full. Accessed December 4, 2014.

ISO/IEC/IEEE. 2004. *Systems and Software Engineering - Life Cycle Processes - Risk management*. Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), ISO/IEC/IEEE 16085:2006.

Leveson, N.G. 2004. "Model-Based Analysis of Socio-Technical Risk." Cambridge, MA, USA: Massachusetts Institute of Technology (MIT) Working Paper Series. ESD-WP-2004-08.

Leveson, N.G. 2011. *Engineering a Safer World: Systems Thinking Applied to Safety*. Cambridge, MA, USA: MIT

Press.

Lund, M.S., B. Solhaug, and K. Stølen. 2011. *Model-Driven Risk Analysis: The CORAS Approach*. Berlin and Heidelberg, Germany: Springer Berlin Heidelberg. Available at: Springer http://www.springerlink.com/index/10.1007/978-3-642-12323-8, and doi:10.1007/978-3-642-12323-8. Accessed December 4, 2014.

Mordecai, Y., and D. Dori. 2013. "Model-Based risk-oriented robust systems design with object-process methodology," *International Journal of Strategic Engineering Asset Management,* vol. 1, pp. 331-354 (CESUN 2012 Special Issue).

NASA. 2007. *Systems Engineering Handbook.* Washington, D.C.: National Aeronautics and Space Administration (NASA), NASA/SP-2007-6105.

Pereira, S.J., G. Lee, and J. Howard. 2006. "A system-theoretic hazard analysis methodology for a non-advocate safety assessment of the ballistic missile defense system," Vol. 1606. Available at: Defense Technical Information Center http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA466864. Accessed December 4, 2014.

Rad, P.F. 1999. "Advocating a Deliverable-Oriented Work Breakdown Structure." *Cost Engineering,* vol. 41, no. 12, p. 35. Sage, Andrew P., and William B. Rouse. 2011. *Handbook of Systems Engineering and Management*. Hoboken, NJ, USA: John Wiley & Sons.

Sharon, A., O.L. de Weck, and D. Dori. 2012. "Improving project-product lifecycle management with model-based design structure matrix: A joint project management and systems engineering approach," *Systems Engineering,* vol. 16, no. 4, pp. 413-426. Available at: doi:10.1002/sys.21240.

Sharon, A., and D. Dori. 2009. "A model-based approach for planning work breakdown structures of complex systems projects," in Proc. 14th IFAC Symposium on Information Control Problems in Manufacturing.

Wand, Y., and R. Weber. 2002. "Research Commentary: Information Systems and Conceptual Modeling-A Research Agenda," *Information Systems Research,* vol. 13, no. 4, December, pp. 363–376. Available at: doi:10.1287/isre.13.4.363.69.

Primary References

Dori, D. 2002. *Object-Process Methodology – A Holistic Systems Paradigm*. New York, NY, USA: Springer-Verlag.

Estefan, J. 2008. *A Survey of Model-Based Systems Engineering (MBSE) Methodologies,* Rev. B. San Diego, CA, USA: International Council on Systems Engineering. INCOSE-TD-2007-003-02.

Golany, B. and A. Shtub. 2001. “Work breakdown structure,” in Salvendy, G. Ed. 2001. *Handbook of Industrial Engineering, Technology and Operations Management,* 3rd ed. Hoboken, NJ, USA: John Wiley & Sons, Inc., pp. 1263–1280.

INCOSE. 2012. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. Version 3.2.2. San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-03.2.2.

NASA. 2007. *Systems Engineering Handbook.* Washington, D.C., USA: National Aeronautics and Space Administration (NASA), NASA/SP-2007-6105.

Additional References

Kristiansen, B.G. and K. Stolen. 2002. “The CORAS framework for a model-based risk management process,” in *Lecture Notes*, S. Anderson, M. Felici, and S. Bologna. Eds., vol. 2434, pp. 94–105. Berlin and Heidelberg, Germany: Springer-Verlag. Available at: doi:10.1007/3-540-45732-1_11.

< Previous Article | Parent Article | Next Article >

SEBoK v. 2.10, released 06 May 2024