Digital Engineering and DevSecOps

featuring David Shepard as Interviewed by Suzanne Miller

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: Hello, my name is Suzanne Miller. I am a principal researcher here at the SEI, and I am joined today by my colleague Dave Shepard. We both work in the CDC Directorate of the SSD, the Software Solutions Division of the SEI.

Welcome David. I am excited to be here to talk with you about digital engineering and how it intersects with DevSecOps or development/security/operations as we know it. Before we get into that topic, our viewers like to know, Who are you? What is it that brought you to the SEI, and what is it that brought you to working in digital engineering because that is not necessarily the thing we immediately think of when we think of DevSecOps. So, over to you.

David Shepard: Digital engineering is a part of my current work as we try to help the DoD develop their plan for how digital engineering will be implemented across the department.

Suzanne: That is very important. We know that there is a congressional National Defense Authorization Act from a couple years ago. It was really what introduced this; we want to move in this direction throughout the DoD. Before we get into digital engineering and DevSecOps, what is digital engineering? There is an argument that says, *Well what is the SEI doing in digital engineering*? *We are software engineering and cybersecurity.* This seems like this is a higher-level thing. Why are we doing this kind of work?

David: The textbook definition that we are using for digital engineering is it is an integrated digital approach that uses authoritative sources of systems data and models as a continuum across disciplines to support lifecycle activities from concept through disposal. What that really means is you are developing models for everything, not just for your software, but for all of your components of your system of systems (SoS), hardware and software. You are storing those models in a singular repository of knowledge, both the models and the data that you use when you work with those models. Those models will be the single source that is used by all contractors and everyone working on the project.

Suzanne: I can certainly argue that SEI knows about modeling. We have things like Architecture Analysis and Design Language [AADL] that are model-based kinds of engineering supports, but what is it that makes this a problem that the SEI really needs to work on? This is where I see the DevSecOps connection, because I see connections there, and so let us talk about them.

David: Yes, DevSecOps is thought of traditionally as like a software process. That is fair because you are trying to integrate development and operations, and you are trying to do so in a way that ties in security throughout the lifecycle of that development very software intensive, but our systems today are neither all software nor all hardware. There is always some overlap. We have been looking at ways of integrating hardware-integration concepts into our DevOps and DevSecOps thinking for a while now. It is an interesting take coming from the other perspective, starting from the hardware mindset and looking at how to roll everything in including the software. They are different approaches, how you go about doing that. We can talk about that a little bit later.

Suzanne: That argues that the digital engineering, we want models—software-based, obviously models—and so somebody has got to develop that. Somebody has got to develop the repository. Somebody has got to develop the DevSecOps pipeline that is going to generate data about how all this stuff is working in the software especially. So, there is this confluence. There is a lot of data that comes out of both of these areas, and there is a need for disciplined engineering in building the models, building the modeling languages, the modeling platforms, etc. What is your role in working through all of that?

David: Can we talk a little bit about the differences in modeling of a hardware system versus what we typically do in software today? I think that is a good place to start. When we think of software modeling, a lot of people think that you are going to use SysML or something like that. I would argue that a lot of work goes into modeling software today across the industry that actually doesn't involve a formal modeling or graphical language like SysML. There are a lot of different ways to capture a model, a lot of different ways to capture requirements, and our processes that we use for developing software today, like Agile, have a lot of tools that help with capturing models and information that is useful for developing the plan for how you are going to build something. I wanted to say that because it is important to recognize that there are different ways that modeling happens today in software that do not necessarily look like what people expect software modeling might look like, the SysML and tools like that. Sorry, what was your question?

Suzanne: What is your role in integrating that digital engineering, DevSecOps perspective, so that we get something that is disciplined, secure, and covers that whole gamut?

David: From my perspective, I work in DevOps, so I am focused on infrastructure and tooling to help bring that pipeline of continuous integration and delivery together. From where I sit in the process, it actually looks like a question of how digital engineering and DevOps will share resources and share information. For example, the DevOps world, the software-development world has kind of settled on Git as a de facto version-control system.

Suzanne: Right.

David: Git would not work real well for many digital-engineering artifacts. They tend to be rather large, and they have their own tools for managing versions of their models. They are version controlled, and they need to reside somewhere. Where we tend to see them coming together is like on the cloud infrastructure that you are deploying your solution to. They can both live in the same space is the basic gist of that. Then it becomes a question of how much crossover you need between the two environments and how you bridge those. OpenMBEE is a repository system for digital-engineering artifacts as opposed to Git.

Suzanne: That is actually one of the things that I am seeing in at least one of the customers that I work with is that the tooling that is being used to build the platforms and the data repositories for digital engineering are coming from a different source and different idea base than what has been used for DevOps. So, this is one of the challenges that the SEI is really a good place to deal with, is helping people to see where are the disconnects and what are the ways that we might approach the kind of integration that is needed and where do we not need it? Because that is the other thing that I have learned in working in different modeling spaces, is that not everything needs to integrate to the same level. So, finding out where we need that data to integrate is a really important part of this problem. The DevOps folks have a perspective on the data we need to get stuff through a pipeline. That is not necessarily the same task as informing, getting the results for example of that pipeline into new parameters and a model.

David: Yes, so digital engineering is trying to capture the entire lifecycle of the product. You can think of it in a sense as an extension of what used to be called product lifecycle management or PLM that started in places like the auto industry. That extension might seem natural from a hardware perspective to manage all of the artifacts and how things have changed over time with the hardware. That may seem natural to the hardware world. It is not something that is quite so natural or has been captured quite so rigorously with software. One of the things that I like to say is the best and worst thing about software is that you can change it.

Suzanne: Sure.

David: That is one of the things that has for a long time differentiated software from hardware systems. But we are trying to change that with digital engineering. We are trying to make it easier to change hardware. That starts with modeling. You can iterate faster on things when it is a model in software than you can if you actually have to instantiate a hardware version of that thing that you are building.

Suzanne: So that brings us to the idea that you talk about in the blog post, and it is out there quite a bit in the world, this idea of a digital twin as being one of the focal points. That is the mecca in some ways of digital engineering is accomplishing a model that is a digital twin of the actual system as it is built. That has an impact on testing requirements. We have already in the DevSecOps world come up with ways of fast testing of requirements and fast feedback of testing in the CI/CD world, continuous integration/continuous delivery world. How does the digital twin idea resonate with DevSecOps, or how is it different in a way that we really need to pay attention to and not as I am doing right now, saying, *Well, we use some of the same ideas that we get fast test results in DevSecOps, we will use those ideas in digital twins*. Can you speak about that for a little bit?

David: Yes. Let us start with the digital twins. So, Dr. Roper if you follow him, he…

Suzanne: We all do.

David: Yes, he would say that you have to go all in if you want the maximum benefit out of digital engineering. What I take that to mean is you should have a digital model for everything in your system. If you do that, then you can simulate your entire system at any time and work on that integration, and really get the robust testing and iteration on that system that you are developing. If you do not have the full model, or there are parts of it that have to be actually instantiated in hardware to do testing on them, then that slows your design chain, and your ability to change things down. Modeling in software tends to look fairly different. When you think simulations in software, it is usually like a videogame sort of effect, or like a car being driven around a closed track because you want to gather data or simulate some effect of a system. That is not typically how we think about building software. The product of software is a system that does something for you, but the modeling of that thing is not a graphical output for you. So, our simulations in software tend to be unit tests. You are testing some functionality of the system.

Suzanne: So, you are looking at a timing issue. You are putting a trigger in place and you are running it through and looking at how long does it take, for example, is a simulation…

David: *Does it perform as expected? Does it do what it is supposed to do?* Testing and writing different sorts of tests is how we typically do simulations of software systems. Not the whole, *Build your system in a modeling application and then run it through some series of events that you can visualize*. The output is log files and things like that. So, it is simulation and it is testing, and you are getting similar effects, but the outputs look very different.

Suzanne: When we start thinking about digital engineering, and we bring that perspective, I will call it the *video-game perspective of visualization*, really what you are talking about when you talk about videogame, you are talking about an ability to actually visualize the physical space that the product, whatever it is, is running in. That is different than how we think about it in software.

David: It is.

Suzanne: We may not want to use exactly the same methods or expect the same things when we try to integrate our software models into that digital twin, for example.

David: Yes, and one of the unique things that I think you get out of the simulation when you are doing a digital twin is that it is not just the visualization. You are also getting the data logged to log files and things like that, like you would get with a software simulation. In some ways, it is probably more complete in that regard because you can actually look at your simulation in more dimensions. The effectiveness of your simulation though really comes down to how much effort you put into building that model. There is a place of, *Where do you stop? How much fidelity do you need to simulate the system that you are trying to build?* I do not know that that is a question anyone really has a good answer to. We have seen jets built from inception all the way down to the full product, delivered as digital models. They can then go and build that actual jet off of that system. But you can still, just like anything that you are simulating it—your ability to test it and understand where the problems are and your ability to resolve them, still comes down to the people who are working with those models.

Suzanne: Well there are…I mean the classic example is satellites. I can't have really, really high fidelity in the simulation of the effects of space. I have some level of fidelity I can achieve, but there is some stuff that we are going to have to use other analytic methods to predict how things are going to happen after we launch. We have to launch to be able to get some of that data, and that is expensive, and that is why we are doing this. I think that is the tradeoff you are talking about in terms of digital engineering is, How much money do we want to spend while we are essentially on the ground, whatever we are doing, before we actually put our system into its intended environment where we do not have control over a lot of things, and where it is very expensive to say*, Oops, I am remembering the movie, what do you mean oops? Independence Day* is the movie I was remembering. We don't want a lot of the *oopses* out in the world, so we are actually, as our technology becomes more adept at doing those simulations, we want more fidelity in those models so that we do not have as many *oops* moments when we are out in the environment.

That being said, what do you think are the "big rocks" in getting to a place where we really trust digital engineering both for the hardware and trust it as one of the elements that we use for data in our DevSecOps pipelines? What are some of the research challenges that we really…If I am somebody out in the world, I need to be aware these are kind of my caveats, we are not there yet in some of these areas. What are some of the things you are seeing?

David: There are a ton of research areas that are potential. One of the big ones from a technical perspective is there still is not really an agreed-upon definition of, for example, what is a digital twin? What is required for something to be called a digital twin? What do the file formats need to look like? How does tooling tie into these things? Because over time, if you are building a model that you want to last for the lifecycle of a product, which could actually be a very long time, you would prefer that that is something that the model, the data itself especially if it is very large, can actually survive over time as your tooling and your needs for how you work with that model evolve. There are a lot of open questions there as to how to do that. Some of these models can get very large. So, we would like to get that right.

Suzanne: That argues for some of the research that people are doing in data, data-science research, and how do we preserve that data in a way that is going to be useful over tooling changes and other kinds of environment changes. How do we get stability in this very uncertain technology space, quite frankly?

David: Yes, technology never stops changing. I wanted to say one thing though about your previous question. You were talking about how from the hardware perspective you have to deal with modeling and fidelity, and when is the right time that you can deploy that thing into a real environment? We deal with the same thing in software. You can do all kinds of testing, but until you actually put your system out on the internet in a public-facing connection, you do not really know what it is going to look like.

Suzanne: True enough.

David: Testing is one of those activities that you always have to make that judgment call of, *When is it done?* I do not know that that is a problem that we have solved yet.

Suzanne: Right. Yes. My experience is that it is still…It is a research topic in and of itself, but it is also a, "and it depends" on context and how critical, how expensive, all those kinds of things. I do not think we are going to have a, *Never take it below this level for everything*. Because that is just not how a world works.

I want to switch a little bit to transition and learning. If I am somebody who is new to digital engineering, where do I learn about it? Some of the things you just talked about, we do not really have good standards for what is a digital twin and things like that. How do I learn about both the opportunities and some of the caution areas? What are the resources that you would recommend that people take a look at?

David: Wow that is a great question, and one that I am probably not the best person to answer it. I simply started using my search engine and going through the various tooling and framework…

Suzanne: OK, so, don't be modest, David, because one of the resources that is now available is your blog post.

David: Oh, true enough. Yes, so as an introductory to what is digital engineering, that is out there. There are a lot of software projects out there on GitHub for example that have different efforts going on to provide tooling for the digital-engineering space. Some of our collaborators in places like SERC [Systems Engineering Research Center], they have done a lot of work in the space as well.

Suzanne: So start with the blog post and go from there, but this is a space as you are saying, there is a lot written about, but I think the caveat is that it is still evolving so make sure that you look at multiple sources and make sure you understand the viewpoint of your sources. The example I am going to use is, I was at a meeting where there was a vendor who was giving a presentation about the digital twin that they had built for this one type of component. When you started digging down just to the layer of where we have been talking, what are the file formats, and how do I exchange this with other things, they just went deer-in-the-headlights. Their concept of digital twin was, *You take our model and you use it in our system*, and there was no interface with the rest of the world. I think that is probably one of the things that you want to look at when you are looking at resources is, are they closed or are they open?

David: Yes, it is easier to build self-contained vertical solutions than it is general solutions that are going to work across an ecosystem of products.

Suzanne: It is not even so much a general solution but put the hooks in your vertical solution, so it can be interfaced with in other areas. That was the thing that was completely missing from this solution.

David: Yes, there are actually tools and frameworks out there that are going in that direction of trying to provide a general set of interfaces that hook into a lot of different products that are out there. That is a really hard problem. That is a complexity problem when you start getting into it. Every new solution has got to come with a new set of integrations to hook it into your authoritative source of truth and make the tools, like you were saying, integrate in meaningful ways to provide functionality.

Suzanne: That is something that you mentioned I want to sort of reiterate, is that this is something that we are not done with this, both in terms of research and adoption. So everybody that…Dr. Roper says, *Go all in*. But when you go all in, be ready for some of the things that are not done yet. You are going to be an explorer in the space. It is not something where you just can pull digital-engineering tools off the shelf, and it is really already done for you.

David: Yes, once you start coming up with the use cases for how you would want the various toolings to integrate together, I think back to your research question, I think there is a big opportunity for some human-centered design research there, to figure out what some of those use cases might look like when you start putting them together in tool format.

Suzanne: OK. What is next for you? We have all these research opportunities; what are the areas you are interested in? What am I going to be talking with you about in six or nine months because you have gone off and figured one of these areas out for us.

David: Hopefully by then the effort that I am working on, we will have delivered our report to Congress on the state of digital engineering across the department. It is really interesting to see what a lot of the different programs are doing. Everybody—well, not everybody—but a lot of programs across the department are doing digital engineering in various forms, and we have been trying to capture that and really examine the best pieces of what everybody is doing to put that together into something that the department can use.

Suzanne: We will have to put that on our calendar because I definitely want to have that conversation with you. Maybe we will even do a panel with some of the other folks that are part of that. That would be fun.

David: Yes.

Suzanne: Good. This is good news for people that are trying to figure out whether or not their program should be, how they should be, that this report is coming, and that is probably within the next six months that will be done?

David: Yes.

Suzanne: OK. So, that is a good time frame. I want to thank you for talking to us about this today. I am working on large programs, cyber-physical system programs where this is a big rock, and so the research that is going on here I think is really important to everybody from the systems-engineering through the software and test communities to really understand what can we get from digital engineering? What are we getting from it? and what can we hope to get from it as we move forward in the future?

Thank you for the work that you are doing here, Dave, and I really look forward to that state-of-digital-engineering report coming out. I enjoyed your blog post. I want to tell our viewers that if you want to enjoy Dave's blog post, you just really want to go to insights.sei.cmu.edu. Easiest way to find blog posts is to look for the last name. In this case it is Shepard. So find Dave Shepard, and you will find his blog post on digital engineering.

This podcast will be available in lots of different places and wherever you get your podcasts, you will be able to find it. Our transcript, as always, will have links in it directly to the blog post. As other resources become available, they will get added into that to things like the state-of-digital-engineering report, so you will be able to see those on the transcript as well. If you have any questions, please feel free to send an email to info@sei.cmu.edu, and we will do our best to answer those questions for you. I want to thank Dave again, and I want to thank all of our viewers for participating in our podcast.

David: Thanks Suzie.