Benefits and Challenges of Model-Based Systems Engineering

Featuring Nataliya Shevchenko and Mary Popeck

Welcome to the SEI Podcast Series, a production of the Carnegie Mellon University Software Engineering Institute. The SEI is a federally funded research and development center sponsored by the U.S. Department of Defense. A transcript of today's podcast is posted on the SEI website at sei.cmu.edu/podcasts.

Suzanne Miller: Hi, my name is Suzanne Miller, and I want to welcome you to the SEI Podcast Series. I am a principal researcher in the SEI Software Solutions Division. Today, I am very happy to be joined by Natasha [Nataliya] Shevchenko and Mary Popeck, who are both engineers in the SEI CERT Division, our sister division. Today we are going to talk about model-based systems engineering [MBSE], which is one of my favorite topics. So, I am really happy to have both of you here. Natasha and Mary, welcome to the podcast. Let's start off—before we get into model-based systems engineering—people like to know a little bit about our speakers, and let people know what is it you do at the SEI and what is it that is involved in your day-to-day work. I am going to start with Mary, and why don't you let us know?

Mary Popeck: Okay. Well, hi Suzie. It is a pleasure to be here today. I came to the SEI to better understand and research and improve the state of software and engineering practices, as well as to assist our customers with enhancing their acquisition and security methodology when they build software-intensive systems and systems of systems. At the SEI, I have worked with a large variety of customers, such as DHS [Department of Homeland Security], the U.S. Air Force, Army, the intel community, VA [Department of Veteran's Affairs], the FDA [Food and Drug Administration], and even some commercial entities such as Roche and Westinghouse. We go in as an SEI team, we go to our customers, and we evaluate their methods, their architecture, and help them better understand their current state, their current practices and then what are the gaps with any accepted best practices or standards and how to better mitigate any resulting risks that we find.

Suzanne: Good. That has a natural connection to systems engineering, and we will talk about that in a minute. Natasha, what is it that you do at the SEI?

Natasha Shevchenko: Thank you, Suzanne, for hosting this podcast. It is a pleasure to be here. I really appreciate the opportunity to talk about model-based systems engineering on this platform. In SEI, I work with different kinds of customers to help them to move…using model-based systems engineering, to use digital engineering to adopt [in] their processes, including Agile development process with their engineering practices. I help them to start [to] model their system to understand how modeling affects their work, help them to do better development, to analyze their system, and increase the quality of systems. This is one aspect of my work. Another aspect, since I work in CERT, I work with cybersecurity and threat modeling. Model-based systems engineering has a part in this area as well. We can model the risks on the system, system architecture, and increase the resiliency of the system using model-based systems engineering by actually following the mitigation strategy and analyzing the system consistently. These two areas come together nicely, and I hope we will be able to address at least some of these topics during this time with you.

Suzanne: Excellent. You guys make a good partnership. We have got the architect. We have got the modeler, so together, you guys can create synergy for our customers in understanding their situations. That is always the first step to getting better, to understand where you are. When we talk about model-based systems engineering, I know some of our audience is new to this topic. And I also know that model-based engineering, model-based systems engineering, and digital engineering are all somewhat related. You will hear those terms a lot in the industry and in the government. Why don't we start by having you give us a little bit of what is the difference among those three terms, and why should we care about that difference? Whoever wants to answer that, go ahead.

Natasha: Let me start a little bit. For example, how model-based systems engineering and digital engineering [go] together. Model-based systems engineering is not a very new idea. First, models, regarding engineering, happened in the late '40s. It became really popular in the end of '90s, but most companies that used it [were] big companies with huge security risks and safety risks. They use model-based systems engineering as a continuation of their model-based engineering. But, with new tools that start to appear in the mid-2000s, digital engineers started to play a bigger role, allowing to move these models from paper-based, document-based modeling into digital environment and allow us to make the system analysis repeatable, faster, and to do validation and verification of a system on a regular basis on a digital representation of our model. It actually improves the ability to cope with complexity with systems that was almost not possible to do on a document-based. As engineers and system engineers, we needed the help of tools to do model-based systems engineering and analysis. Digital engineering, technically, it's moving your architecture, your models, your model-based architecture into a digital environment to use all of its [power] to help you with complexity and necessity to analyze your system regularly and have repeatable processes while you are doing that.

Suzanne: Mary, did you want to make any comments on that?

Mary: Yes, I would. To me, model-based systems engineering is a subset of digital engineering that supports the system-engineering activities and functions, such as requirements engineering, architecture analysis, design, verification, and validation. So, it's the use of a digital model, but it may, in fact, be multiple models from model-based engineering that are integrated together into one model that allows you to perform the system-engineering functions. By putting them all into one model, it requires more formal methods than what you would do with document-based systems engineering. It adds some rigor and precision. It also helps communicate with the team much better, because now you have got a way to graphically and mathematically represent the system, so that all of the stakeholders have a common system viewpoint. [Editor's note: A purist might argue that MBSE can be done using paper; and that is what we are referring to as DBSE. The term MBSE is used assuming it is done in a digital environment and DBSE encompasses MBSE in a paper environment.]

Suzanne: So, if I were to, for our audience, the way I would frame it is, in the past, the models…If you wanted to analyze one or more models, you had a lot of different, you had a lot of paper to look at. You had many things that, you know, I remember the days of the great big posters, right? There were the printouts on the CAD system of your model. So, anytime you're in that situation, anytime you find something that, *Wait, that's not right,* or *Wait, we need to change that* or *Wait, we implemented that differently*, we have got a big cycle to go through to actually make changes. With digital engineering as the home, if you will, for models that are developed using model-based systems-engineering techniques, now we have got that dynamic. Now we have a 72-inch television instead of the poster. We still need a lot of visibility, but we have that ability much easier to change, to talk about what happens if we do this, we can do what-if analysis and things like that, that were not possible or were not easily feasible in the past. Is that a pretty decent characterization?

Mary: Right, for example, if an engineer working in one discipline, maybe thermal or structure/mechanical engineering, wants to add something or change something; they have a new requirement they want to implement, before, with just model-based engineering or document-centric systems engineering, they would make the change and analyze their own domain, their own discipline. But, now when you put it into model-based systems engineering and everything is in one model, you can now see the impacts across all the different disciplines that are in the system and have a much earlier understanding of what all the impacts are before you actually start building and implementing. And then you don't find out the impacts till much later in test.

Suzanne: So, for the example you just gave, if I make a change in thermal, then that could very well imply a materials change. I may be going outside the boundary, the heat signature of the materials that I had planned to use, somebody needs to know that. In the old days, that was the kind of thing that would…might fall through the cracks, where when we have these models together in the same environment, it's not always guaranteed it'll happen, but it's much more likely that you'll be able to pick up that kind of thing. Excellent.

Natasha: And, you can add the boundaries between hardware and software parts of a system.

Suzanne: Yes.

Natasha: Usually even bigger than between two areas of hardware, for example, material science. They operate almost in silos. Even taking this specific example, the monitoring system that is supposed to monitor the temperature will be factored [in] too.

Suzanne: Yes, yes. So, we have been talking about the advantages of using model-based systems engineering. That is not what we started, but we ended up there. Mary, do you want to talk about some other things besides change management that make model-based systems engineering…gives it an advantage over document-based systems-engineering practices?

Mary: Yes. I would like to say that when you use document-based systems engineering, humans are managing the documents; they have to keep them updated, keep them consistent, keep them accurate. And, we all know that humans aren't as good as computers at doing the detailed, mundane tasks. You know, humans, we forget things, we skip steps, we're in a hurry, we make mistakes. This human tendency introduces defects and vulnerabilities, and even confusion about what the system really is. So, with model-based systems engineering, the record of authority now shifts away from the documents to the digital model. So, a human only needs to put their information in once in one place. This allows the model to be quickly updated versus document-based systems engineering. And, all the viewpoints for all the stakeholders, they are all consistent. An additional thing is, if a modeler makes a mistake when they're putting the change into the model, the application can alert you if you violated a rule or a framework violation. As well, you can also create tables to help you find those mistakes that are often made. So it's so much easier to find defects, and any defects that are injected, you're going to find them even sooner than you would. You don't have to wait until the test phase to find your defects. I think another big advantage is all your interface problems. You are able to model those early and flush out interface problems. You don't have to wait until you get to integration and testing where it is so much more costly and time-consuming.

Suzanne: Natasha, you had a couple of things I know you wanted to talk about on that as well.

Natasha: Yes. I would like to touch a couple of points. One of them is impact analysis and validation, verification of a system in general. In digital environment and model environment, you can script it up, it can be done the same way for everybody, regularly, overnight, and make it continuous. Another aspect is we can do simulation on the system even before it was implemented at all. So, if your model is mature enough, you can simulate the condition of your system, tweak and change different parameters, see how your system behaves with the different conditions. You can simulate even different architectural alternatives of your system, and see how it changes under that and support the different kinds of -ilities you have in your requirements for your system. All of these that Mary mentioned, and simulations, move your architecture and design to the next level. A lot of mistakes that now and recently were caught only after development and during the testing, can be caught relatively early.

Suzanne: So, what you are moving into is talking about how the model-based systems engineering not only improves the product, but actually improves the practices themselves. Mary mentioned, in terms of the model becoming the record of authority, and giving people a single source of truth as we talk about it, things like that, that allow our architects and our engineers to actually spend more time on the analysis and getting it right. The simulations allow us to do a lot more what-if analysis and things like that. What are some other development-practice improvements that you have observed. I'll start again with Mary, in terms of seeing how systems that are composed, especially systems composed of both hardware and software, how it's improved the development practices in that arena?

Mary: I would say, expanding on what you already mentioned, that doing system analyses, such as performance and timing. You have to have the entire system to do that, all the hardware and software to do a good job with your performance and timing analyses and trying to figure out how to improve those. So, model-based systems engineering helps with that. Where it can also help you is with your quality attributes to a certain degree. If you want to make a change to a feature or a requirement, you can find out across the system, how is this going to impact security? How is it going to impact performance or some of your other quality attributes? You have a lot more insight early on. It also enables you to visualize your requirements allocation, and how you are flowing requirements down or even functional allocation to all the different components. And then you can do tradeoffs. Maybe I want to change how I allocated the requirements between hardware and software components, you can play with that. And, it also allows you, when you're doing these trades and making decisions, you can capture all of that information in the model. There are plenty of places to put notes and descriptions. So, it's all there. It is not just written down on a piece of paper in somebody's desk or safe somewhere. It is now captured in the model that anybody can find in the future if they want to understand why you did certain things.

Suzanne: The thing that I have noticed in working in some of my customer spaces that use model-based systems engineering, is that aspect of, *I can find notes from other people*. That's one that is really, really big in terms of answering the *why* question. Because as a systems engineer, you ask a lot of *why*s. So, answering that *why* question is important. But another one that I've run into is answering the *when* question, because one of the things I've seen models help with is, *When do I need the hardware so that I can have the software integrated with it? What if I change my hardware delivery? Which pieces of the software now don't have anything to integrate with and test with, and can I do some lower fidelity stuff with the model, do I need to add a breadboard in here*? There are some programmatic aspects of this that modeling helps with, as well as the more technical things. I just wanted to bring that up, because a lot of times people think that model-based systems engineering is just for the really hardcore engineers. But, that connection to, *When do we need things integrated?* really gives it more importance in the programmatic space as well. Natasha, did you have anything you want to say about that? We didn't talk about this before, but if you have any comments on that, I'd be interested in hearing them.

Natasha: Yes. I think recently, especially recently, the implementation of different architectural frameworks using SysML became really popular because it allows you to do strategic planning in the model early on. So, even your conceptual view of your system, very high level and on an enterprise level, including organizational structure, you can put in the model and connect to everything downstream, to the behavior of a system, to the solution of a system before implementation, and then to implementation. With DODAF [DoD Architecture Framework], UAF [United Architecture Framework], TOGAF [The Open Group Architecture Framework] architectural frameworks with enterprise level use in the model environment, it allows enterprise architects and management to see their view of the system as well as the engineers and system engineers. They are right now very eager to see this traceability, because most of the time PMOs [program management organizations] and management really lack requirements traceability to a solution. They really want to see that none of their requirements fell through the cracks or were implemented in the wrong place or wrong time like you said. And model-based systems engineering in a digital environment actually allows them to do that, to be in control of that, to plan the capabilities on months [ahead], and if they work with Agile, into [program increment] PIs, and see exactly how it will transfer to the low level of engineering on the system levels and implementation levels of the systems. And it allows us to do system analysis on that level as well. Not on a lower level, but on the higher level of, you have a system, so when the architecture is ready for implementation, it's developed much better, with small details necessary for a solution to be developed for implementation to happen. I want to talk one more time about the model being a communication tool, between very different levels of engineers, management, and the people involved in the creation of a system, especially large system.

Suzanne: Well, and we work with some, you work with some very large systems. That ability to communicate across a wide swath of stakeholders is something that we have to find ways to do because we have a lot of them. I know that one of the things you really didn't mention was that all of this is served by having common access to the tooling environments that allow us to not have to print out a version of a model to give it to stakeholder X or Y or Z. The other aspect of digital engineering is making the digital environment accessible to all the people that need to use it. And, I am going to put a plug in for people that haven't thought about it until now, but learning SysML is not that difficult in comparison to a lot of other modeling languages. And, that is one of the most common languages that is used to articulate models, and that is one of the ways that you can become an active stakeholder in the model-based evaluation and model-based validation. I am putting a plug in for people to up their game on learning modeling languages. Even if you don't write any computer languages, learning SysML is probably not a bad idea. So, I will leave it at that.

Security. So you both work in CERT, CERT is the home of computer emergency response teams, cybersecurity engineering, etc. What are some particular things that model-based engineering allows you to achieve, so that you can get closer to the state that we're all looking for of secure by design? Mary, Natasha, who wants to take that one?

Natasha: Mary, go ahead.

Mary: OK. So, here at CERT, we work to assist or coach our industry and government customers to help them reduce security weaknesses and make their software-intensive systems more resilient. MBSE helps with this because you are putting all the information and artifacts, they are all maintained in one place. So, there is a single source of truth. This allows you to reduce and mitigate security risks early in the lifecycle. We all know risk identification and mitigation is most effective when it's done early in the lifecycle. This allows the developers to add in security right at the beginning of the design. They are not tacking it on at the end. And, they can try out different security techniques, figure out which one fits best in their system. Otherwise, you've already built the system, and you're trying to put something on. That is not as useful.

Suzanne: OK. It sounds obvious, but the ability to access both the aspects of the system that have security needs, from a quality-attribute viewpoint, and be able to do that what-if analysis inexpensively in comparison to actually implementing something, that continues to be one of the things I hear about in terms of people's desire to do more of this in a model-based systems engineering environment.

I want to ask a question that is always something that we ask in our podcasts, which is about transition. So, model-based systems engineering, yes, I can go to INCOSE [International Council on Systems Engineering], I can learn SysML, but what, if I'm new to this area, are some SEI resources or other resources that are available. How do I learn more about this, because I have decided that I need to be an active stakeholder now. So, where do I go?

Natasha: If I may touch this question, I think the number one resource outside of SEI would be OMG [Object Management Group] resources. They have an MBSE wiki with a lot of material on that. They have a SysML site, including the certification if somebody wants to be certified as a modeler or a model user. And even without doing certification, just go through the material they provide for training for certification will give a lot of information as a starter. I understand that actually, there is not a lot of information out there, how to start with modeling. It's a few books that tell you how to use SysML [here and here], and you can find videos here and there showing specific aspects of modeling, using different tools available, modeling environments, available right now. I think it's time to, for us, as the SEI, to chip in to give people initial help, how to start to think about, *Do we need model-based systems engineering in our company?* How we do that? where we start, who we need to train or not. This is right now pretty much nobody knows.

Suzanne: That should be in your job jar, start working on some of these things. What are the things that you and Mary and the group working in model-based systems engineering are targeting as near-term research and transition areas in the space?

Mary: I have been building a digital-engineering model of a generic federal agency that needs to perform defensive cyberspace operations, DCO. My coworkers have also been working in this model that I have been building. They are reviewing it, adding elements, changing things based on their knowledge and experience, which is different than mine. What we are hoping—we are building up this model—we hope to tailor it for each of our customers. If a customer comes to us, says, *I'm doing DCO, how do I do it better?* Well, we want to take the model and tailor it and model their particular DCO activities and then help them identify where do they have security gaps? Maybe they didn't implement certain security controls. Maybe they didn't implement them properly. Now we can make recommendations to them, help them improve their security posture and resiliency. We will be able to make recommendations to change their system. As we use this model with our customers, we will learn more. We will update the model, then the model will become more and more realistic and more and more useful for our customers.

Suzanne: So that model will go into Natasha's class. But if we want to educate people, having those kinds of reference models are often critical in helping them to be able to make it real. We can put up all the pretty slides we want, but until you actually touch the model, you are not going to see what is actually feasible.

So, you've got some things like that. Are there opportunities for people out in our audience to collaborate with you, can they get in touch with you if they're interested in saying, *Hey, I'm in a DCO position, I'd like to look at a model like that*, or *We're not used to SysML but these things, X, Y, Z*. Is that something that as research you are looking for, as you move forward?

Mary: Yes, we are certainly looking for a collaboration with people working in this area. Please, if you would contact info@sei.cmu.edu, please include my name and Natasha's name, Mary Popeck and Natasha Shevchenko, in the email that you send, and we will definitely get back to you.

Suzanne: All right. It's always good to have people that are going to help you with those kinds of things. As you said, humans are not perfect, and we always forget things. I love it when I have people that can help us with what we're moving forward.

Natasha, Mary, I want to thank both of you for being here today and talking about this work. This is an area that I also have a passion for and do with a lot of customers. I am glad to be able to talk about it. To our listeners, I want to thank you all for joining us today. We will include links in our transcript to resources that we have talked about here as well as resources that you may want to look at from outside the SEI as well.

This podcast is available on the SEI website at sei.cmu.edu/podcasts, and all the other places you get stuff like that. Thank you for watching. Go forth and model your systems, don't just talk about it. Thanks very much.

Thanks for joining us. This episode is available where you download podcasts, including SoundCloud, Stitcher, TuneIn Radio, Google Podcasts, and Apple Podcasts. It is also available on the SEI website at sei.cmu.edu/podcasts and the SEI's YouTube channel. This copyrighted work is made available through the Software Engineering Institute, a federally funded research and development center sponsored by the U.S. Department of Defense. For more information about the SEI and this work, please visit www.sei.cmu.edu. As always, if you have any questions, please do not hesitate to email us at info@sei.cmu.edu. Thank you.