Shane McGraw: Hello and welcome to today's SEI webcast, our software development open forum, Ask Hasan Anything. My name is Shane McGraw, outreach team lead here at the Software Engineering Institute and I'd like to thank you for attending. We want to make our Q&A today as interactive as possible, so you can submit your questions in the YouTube chat area now. If you're on LinkedIn or Twitter, you can use the hashtag Ask Hasan SEI. Once again, that's Ask Hasan SEI and we will get to as many as we can.

As I mentioned, our subject matter expert today is Mr. Hasan Yasar and Hasan is the technical director of the continuous deployment capability group here at the SEI, where he leads an engineering group developing prototype solutions with DevSecOps. He specializes in secure software solutions, design and deployment in the cybersecurity domain, including digital investigation, incident management, large scale malware analysis, and Hasan is also adjunct faculty at Carnegie Mellon University.

That's his formal bottle, but Hasan is one of the great resources for our nation, all things DevOps and software development in general, and he has a great passion for this, so I want to welcome Hasan. Good afternoon.

Hasan Yasar: Good afternoon, Shane. Thank you so much for having me. This is one of the passion topics I have been really eager to research and discuss and share the knowledge with the community, so we can all get better in building reliable and quality software at that. Thank you so much, Shane. It's good to see you as well.

Shane McGraw: All right, enough for the small talk. You're now in the hot seat so we're going to start firing our questions to you.

Hasan Yasar: Ready.

Shane McGraw: We've got a number even prior to the webcast, and then we will get to some that are coming in live as well. So let's get to question one for Hasan. From Laura asking, SBOM or software bill of materials is a hot topic. From what I know, SBOMs are a way to quickly act and determine the impact of an upstream compromise. Will they detect a build compromise?

Hasan Yasar: Yes. It's certainly a growing topic right now, Shane, for the SBOM, especially for dependencies or open source, whatever you call about it. I think let's define what the software bill of materials means. It's basically, all the software components, that is including open source, including any type of dependency in the application that you're using it, and make them as visible, make them as traceable throughout the life cycle. When we say traceable, it's basically, we have to know what other components are you using it? What diversions are you using it? Where are we using it-- that's actually a couple of things, like what diversion that we have it, and where we are using that libraries in which context? Like, it may be a server, it may be some of the (inaudible), that some of them is a deal package, some of them's runtime package.

If we know that where it is we are using it so we can be ready for any type of incident, any type of vulnerability, it's becoming for a specific (inaudible) for that libraries. Because if you don't know where it is, if something happens, if something happens for a specific libraries, if we don't know what we have in our infrastructure or the build process, or (inaudible) it's very difficult to find by them, actually. Because we are racing against the time. And like, there's a concept of the zero day, even though the vulnerabilities exist, like identify (inaudible) in the community. As an organization, if you don't patch that vulnerability timely that did an application, it is going to be a zero day still for us.

So we have to be very effective to that. How can we do this effectiveness? It's really a good software delivery and (inaudible) such as in a good DevSecOps end to end, will help us identify those vulnerabilities, identify this type of virus easily. Or we can keep track of libraries, track life cycle, we can go and patch that. I know we're going to discuss most likely on that-- I'm guessing it, because this is going to continue thing, we're going to see how we can build up SBOM as part of (inaudible) integration. How can we build up the SBOM using a single source of (inaudible) example? How can we connect the dots, so once we go to the further dive into the question, I will come back again and answer that specifically. Overall, using a good practices and enter in life cycle with DevOps component will definite, definite help us and speed up and be aware for any type of vulnerable, any type of the (inaudible) life cycle.

Shane McGraw: Great. Question two from Raoul. He wanted to know, what are the core operations of DevOps? Can you define a typical DevOps workflow?

Hasan Yasar: Oh, great question again. I'm glad that that question came at the beginning of this topic, because we're going to build up the rest of the (inaudible) most likely going back direction. So I'm going to go back to the whiteboard and draw something in the whiteboard, so you can really discuss more interactively, see what is the common components are.

So when I really give the what really DevOps means, there is four fundamental principles of DevOps. And one is the collaborations that we do. And the second one is the infrastructure, as code. We kind of call as IAC. And third one is automation. And fourth one is about the monitoring. So this is kind of like more about the common principles that we were talking, other there are dependencies. So what really means this core component for us? I know you people heard about the culture as the one of the elements. I kind of give the knowledge of building up a DevOps is, think about that you are really driving a car on a highway. Like, you just got to go fast as much as you can in a highway, but you are looking for a good highway, right? If you don't have a great highway, no matter what type of car that you have, no matter what type of brand that you have, you cannot go as fast as you can.

So let's think about this analogy in here and take a look at perspective. We need a highway, we said. We need a good road. In this road, we can drive cars fast. So what means in the software role? We will laterally build up the code. We deliver as fast as we can. And end of the day, we can safely arrive our production and why, which is our destination, right? We will like to go into production at our destination.

To achieve this road is really requiring the build up a great infrastructures. It is important build up a right infrastructure, which is give us ability to (inaudible) and give us a repeat the same environment over and over and again. It's going to give high degree of assurances that we don't have any intruders that go into the road, so kind of block that, which goes back to as (inaudible) we discussed. So we can really block anybody goes to the (inaudible) which is improving the highway.

And how are we going to build up the highway? We have to understand our team, which goes back to the collaborations. Like, teams are-- people are going to use this and where people as us, like the users, we would like to use this highway. We would like to use the application that we are building. We have (inaudible) and we collaborate with the team to build this infrastructure. At the same time, we can use effectively with this infrastructure, requiring the stakeholders. So collaboration's all about the stakeholders and the stakeholder has a visibility or has to say something in the software life cycle that we are looking for.

So let's look at the other two elements with automation and the monitoring perspective. Let's build up the highway. Teams are agreed on it, and we have a drivers, we have users, we have developers, we have architect. Everybody's kind of a part of our discussion, our team, even though we had a cops on the (inaudible) security people usually, right, perhaps are looking for anybody goes high speed or anything else, we can trap it. Same analogy in the software world. We can have a cops. Cops like to helping us to go as safely in our destination. So we get concept. Now we would like automate, and what we are automating is, we don't want to have-- if you're going too fast on a highway, like, you know, you have a great car, you know, 75 mile per hour, whatever the speed limit is, because we don't want to go above the speed limit of course. So within that condition, we would like to go as fast as we can. How can we go as fast

as we can means, we have to automate that some of the steps. What automations are? Let's say we go through the payment system, like, you know, we have to go through the ticketing system, probably, in your highway. Or you may have the checkpoints in the highway. Same thing for the software road is true. We will like to go fast by building the components, by compiling, by testing, by delivering, by deploying. It's requiring a automation. So we build that IAC that's got a little bit layer. So we have the infrastructure as the first layer we build it. So now we put the code and application on top of here. To make sure that it is going fast, we have to use automation in our deployment. So like, we can test it, we can build it. So we have this combination. We automate the build and deployment. Now next one is, how are we doing well? Like, how are we targeting a time closely into our production or new item that we have? Now production or new item or we may have other testing, or we have a staging a new item that that we did, so go in this things like, how we are doing well, in terms of our driving. We can improve our practices, we can improve our knowledge. We can improve and some-- a performance of the system, it is requiring a monitoring. So we have to really monitor our progress, how we are doing well through the life cycle. We can get better. And also, monitoring is also loaded term. We would like to monitor how application is behaving well, and also we can collect the feedback. So feedback is from stakeholders, from our users. We can collect the feedback. We can monitor those, get into our team that we were talking about. Let's go to (inaudible). So teams, it's going to be our collaborations, people will not be share that.

So like, fundamentally, DevOps is kind of building up a good way of building the software, that we have the speed up. When we go fast, we have to use automation practices to go fast, because otherwise, we can not go fast. But to (inaudible) some of the component that we are building, there is another dimension in the DevOps which is more architectural element. But if you cannot build architect your system well, we cannot automate some of the practices. If you don't have-- you have a great road, you have a great turnpike, you have a great, great component you build in your pipeline as your road, if you don't have a good car that you cannot go as high speed, which is the similar analogy. Like, you need a good car, you need a good architect. You need a good component just so we can drive fast. I hope that answered the question, Shane. It was more of a fundamental--

Shane McGraw: No, that was terrific, the drawing definitely helped.

Hasan Yasar: Fundamental topic.

Shane McGraw: No, that was great. Like I said, the drawing definitely helped as well, so let's keep that going too, Hasan. We're getting great chat interaction. People letting us know where they're from. We've got a truly worldwide audience, which is great to see. So I'm going to work in a chat question here from L. Grant, asking, I don't know where to learn to go from being a developer-- I'm sorry-- from being a programmer to being a developer, Hasan. Any questions-- or any suggestions, how they can go from a programmer to a developer.

Hasan Yasar: So let's take a look. Again, this is more about the work post related question we are talking about. So for us, like a programmer, it seems very little bit limited to the certain language we are talking about. Let's say, in our project, you know, our project probably might be using some specific language, maybe using a C++, you're using maybe a Node.js or some other things, kind of like you are programming. But when you talk about the developers, a developer has more a content in it, like the developer's not write the code, but we sometimes did the designs (inaudible). Sometimes you will like architect our system to address some potential problem that user may have it. So developers, it is more, I think, more about the software engineering practices, we say, and that has the good understanding about how can we do a database design. What is relation of database I can go? What is the typical algorithms that we can use in our context that we are building? So where we can start, you know, typically, it's about the passion. When I give like either programmer, or the, you know what, the developer, you're thinking, you should have a passion. So what really means like, I have been programmer almost 25 years. If I did a great code that I'm writing, my one should be clear. I should have really put myself into the situation. I should put myself in a person's role, in a user's role, I can do the better in writing the code. So it's kind of like I describe as the developer. It's basically, like writing a poem, or drawing a picture as an artist. So your mind should be clear. Your mind should be in the work, the really the good job, because or you're understanding the needs. Because we are using our head. I mean, whatever we learn, whatever we have understanding for the system, that system we are building, it is requiring to build up our knowledge, build up understanding, build up a domain which can be like one of the users. We can be like one of the end users. We can be like one of architect in example. So where we can start again, really you

should have a passion. If you're looking for just the writing the code, maybe you can be the good programmer, you can write the code. If you're looking for a good developers, it's-- you should have a passion about understanding the context, understanding the system, understanding algorithms in development. And where you can start, there are different curriculums, different universities are offering. And like, as far as I know, there's a computer scientist almost everywhere in US, most of the universities has. And some universities are also teaching software engineering practices as well. And if you're going to be just the coder, there are a lot of open source or there are a lot of learning platforms available, just teach the code. Like, you can learn C++. Another things I would like maybe here, I can be a good developers. If I'm going to switch to different language, I can go take a look at what the language means for me, and I can learn that language, then I can still write the code. It's like, being a developers, understanding the context, understanding the system, understanding how the memory are lay out, how the communication, how the interactions are from the system to the other components will help you to use the different languages. Like, I can different-- like, I give the analogy of, let's say I'm speaking in English right now, right? Kind of (inaudible). If I'm going to speak different language, the object is not really changing. Object is the same. Like, chair is a chair. But if the verbiage is changing for a different terminology. The programming, it's going to give you an ability change the different term. You may have a full (inaudible) in C++, you may (inaudible) in Pascal or other languages. It may be different interpretation, but the context is the same. Context is the same of writing algorithms and component. So again, going back to the how can they learn? And if you're going to be developer, you should get trained from the college, or the engineering perspective. You look for programmer, just to learn the language as well, from any open source environment.

Shane McGraw: Right, next from Steve asking, how can you assure the security of the software development pipeline? If a pipeline is hacked, the software could be compromised and internal testing may not show that.

Hasan Yasar: Yes, so I know we are paying that a lot of attention right now, with just a lot of the happening through especially for the SolarWind. Couple other example I brought that screen so we can discuss more in the pipeline perspective. So it's a great things. We typically, as the community, we focus on building a good quality, secure application. We often forget how to build it. How to build is the key point, that where we are building, which is the software

pipeline. So if you look at this picture, this is the section and we typically use right here, this is our pipeline, which has left a little bit bigger, which has the contuse integration, contuse delivery, other practices that we have. This is our pipeline. But if we don't secure this pipeline, what that mean securing pipeline? Let's say I a developers. I'm right here as a developers, and I'm going to use, like I'm on a development team, I will might be use some of the libraries and pull to achieve my code. Because, you know, typical a current developers, and as an engineers, we are depend on so much-- depend on so much complexity that we have in our environment. Like, if I'm going to write the code, I just go to Google it and go to get-- have to get some things available for me. I don't want to reinvent the wheel, because we are racing with the time. So with this rationale behind it, I can get the libraries, look into the my pipeline that's going to be part of my (inaudible) integration example.

So suddenly, I'm inserting x library here, into the my build process, which is how we're getting the open source, which is SBOM component. Second component is, how am I going to be securing as the configuring of this pipeline? Let's say we have a continuous integration that we have. We have a continuous deploy. In the continuous integration, we might be using some build server, like maybe Jenkins, just to make it more realistic example. In the Jenkins-- and I'm giving another one example which happened recently by in November, we're using a some static analysis tools that we may have for (inaudible). So if you don't configure this two tool properly, now we are opening up a vulnerability for the people can attack it. So now it looks like we are in the parameters of a network infrastructure that we have here protecting, but if you don't configure each of the tools that is part of your pipeline, it is opening up a tech vector people can take an advantage of using the one of the Jenkins configurations back to you. Even though almost every tool vendor are describing it as the best practices to set up your Jenkins, set up your git lab, set up your (inaudible), set up your (inaudible), it's exist already. As a practitioner, we just go and download and grab it by default and then try install it environment, without even thinking, all right, how can I make sure that I have a good and secure implementation of this component? How are we going to make sure that Jenkins able to segregate multiple builds in the build staff's build perspective? So we have to step back, think about that. How we are thinking about application security? Have to think about as the pipeline itself, and how can I secure the pipeline? How can I secure this? How can I configure it properly? And then, how can I be ready for any type of outage the pipeline?

One specific example I can give, which actually happened a couple of years ago, one of the smart, I'm going to say, hacker people, they used the one of the build agent and then put a data mining vulnerability into the build agent as part of the build server, did data mining through the pipeline. It's a perfect, great idea, because build is running and then you don't really pay attention how the build is acting, so you're giving a free resources for adversaries, and taking advantage of build agent, and running a data mining on the build agent, in some other process as part of it. How it is get there, which is about the lack of configuration, lack of not monitoring libraries. And then we don't have a monitoring that we discuss at the first-- second question, I believe. If we don't have a good monitoring, how are we doing well of the system's behaving? There is no way we can capture any type of abnormal activities as part of that scenarios. So we have to configure it as a summary. We have to configure our pipeline, make sure that we got to following best practices. We have to be ready by monitoring any abnormal behavior of the test or the build stages. And third one, we have to have a full traceabilities, who is accessing our libraries, our pipeline? What they are changing? Like, we should establish the good as secure a commit process into the (inaudible). We should have a kind of SSH or we should have a token rights. We should have a (inaudible) deployment. We're going to make sure that they are accessing the pipeline as an individual, we are following the best practices as well, Shane.

Shane McGraw: Great examples, Hasan. Thank you. Let's go to Matthias, asking, large scale agile development often involves huge obstacles to consistently applying security policies across environments with multiple contractors, across government and network boundaries. Any tips on how to deal with this, without compromising agility by crippling red tape?

Hasan Yasar: Oh, it's another loaded question. I think it ís going to probably an hour to discuss it. Just to make the great, great ideas for that perspective. First of all, when you-- let's step back little bit. So when you look at the scale agile framework, or any type of scale agile one. So the bottom line, all the work, like we have the multiple systems we might be using it. We have system go on and two and three, whatever the system we have. So we have multiple stream it's going on. But if you a little bit go in the level, we are in the developer level right here. So I'm going to pause it here. Let's open up a little bit more. So we got the system that we are building. We have different actual components. But end of the day, an agile practices says, you have to really be ready. Like, I don't want to go there for manifestoes or to have poor (inaudible) principles. Overall, we should be ready for to accept any changes what agile

mindset seeks, right, which is kind of not compromised. But what we should do for the (inaudible) perspective, when we are writing any stories that developer going to work on their task, like a writing stories. Each of the stories should have a security stories as well. So what the security stories mean for us? When I'm writing any stories, like maybe, you know, it can be a high level, it can be epic, or it can be in a something like a high level of the whatever the project. But when we look at this epics and stories, we can write a security stories as part of our requirements. So now suddenly, we are bringing our security thinking into the level of developer by using the various agile techniques. To look at the scale agile framework by the line, we are using a scrum as the bottom line work. We do the scrum, scrum work, we do the bottom at developer level. Maybe using a combine as one of your practices. So these are the typical agile practices, and then it goes back to the team level. Teams are using the scrum, teams are using the combine. And team of team is basic (inaudible) and make it more a scale and each of the system that you are working on.

So let's drill down in the team level. We start from the security that part of our code that we are writing. We can bubble up and then during I'll put security into the system level we're writing. So there is a relationship between these two. One is, if I know my system's correct logistics, if I know my domain, and I should really think about, what are the applicable security stories that I have to write in a develop level? There is usually, there's a disconnect between what the domain requirements are, security perspective. How can I make sure it can be applicable to my code I'm writing in a develop level, which is the stories level. So if you have a good bridge and connection between the higher level, in a domain specific system space, we go into the level of future that we are writing, it's going to be better chance to get security as in the scale. But we often, what we do, we not doing this type of work at the beginning. We always relying as late in the game, like just using (inaudible) as an example. It's sometimes too late, or we're just relying sometime (inaudible) maybe sometimes we're ignoring. We say, okay, organization, I will continue. If something happens, I'm going to go back and take a look again, because you know, I'm just eating up the chance rate now. So we never able to get the more organizational risk, more organizational domain for a security specific. So instead of going kind of like a specific, they're not do it, let's make it applicable security for our domain specific, so we can scale up. And none of-- almost every adjunct framework, it's basically not against it. It is basically helping, but we have to change our mindset. How can I add security into the our agile thinking? How I add a threat modeling concept into the our agile think? It's just a way of thinking here, Shane. Like, how we do the function requirements? I'm just going to go back one more time and just explain a little bit about the software developer perspective that we

discussed. If you look at any software that we are building, and that is definitely a quality attributes that we are building. So developer knows what the quality attributes are. One of the quality attributes is security, resiliency, reliabilities. So we have to really think about all the quality attributes, how we are writing the functionalities. So business say still got that function, but as the engineer, as a developer, you have also think about what are the quality attribute with respect to (inaudible). So it's not-- nothing is bl-- it's about the way of thinking. So I don't see any methodologies blocking us. Its way of thinking is blocking us. We are not able to prioritize the any type of work that we do. It's not in our list. See, if it is not in our list as a developer, we're just always catching up next one. We don't need our time technically, either.

Shane McGraw: All right, so we're going to move on and we're going to combine two questions here, Hasan, as I think they're somewhat related. Kim had previously asked, what are some differences between continuous integration, continuous delivery and continuous deployment? And then Kartik added to the chat, what does it take to achieve continuous integration and continuous deployment? So can you tackle those?

Hasan Yasar: Yes, so I (inaudible) that's a good one, because I know it's going to come up in the continuous integration. So we have been in the integration of maybe years, like even though when I wrote a program from 30 years ago, I was in integration. But what the continuous integration means, that's a little bit, going to come back this graph shortly. So we have a various team members, like the team A, team B, team C are building the code. Everything their environment. Eventually, we have to merge this code, and then we have to build that. What build means, like we are just in getting a code and then creating a binary part of it. So this binary write affects-- it's going to be used in the production or the testing environment, right? So this is kind of like a build process. We have to understand what the build means. Beta's going to get the code and then package it, and the person, the build steps and create some binary artefacts, so we can use for various things. So now in the concept of the DevOps or the (inaudible) thinking, we're adding a continuous term here. We're adding a C term. Like this is kind of like a integration level. Now suddenly, we are writing a continuous integration. What a continuous integration means? So we're going to get a code. You can get the code from repositories. Remember, we mentioned about infrastructure at the beginning? So we will get the code plus other artefacts. Other artefacts is the infrastructure code. Other artefacts, other

config in system. Now build that. Now we are building, so created a artefacts, creating a (inaudible) testing and then you actually going to do the deployment shortly.

So continuous integration will help us to gather any changes from repositories, we get from repositories, the kind of change it happen in (inaudible) and development pushed it. A team pushed something and B team, for the (inaudible) for the person-wise or a team. So we can grab the code from repository and package it and then make sure delivering to the staging testing environment. So in the build process, we're not again just getting a code. We are getting other artefacts. Because the software that we are building right now, there are so much dependencies in the software. That goes back to our SBOM question at the beginning. So we get the code. We get other artefacts, how we are creating a deployment, is an example. How we are creating a dependent libraries as the another package. And so we get all together. Maybe we building the code, we just going to push into production. We didn't know what operating system requirements are behind us. And third one is, this is the first time probably you may be hearing that, write the code. We have the infras-- now we're having a documentation end. So don't think about that we are creating a word documents in the context. This is more about a specific information about our system. Our system that we are building may be some specific user guide that we are creating. Maybe we have us some notes that it is relevant to the how we are configuring an environment. Maybe some notes that specific with the how we are writing the requirements for the code that we are build. So suddenly, build is combination of the code, artefacts, other documentation, we can build it. Because as a result of the build, we are creating a artefacts. So artefacts is a binary. But sometimes, binary is not enough to run with the production environment, because you need other, get other information for a security perspective, for a maintenance perspective.

I saw so many examples in the bad practices in software engineering thinking. We are running in the build in the production environment. We never able to go back and reproduce again because we don't have right build steps, right configuration, right documentation, right infrastructure, code pieces in place. So the continuous integration is basic combining all of them. So let's look at the continuous deliver and deployment. Let me get bigger here. So I'm just going to give a quick example about the-- just to set the stage SDLC. So we have the requirements that we (inaudible) and we have the design, and kind of like architect as well here. And we have a development and we have a testing, so we are in the deploy and course in the

production environment. So typically, continuous integration is happening somewhere here, as continuous integration. So we got the code, we are building. Now we make sure that we have the (inaudible). So continuous integration's going to get a build artefacts for us. What is next? So after build artefacts, there are two concept. One is a continuous delivery. Another one is deployment. It's often a confused term. So both of them intend really go into the production environment. So deployment get artefacts and do the some testing or requiring testing (inaudible) security, and automatically, push into the production environment. That's kind of deployment goes into the production which user can see as live, as (inaudible) as a result of the build. What the continuous delivery is, it is manual to production environment, but it is automated into the staging environment. So as an organization, we really want to build artefacts, we can go to a test harnesses. We can initiate a different test harnesses. We can create a more test cases, test environment. We can deliver the artefacts into the staging or test harnesses, so we can do more testing for the artefact that we build. When we are done, we can manually put this artefacts into the production environment that's a continuous delivery, and deployment goes automatically. So next question we may think about that, can everybody look into this deployment? It depends on the business states. It depends on the system we are building. Can everybody continuous delivery? Yes, we should be continuous delivery because that's the intent of the building the software, as we can test more and more and more. When you're ready, you can manually get artefacts if you want to put into your production and why.

So quick summary, continuous integration will help us to build artefacts from code, infrastructure, like (inaudible) only for mission. We can package it, version it, then we can test it, the content, in the staging environment, make sure everything is working fine, which is delivery. And when we are moving into production as the users that can see automatically, that's the continuous deployment. So deployment, really it's specific to the business. Deployment is specific to organization, it's as an example.

Shane McGraw: Great answer, Hasan. Thank you for that one. Next we have, we're hearing about the software factory concept from our government customers and seeing it in RFPs. Can you explain what is a software factory?

Hasan Yasar: Yes, thank you, Shane. That's another kind of old term and let's make sure that it's the same concept here as the drawing. I think I can quickly make sure it's ready, so we can reuse again. So the software factory is not a new term. That came about 1971-ish, like when we are building up our softwares, when we are writing a code. So software factory's going to help us, or as a concept, as an umbrella, and we may use the multiple pipelines. Right here, we have a multiple pipelines that we can edit, so we can produce a software, or we can produce this system. So let's (inaudible) open up here, just going to come back shortly. So you may have software factory here. So you may have the various software delivery environment, you may call this DevSecOps. And why, let's make it little bit shorter so we can make the graph look better. You may have a DevSecOps environment here. We may have another DevSecOps environment here, another here. Because each of this pipeline that you are building has a specific requirements for the system, maybe the components that you are building. Maybe you are building some of API here. You may have some other (inaudible) system probably you're building. So the combination of the multiple pipelines is constructed and building a software factories.

So software factory will help us use the same practices-- this is important-- we're using the same practices, same process, like process our-- we have the check in, check out, we have the connected, we have to create a issue track and we have to create a chains, we have to create a commit message. These are the more about practices and processes. But however, we may be using a different tools to establish that, because DevSecOps pipeline depends-- sometimes depends on technology we are using. Like, one example, C++ build process is different than packaging a (inaudible). Even a different type of build agents to build up the C++ code versus a packaging about the (inaudible) libraries, differences.

So still we are building a software, but because of the component has a different needs, we may have a different pipeline. So software factory is consist of multiple pipelines, but enabling using a same practice, a same process. As an organization, we can do. One quick example, going back to the first question again, because it's hurting us. It's hurting us specifically. If I don't have a good practice of getting the libraries, it is more organizational policies. More organization practices. As an organization, we should help or guide the developer pull the libraries properly from open source, or manage the (inaudible) an organization, we can use it over and over again in the pipeline, which is a practice. Or, if I'm writing in any type of stories, it

doesn't matter what type of (inaudible) you might be using it, think about the security knowledge at the beginning, in an organizational policies, organization practices. So software factory is the combination of multiple pipelines and letting us use the same process and practices, we can deliver an applications, or we can deliver apps as a result of this combinations.

So single DevSecOps pipeline is not one quick example here, before going to the next question, like AEES or Amazon. They have hundreds of pipeline, hundreds of pipeline. They design the pipeline for a specific needs. Maybe they are creating a (inaudible) of the micro servers that they might be creating the pipeline. So technically, the Google or Amazon is a typical software factory we are talking. Building a software. But (inaudible) if you look at that, Google has a good software engine practice it's everybody's following. It is not a Wild West, everybody can do different things, no. Use the same practices, process and practices, we can deliver. There actually a definition of DevOps is also a set of principles and practices that enable Dev and Ops, including other stakeholders. This is a quick definition. It's going to tell us again, set of practices and processes, such as continuous integration, such as continuous delivery, such as continuous feedback. (Inaudible) are the common principles.

Shane McGraw: John sent in, how well are high schools, colleges and universities preparing software developers to take advantage of newer technologies and trends, such as the software factory principles you just talked about, model based software engineering and low code/no code platforms?

Hasan Yasar: Well, that's a good one. Actually, I'm going to give example from my son. So he just-- this year, he was in the ninth grade, actually, tenth grade, and he's doing more about the pipeline learning. So high schools, it's a good opportunity to let the people explore what the programming that we discussed, what the context are. So high school learners, and I will advise that, enable the students to see how they are capable to use a new building of the code and have them write the code. Actually, and I get it now yeah, as well, like we are in a digital native right there, because most of the kids, most of the students are learning or exposing their digital environment, they number one when they go to the schools. Like, they can use the ecosystem such as mobile devices. They can use the tablet. They can use the laptop. They

can use everything as a computer. So you never-- the high school, they should really give an opportunity how this system is working behind the scene. How can I make sure that any of the students are learning basically collaborating with the Google Doc as an example? Maybe writing a basic script on Excel file as the macro. Maybe writing the basis script of getting their job done. So like, building the concept of how it is going to work, and I can really get the students to be ready of understanding the reason behind it, understand the cloud, understanding the operating systems, understanding the basic coding, and the basically, like you can use the Python code as my son used it this year, writing a Python code and start to build up the specific example, or the application for them, he can learn.

So when we go into the higher level and the college level, we also think about, how can I make sure that I'm building a software system eventually will be used in the bigger context? So in the college, we always teach about the algorithms, the databases and other component, which it is good to have it. We should have it. No question about it. We need to understand the fundamentals. We need to understand how things going to work out, what the best algorithms are, how can I really use the better memory? How can I communicate in a short messaging system? So we can learn at the college level, but the same time, we have to tell ourselves, what is going to use in the industry practice? We should connect ourselves more about experience, more about trying something. I always encourage the students, if you're in high school, try to build up something for yourself. Maybe you just can automate your documents sending to your teacher. Like, try to automate that if you can. You know, maybe you're writing a basic PowerPoint. Try to write the basic macro in your PowerPoint and try to automate some of the visualizations and graphics in your PowerPoint, which is doable. So think about, how can I work? It's going to tell you to start some knowledge of the scripting. So when you go to the high school, the college, now you're going to get, how can I build up something that I can use and use my GitHub environment, open up a space in the GitHub, try to expose a more realistic scenarios, more realistic example. So what we are learning we can apply in the real world scenarios. I see a disconnect thing, because even in my students, I saw sometimes, they know the fundamentals, but it's not as realistics in the real world scenarios. How can we manage that? How can make users? How can I manage some complexity? How can I manage, you know, basic writing code, versus writing a desktop or sharing somebody else? It is more about experience.

So I want then think about all the facts, think about, how can I use different context more realistic? When they go into the more realistic example, they will learn how to build up the software. They will learn how can I speed up the pipeline which goes back to software factories? They will learn, how can I reuse my code over and over again, because they're going to write the code, they use a similar code again, and they will think about, how can I make it little bit (inaudible) of my code, that I can reuse again in the futures? So this is basic go in that direction.

And then in low code/no code, like one of the-- the part in the questions, it is more about the reusable of existing application for over and over again, without writing so much code. And most of the-- even sometimes we use that, like we're not writing the code but using a shape as an example. You're using other ready solution as either drag and drop or connecting dots, without writing much code. We're just reconfiguring an application to suit our needs. That's going to help us, so most of the students are already using that, that directions.

Shane McGraw: Another great answer, Hasan. Thank you for that. We've got a good question in the chat from Rajid asking, how does container technology change DevSecOps?

Hasan Yasar: Well, right, that's another good question again. Love that discussion. So when we-- let's talk about what the container means, so we can maybe come back to DevSecOps little bit. So let's go this way, like DevSecOps is a set of practices and principles for us. So we are building continuous integration, continuous delivery, continuous feedback, all this continuous term in the DevSecOps practices. So what the container is, as a little bit background, container is giving a process and isolate it from operating system for a serving a specific need. So it's basically, a specific to the app content and isolated from others. I have an app one here. I may have an app two here. Are both (inaudible) only with each other's, I can do that. So going back to the continuous configuration, as we said, we will like automate a build as best as quick as possible. And also, we will like to deploy without changing much in the production environment. So automation and deployment are the key element is helping us make the container rights applications. If my application is more container, more isolate of others in a small size of the

(inaudible) or the specific futures, then I can automate the build and I can deploy without affecting others.

So in this context, it's really helping to speed up delivery, speed up testing, speed up automation, because I can stand up at different test harnesses for this container, separate, this container, separate. I can run 12, 10 minute, same environment. None of them is going to be interacting each other as they're using different-- maybe using a different libraries, different versions, same libraries. It's going to help us to be quite quick. So how can we use this container? I think we should discuss a little bit more as well. It's about architecture element, as we said at the beginning. So you may be creating a once service as an application one. It may be a one of the micro service that you may be writing here. So because you are getting more specific to the service that how application one or app one is communicating with others. Same thing for A2 and communicating others. So you may be writing a micro services. So micro services is the way of architecting. You're packaging your services in the container, so you can push into the your pipeline. So like, pipeline, as we established, now we are adding an architecture, components of the system that we are building, and packaging as a container, and calling as a services. That's the relationship. I can deploy it, maybe one of the messaging services that I may have, maybe some of them are payment related, some of them are report and some of them have-- other may be sending some message to other components. Some of them may just a database communications. So it depends on the service type and we are packaging the service and putting the container and then chip into the production or delivering to the staging environment.

So it is helping the DevSecOps go much faster, whereby using continuous integration, continuous (inaudible) practices. If we don't have a container, yes, we can do the same DevSecOps practices, but we have to write a test automation. We have to write deployment packages. Maybe have to write different build packages. Maybe have a multiple pipeline we set it up. So in that connectivity it is helping to go much faster in the context. But it is important to make sure that we are writing the right container files to meet our needs. But we don't want to creating a container file is about the one gigabyte size. It's not going to help. So make sure that you're following the best practices, how to make sure that we have a right layer architecture in the containers. How can we write the very minimal size of the containers which can go fast in the DevSecOps deployment and test cases? So we can enable the reusable of the components

in the libraries that we have the layers of the container, we can use that. It is really important to make sure that we have a right container architecture that we are following, and we can use over and over again. Because otherwise, everything, put everything in the one container, make it a five gigabyte size, it is not going to serve the needs of the DevOps perspective. It will delay deploy, it's going to be problematic to deploy into the production environment.

Shane McGraw: All right, switching gears. Kim sent in, can you explain the DoD enterprise DevSecOps initiative. Or DSOP?

Hasan Yasar: Right, so I'm glad that that's (inaudible). This is a great (inaudible) part of the DevSecOps and working group as well. So in the DoD, we see many different type implementation in terms of DevSecOps practices, and in terms of the different understanding. As this subgroup is really helping the common understanding of what the DevSecOps culture should be in the DoD context, like maybe describing the playbooks, describing the framework, describing the whatever there is, describing acquisition perspective and it is giving a guidance to the rest of the DoD follow it. Remember that the software factory concept we discuss, we are looking for the same practices, same process, at different implementation. This is exactly DoD DevSecOps working group we are like to do, which is beginning already, after (inaudible) we can post the URL into the YouTube channel and people can download the later version of DevSecOps framework and playbook, and some other guidance that come from the US NCS (inaudible) as well. It's going to help us understand what the commonalities of building up a continuous improvement. What the key capability this far? You're looking for what the DevSecOps pipeline means. What the continuous integration means. What the continuous authorization means. So it's going to describe the vocabularies and describe the common understanding, and also giving some reference architecture that (inaudible) like use the (inaudible) use that for architecture for you as an organization. You can follow that reference architecture, build up your own (inaudible) container based DevSecOps environment.

And then the other things also, it's important to mention here, Shane, and a little bit about the container as well. It's also offering a hardened containers, so rest of the DoD folks can use the hardened containers. What really means, and in DoD context, we always would like to have an

approve for the software. We don't have any approval, none of the software and systems will go into the production environment. It is requiring authority to operate, which is (inaudible) and then we are adding a continuous ATO. So continuous ATO, authorized to operate, is continuously requiring a constant evaluation, constant check of the libraries and dependencies application with respect to security. What the DoD DevSecOps initiatives is really building as hardened containers, the hardened containers has all of the approved dependencies libraries for the one of the modules that developer might be using it. That module can be a node.js, can be one of the pipeline component, can be a GitHub, can be a Jenkins, maybe data science is using as some Jupyter, some other tools, has approved already in the available public platform and then people in DoD, users, they can pull that libraries from the common portals and use in their application context, we can cut the reapprove process in our approval component. So we can use the approved containers to speed up our approve process, so we can go faster. So that's another things as the DoD DevSecOps initiatives groups is helping.

There are many other activities as well as part of this group and there is embedded software development group is happening right now. We haven't really publicized yet every output, (inaudible) is building up. How can I use the embedded software developed in DevSecOps complex? And there is another studies happening as part of DevSecOps group, it's about digital engineering. And there is another connected with digital engineering in DevSecOps. I think there is a (inaudible) we did a couple months ago, and there is a direct connected with this, how we setting up our digital engineering ecosystems for a software pipeline which is DevSecOps pipeline. We can go back in years and with engineering concept using a source of truth. Other component will be part of it. So it is giving a good guidance in DoD. At the same time, lot of industry folks also using that. How can I use a typical DevSecOps framework in a highly regulated environments, like the health industry or a finance industry? Or a good examples of consumers of the DoD build it a DevSecOps framework.

Shane McGraw: We're down to about seven minutes, Hasan, so we're going to just rapid fire as many through as we can here.

Hasan Yasar: Sure.

Shane McGraw: Some are too long, Hasan, we can pass, but we'll just try to get through as many as we can here. M. had asked, does the size of an organization have any effect on DevOps implementation? What he means, are big organizations better suited to implement DevOps or does size not matter?

Hasan Yasar: So I will answer in different way. Size matters, but actually, it depends on organization culture. Let's put it this way, if you are starting from green field application, probably much easier to implement a DevSecOps practices in organization, or the system that we are building. Because everything (inaudible) from scratch. If you are going into the brown field application, let's say you have a legacy system, you may have a component has been done already. To convert into the DevSecOps it will take time. So DevOps is applicable for everyone, first of all. How are we using the DevOps in our business context? It depends. It goes back with our continuous delivery, deployment discussion. As the good organization-- it doesn't matter which organization you are in. Make sure that-- maybe I should use the drawing. Probably much better quicker with explain that. We always, always, I always suggest that. No matter what organization you are in, keep your repository management in your organization. Single repository management. You may have different team, team one, team two, team three, team four. So everybody's going to use the same repositories as the source of truth as the repositories. Then you can build up a set of principles and practices in this thing can use. They can build up their own DevSecOps environment for their needs, for their application. They can build up. Suddenly, this organization is becoming a software factory. It's becoming they're setting up their environment. So it is good everyone, but implementation is really changing. And think about if you have a team, a bit with ten people, your communication will be only limit to the ten people. If you are thinking about that bigger scale and suddenly you need to communicate each of team members. Each of team members has to communicate each other, you now get collaborate everybody in organization, but you can build up a common understanding of each of this teams, how they will like to do DevOps implementation, let them share information between and go into the team level. Let them do what they will like to do for implementation of DevOps or DevSecOps they're implementing.

So organizations, if you have a good policies, they can use it. It's more about how I'm going to change myself an organization or a small team to do my job in a DevOps way? Like, it's more about, again, the prospect of the cultural element, collaboration pieces. Normally, where you are in, ask yourself, you're a developer, you're a project manager, maybe an architect, maybe a tester. How can I change my job to serve the DevOps needs? What the DevOps needs are, as we discussed at the beginning, we need a collaborate that. How can I collaborate with rest of team members? Think about what is next. Where I'm getting this information? How I'm going to share with others? If I'm not writing a code, what other team members that I need to expose my information? How can I share my lesson? See, it's kind of like a changing of how we are doing it, and then scaling up. So we can scale up by doing the same thing and scaling up other team members so we can build up organizational knowledge.

So differences will be in the brown field versus green field, but the size, it will take time if you're a bigger organization, take the DevOps implementation in bigger organization. But a small organization, probably much quicker, much faster, because small team members are part of it. We can focusing our business. We can focusing (inaudible) much quicker than others. I'm trying to give a short answer, but another long topic we can discuss.

Shane McGraw: That sounds good. William asked, can you use DevOps for machine learning? I've seen the term MLOps. Are there similarities? Can I use my DevOps pipeline for machine learning development?

Hasan Yasar: Yes, yes. Three times yes. Yes, we have to use it, because if you look at the typical deployment rate that is happening the AI engineering and development and deployment, there is a disconnect it's happening. How can I get the package off the model and put into the deployment pipeline? So just pause it here. See, if you look at DevOps specifically, DevOps is really helping to deploy any capabilities. What the ML component's bring up at data element, it's bring another and modules in the context. So applications that we are dealing DevOps environment, actually, it has the part of the model as part of application. You can package

application as the model behind the scene and you can deploy. So the concept of MLOps actually extending that model deployment, model creation, model curation, model deployment, into the pipeline, using DevOps pipeline basically. So basically, getting the model into the DevOps pipeline, using the continuous integration as example. Use the continuous deployment delivery practice example. Use the continuous feedback example, feeding back to the data scientists how the model is behaving in application context.

So why (inaudible) actually came from the Google couple years ago. And they realized that without using a best practices in terms of delivery and deployment, it is hard to make the reality of the model in real environment. And there are a lot of scenarios happen at the past. There's a Microsoft (inaudible) happen a couple years ago. If you don't have a good mechanisms of monitoring how the model is behaving in production environment, and you can act proactively see model's behaving, it's going to get done through the DevOps concept, as the monitoring (inaudible). So I will say, MLOps is adding a ML component, data component, into the old DevOps pipeline. Make sure DevOps pipeline is supporting not just the code, and also supporting other elements, such as data, such as feedback, such as model monitoring in your DevOps pipeline. So maybe say that that extended version of the DevOps thing will including the MLOps concept in it.

Shane McGraw: So Hasan, we've got about a minute left here, so we can limit this to a minute if possible, from Corky. The last one we're going to ask is, can you please describe the POD based delivery model and distinguish it from the factory model? Is that a quick answer?

Hasan Yasar: It's going to be take longer. It will take more than a couple minutes to answer that. I will respond that questions probably in email, or the chat.

Shane McGraw: Okay, that sounds terrific. It just gives us a reason to do more of these, because there's a lots of questions we didn't get to Hasan. But just a great session. Thank you so much for taking the time to do this. You are one of the reasons why the SEI's so valuable.

Hasan Yasar: Thank you so much, Shane. Happy to discuss with you. We can do a repeat this section and add more question. I know there should be more question we haven't touched up yet.

Shane McGraw: No doubt about it. So thank you again for sharing your expertise. Folks, we want to thank you all for attending today, for sending in such great questions. Like I said, just continue to send those questions in. We will try to set up another one of these sessions. Upon exiting, please hit the like button below and share the archive if you found value today. Also, you can subscribe to the SEI YouTube channel by clicking on the SEI seal in the lower right corner of the video window. Any questions from today's even, please send to info@sei.cmu.edu. Thanks everyone, have a great day.