DevOps in Highly Regulated Environments

Featuring Hasan Yasar and Jose Morales 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'm a principal researcher in the SEI's Agile in Government team. I'm here with Hasan Yasar, a technical manager in our CERT Division, and with Dr. Jose Morales, who is a senior researcher in our CERT Division. They're here to talk to us today about DevOps—development and operations together—in highly regulated environments, HREs for short.

Before we get into that topic, which you know I have a lot of interest in, Hasan and I have worked together quite a bit in this area. Tell us a little bit about how did you end up here at the SEI? What is it about the work here that keeps you here? What is it that draws you to this kind of work? Jose, why don't we start with you?

Jose Morales: When I finished my post-doc in Texas, I started looking for new positions and where to go and I liked the SEI opportunity because it tied in academia, government, and a little bit of industry all in this one unique blend so I could pursue my research, I can help government, and I can interact with industry all at the same time. It's very unique. That was one of the main reasons.

Suzanne: What is the focus of your work today? You do things in unwanted software, which is kind of an interesting way to talk about something we sometimes call "dead code" and things like that.

Jose: I focus in two main areas, malware, behavior-based malware detection, and DevOps is the other more recent one that I've added to it. I've been doing malware analysis since, I don't know, 2000, maybe even earlier than that. The first malware I ever came across was in a Mad Magazine.

Do you remember Mad Magazine? It was in there and that's how it started. I've been doing malware for a long time. I like looking at runtime behaviors. That's how you can stop zero days when they first hit, when they first come in.

The DevOps, I started about 2 1/2 years ago with Hasan, and it's very challenging because every group you walk into to try to help them set up a DevOps environment, is a whole different set of challenges, whole different set of requirements, so there's no real recipe, they just rinse and repeat.

Suzanne: There's no one size fits all.

Jose: No. Absolutely not.

Suzanne: Hasan, tell us how you got here.

Hasan Yasar: It's a long story. Before starting at SEI, I was doing a lot of software development activities in industry and I was even running start-up companies. I had so much experience in real-world problems. I ended up at SEI and as part of CMU where I started to use my industry experience and skills.

Suzanne: ...match that up.

Hasan: At the more research side of it and then also helping the DoD. There is a big disconnect between industry and researchers, and typically researchers are producing materials in a form that is not digestible to industry. Representing industry and coming from that angle, it helps to go back to the community folks, which is kind of my old friends let's say or industrial practitioners, and connect the dots between researchers and industrial practitioners. If we are writing a paper, if it is not used by developers or architectural engineers, it is useless.

Typically researchers are doing like 80 percent of the work done, but it is not ready to use. Producing something for the SEI is more useful to the DoD and industry, which makes it more beneficial in a more efficient way. It kind of strikes me here that since I started at SEI in 2010, we never really called it DevOps at the moment, but we have been doing DevOps work since 2010 when I started at SEI.

As Jose said, when we go to any DoD clients or any type of industrial practitioners as well—I have been running like All Day DevOps and DevSecOps and other conferences so far—everybody has their own way of doing DevOps. Everybody has their own way of doing the DevSecOps or TechSecDevOps or no matter what you call it. We have our own way of doing DevOps at SEI. I have been telling communities, it is all about the process. It is not the tools. It is not a fad. It is real. Is it there. People are using it. It is a reality. We have to take advantage of that movement and make it better software, including security and including other quality attributes as part of the DevOps movement.

Suzanne: The SEI does have a particular view of DevOps that we tend to call, are we calling it DevSec or SecDev?

Hasan: Honestly DevOps is the broad definition, which is getting all stakeholder involvement like security folks and acquisition people. It's not one Dev and Ops, it is whoever has a say about something in the lifecycle, they are part of DevOps, including program management, including testing; everybody has some, even the legal department is also part of that movement. Because what we are saying is that we should have continuous feedback from everybody who is responsible for application quality and for application and operations, and also responsible for maintaining liability or any type of requirements of that lifecycle. So everybody has something in it, and security is only part of it. But apparently we have to call it DevSecOps to get more attention from the security folks because we don't want to leave them outside that initiative. Typically security folks are taking themselves outside that movement, because it is really coming after the fact.

Suzanne: In the past it has been after the fact.

Hasan: We would like to bring them into that DevOps movement, so we call it DevSecOps. Maybe you have heard it as SecDevOps name, or Secure DevOps name. Last year, I was running a DevSecOps event as part of RSA, and we did a quick survey of the 900 participants. So we said, *What are you going to call it, SecDevOps or DevSecOps or DevOpsSec or SecDevSecOps*? There are many definitions. So to communicate the idea, we said, *Let's get security in between DevOps.* It doesn't mean it is in between, it's basically throughout the lifecycle. In SEI, we are saying it has to be beginning of the lifecycle, throughout the lifecycle even not really ending at the end, still we have to keep continuously monitoring the security activities throughout the production activity, so we have continuous feedback. We have a continuous feedback back to the operation, back to development team and acquisition team. So it's really we call it DevSecOps to get attention of it. Same as right now, DoD is also calling it DevSecOps. DevSecOps, it keeps growing as common name throughout the communities.

Suzanne: I think this is going to end up being Dev bracket Stakeholder X because that is what you're saying.

Hasan: Actually I put the parentheses in my linked title, it said Dev (Sec) Ops...

Suzanne: We are trying to integrate across a wide spectrum of stakeholders who traditionally have been a little bit siloed in the way that their work happens and in the way it is transferred from one stakeholder to another. This is really the mother of all integration environments.

Hasan: Actually, if we look at the root of the problem, and if we look at the waterfall paper for example, it was written in 1971, we made that waterfall paper as a very siloed process. We don't want the same thing for DevOps or DevSecOps. We would like to create a more continuous basis, more features, and more interaction with the rest of the other team members including security. In the end, every phase is required. If I am doing some sort of future development, I should get security requirements right today. If I have security requirements, I have to carry out these requirements as writing the code, making the right design, making right testing for the components...

Suzanne: ...and getting feedback.

Hasan: Right, it's a process. One of the reasons I have been working with Dr. Morales is getting his experience on the malware researcher and understanding the nature of the security problem, which he knows the bottom line, so why don't we get that operational knowledge put it in the application lifecycle? That is our goal of having interaction with him because we know what adversaries are thinking. We know the mindset behind it. If we know what adversaries are thinking, what their mindset of cracking the systems or developing malware, [we can] take advantage of those knowledge pieces make it more developer-centric, more architect-centric so eventually we will be winning because we can eliminate any type of vulnerability in our actual code we are developing.

Suzanne: That's actually what all of this is about, is shifting things enough to the left that we're making progress early, getting feedback early, so that we're not waiting until Jose comes in at the end and says, *Hey, did you know they really like to do this, and if we did that in the design, we could avoid that*. So we are trying to get that kind of feedback into the early part of the lifecycle and not wait until the end. That's kind of the message.

All right, DevOps, we've got a process, tooling. Part of the tooling emphasis on DevOps comes with the insight that when we are doing these iterative sort of incremental developments, we are touching the code a lot, and so we need a lot of automated testing, a lot of automated integration to make sure it works. That is sort of a baseline for it. Then the processes that make that work effectively, so that we have the human in the loop when we need it, etc.

When we pull all of that into a regulated setting, regulated settings include high security, high safety, high human criticality, high mission criticality. These regulated environments typically have things that people have to do to be able to be certified in some environment, and in these settings now, getting that feedback on a continuous basis becomes even more important. What are some of the challenges that you all have run into when you take already challenging integration of different stakeholders into an environment where we have people that are coming in and saying, *I need to certify that you are safe or airworthy or secure for this environment*? What are some of the additional challenges that brings to the DevOps problem?

Jose: From the DevOps side, the fact that it is a high secure area already implements the siloing isolation concept. You walk into these environments, and you find every group of devs, operators, QA, testers, whatever, they are all separated from one another. That is ingrained into their culture. Already walking into an environment that for security reasons is siloed, is isolated, already goes against one of the basic principles of DevOps, which is all stakeholders together. Besides that, there are several other things that come up. In a siloed environment, it is hard to have things like environment parity. It is hard to have a centralized repository of documents. It is hard to have a realistic staging environment that represents the production environment that whatever they are doing is going to eventually get deployed to. Those are some of the main ones that we come across.

Another issue that happens in the DoD world is retention of contractors. Sometimes a lot of work is done by contractors, and they are only there for a period of time, of which a subset is made to get them rolling into how we do things here. Then they leave, and you have to bring somebody else in. There is no set process on how to get a newcomer into the DevOps way of doing things that just flows with everything.

Suzanne: You have always got to worry about onboarding.

Jose: What we found is in spite of all of that, there are things that you can accomplish within these hardened environments.

Suzanne: What are some of those things that you have discovered are enablers to an operation that is much more DevOps centered?

Hasan: Before going to enablers, I have a couple of other things to add, Suzanne. Let's open up the HRE a little bit more. HRE is available in industry as well. If you go to the financial sector, they have similar type of constraints that we are dealing with in the DoD.

Suzanne: Electric power…

Hasan: Right, and same for safety-critical systems even the health systems. One of the DevOps thinking is the developer needs access to everything, but you cannot really get that thinking in the financial sectors. Sarbanes-Oxley [Act] says you cannot have the developer touching the production environment. So when we look at the community thinking, it is all about the concept of "you build it, you secure it, and you run it". It's not going to work with this highly regulated environment, which they have their own compliances. They have their own policies.

You cannot get the developers are building and touching the production environment simply because they don’t have a chance to go to the production environment. How can they run it? How can they touch it? We cannot really get what the people are thinking in industry. When you get into the environment, it is different. We have to realize that the problem is different.

When we look at enablers, there are many enablers we can take an advantage [of]. One of the things where we have an insight is, we are saying, *You don’t need to be like delivering application every minute because your context is different, but you can take advantage of DevOps environments. Think about the environment parity and thinking about the sharing and transparency. thinking and planning same goal for the projects,. These are enablers. So take the principles of DevOps, collaboration principles, make it automation, and make it environment parity. These are really helping for specific HRE perspective, because you don’t want to repeat the same thing again.*

Suzanne: That is one of the things that we’re finding in some of the work that I have seen in DevOps is this ability to actually do continuous/not continuous delivery. We are not there, but we are continuously providing feedback to different stakeholders, to the test people, to the security people, through what we’re making available to them. And they are giving their feedback to the developers much, much earlier than they ever have before. That seems like that’s one of the things that early feedback is one of the things that would, to me, drive motivation for these other stakeholders to actually be involved in DevOps. Are you seeing that?

Jose: That was one of the innovations that we had to put into place to be able to get end-user feedback as quickly as possible, as soon as possible along with CD and CI.

Suzanne: Continuous delivery, continuous integration.

Jose: You had to have within a siloed environment a staging environment that was as realistic as possible to the production environment that they may not necessarily have access to. This is true in primarily adversarial scenarios. If you have the staging environment—and it is as realistic to what you are actually going to see when you deploy, and it is all inside a silo area—you can do the entire DevOps process all the way through end-user feedback within that environment to the point where we can tell people, *Put operators or end users to interact with your staging environment if that’s required, and those end users should not be anyone else working on the project*. They are just people interacting, and you can get feedback that way. That is more realistic when you go out.

One thing we discovered was the end user a lot of times was considered to be the operator, like the person who is going to use what your...but in a lot of scenarios we found that the end user isn’t actually the operator. There are other people involved who are also considered end users. So inside the silo environment we are able to emulate those with human actors to be able to get real, complete end-user feedback. If you don't have that, you are getting just a partial.

Suzanne: So I am going to give an example that I know about. If we are talking about airworthiness, which is one of the regulated environments we have to deal with, and we are talking about pilots and we are talking about airplanes, it is very easy to think that the end user is the pilot. It's fun to talk to pilots, but what about the guy that's maintaining the plane? What about the people that actually have to do the training schedule? There are a whole lot of other end-user stakeholders that need to understand how the system works that need to provide feedback as it is evolving.

Jose: What we found was that was not the case in reality. The end user, in almost every case, was always identified as exclusively the operator. You were missing a much larger pool of end users.

Hasan: Like when you say *operators* in industry it's called more IT operational people. But in HRE it may be different because they are responsible for maintaining an environment, and the operators are the actual users, so there are multiple user sets. As Dr. Jose [Morales] described, if we have a production-like environment that is very close to the production, we can still do continuous delivery. We don't need to have a continuous deployment. Continuous deployment goes to more of a production environment. We are expecting to tell them, *Let's have a continuous delivery of every build you had made*. If that goes to the production-like staging environment, then the other users, including operators, will look at it and give the feedback. Then when it is ready, the package can be done. Either it can be a container or another type of packaging tools that can push into other environment(staging) : like production. Then it goes to production environment, then the operator knows how to use the systems because they already have experience. They know what is coming to their pipeline, so they are ready for operational purposes, for using, for training. It is very helpful to use that type of concept at the beginning.

Suzanne: One of the things that we run into in Agile, DevOps, the whole sort of space of trying to reduce the time from concept to capability, is that we do have stakeholders within the DoD that are independent of the development and acquisition and particular your DT, your development test, operational tests. What has been your experience in trying to help those people to come into a DevOps-pipeline kind of environment instead of staying completely outside of the development pipeline?

Jose: At least in my experience, we have worked primarily with developers, so we see it from their point of view. We take into consideration these outside stakeholders who are not directly involved but do need to be aware of what is happening. What we push for is overall collaboration through meetings, scrums, critical chains, emails, and any other form of communication. They can be updated daily, weekly, sometimes even hourly, but we haven't been able to get to that from their point of view. We are mostly dug in with the developers, and we work from there out.

Hasan: Some other engagement is we give them a pipeline, so we build up the DevOps pipeline. That DevOps pipeline will be transferring to the other stakeholders. Let's think about acquisition perspective. Instead of getting hard-core requirements, get the capabilities of the acquisition folks, which users are telling that, and get that capability converted into either epics or stories as part of the sprint. That is how you start getting that work into the pipeline.

Really the pipeline, the tool is ready. The tool is basically setting up all this integrated pipeline and compatible system maybe, maybe other type of Wiki pages as we start to build systems then let the other people. Then let other stakeholders use that environment with the right process. Remember you said about the process? You may have a great tooling organization, [but] if there is no right process, nobody can use that. As an example, some organizations, set up a great tooling. The have a great pipeline. They spend money, really tons of money. Everything is integrated, but the developers, they don't know how to use, or the acquisition people, program managers aren't able to take advantage of it. Which they should be able to go and pull up the report easily for the program or project management perspective they've been working on it-case/epic- what's the issue number working on it. Technically, all the program managers can look at the issue-tracking system and get the report directly. This is a process change.

Suzanne: We call that moving from a push process to a pull process. Instead of us pushing PowerPoint slides out to you, you are pulling the data from the source.

Hasan: Exactly, from the source. That is the beauty of DevOps again. Similarly in cultural elements, people have to share what they have. It's a discipline. It's not really as just developing coding we're expecting people have to be more responsive because one of the elements of collaboration is we have to share what we have.

Suzanne: We have to communicate about what's in it.

Hasan: What we find that people are not really sharing, they are not really communicating. So communicating means, *I send email I'm done, but here is the follow up*. So having daily meetings, for example, or having a collaboration portal makes it much easier to communicate because the information will be there. One of the other things we found out, as Jose mentioned before, is we have people are turning quickly like leaving their jobs in 2 to 3 years. What they have to leave behind is the knowledge. So how we can collect the knowledge is based on what they have done in the application lifecycle. We cannot really put the computers in a frozen mode and get from there. We are expecting every action has to be recorded somewhere in the DevOps pipeline, either a note or either messages or maybe project information.

Suzanne: ...script.

Hasan: Script, whatever they are developing as an artifact, has to be in the pipeline because this is giving an opportunity for the organization. When the people leave and we go somewhere else, that knowledge is left there. Somebody will come and take it and use it and continue.

Suzanne: Government environments are often quite accustomed to this concept. I don't know if you have ever heard of a continuity binder, is a concept when you change commands. When you move from one assignment to another, you're supposed to leave behind a set of notes that tell the next person in your job how to do that job. What we are basically saying is instead of waiting until a month before you leave you should be providing that all along the way. There are some concepts that we can actually latch onto that are more traditional, but we are using a different kind of technology and a different cadence for bringing that knowledge together.

Hasan: It's a technology also requiring a discipline inside the cultural changes. Being able to take the notes for ourselves. If you don't share the notes with your peers, the notes will be useless. Also, one of the elements of DevOps is we have to have the feedback come timely. Feedback has to be on that moment. We don't want to keep information for ourselves. We need to take something. We may have many stars on our development team. They are great, but if they are not sharing what they have done, it is useless.

Suzanne: Somebody can trample over something else that is done.

Hasan: Right, and share what problem it is and changing about the problem, analyze the problem, and share the feedback and also the team member and record it. We don't want to repeat the same problem making again because that's one things we would like to eliminate in the rework. We would like to eliminate any type of recoding, because we can use the centralized place. We can use the communication portal. We see what the teams are doing together.

Suzanne: This is one of the connection points between DevOps and Agile methods is this idea that we are doing collaboration and sharing all across the way, not just using the tools, right? That it is not just about the pipeline. Now, one of the things that I know about that we have had some success with in sort of these regulated environments is the idea of a continuous authorization, a continuous ATO, authority to operate. I think that is probably one of the most…It is one of the innovations that many of our listeners are going to be very interested in, because it is a new way of thinking about securing what we are calling the software factory or the software pipeline. Talk for a minute about how that is playing into your research and what you are finding are the abilities to actually do something like that.

Hasan: Actually, we should have another podcast on this topic. We cannot cover everything. I have been giving a lot of talks on this. If we have a pipeline that is available as we set it up.

Especially for ATO process, if we look at the Risk Management Framework, it is requiring a continuous monitoring. It is requiring continuous evolution, continuous testing. What the information assurance people are looking for is, *Give me data*. Data is going to show that, as the developer or architect you have, a control has been selected and approved and as in evidence was part of the code as in evidence, *I verified because I coded. I tested that*.

There are two ways you can collect data: either manually have a checklist or, as you said, like, pull the data from the systems, which DevOps is going to give a capabilities platform that IA people or whoever the authoritative approving the application. As we go to production environment they can pull the data directly from the DevOps pipeline. *Why didn't we change the CI process*? Make it a security testing as part of the work they are expecting and get the result of the report from the CI server as maybe an email, PDF, whatever…

Suzanne: *Come look at this*.

Hasan: Right. Whatever the information assurance people are looking for and then create that data directly from the CI server—that is when we can do the authorization process. Other things such as the elements of the infrastructure pieces, if we look at all of the controls are looking for the infrastructure elements like what the type of OS are using, it is patched or not. These are automatically as part of the program which is 80 percent authorization is done, because we are not really replicating the environment over and over again. Technically we are repeating the same environment. Whatever we are changing, we are changing on the application layer. The application layer is probably 20 percent of the controllers. Why they are spending a lot of time and money looking for the 80 percent over and over and again? With the DevOps concept, it's already done. It is already approved. You just look for the 20 percent person, which is the only coding section that we have to approve. So we'll talk about more in another podcast.

Suzanne: We'll have another podcast just about that. To me, that is one of the most exciting opportunities that DevOps is giving us, is that ability to stop thinking about, *We must look at the entire application stack every time we deploy*, to, as you said, *I need to look at 20 percent each time I deploy new, but I continuously get feedback on the infrastructure and the pipeline*. That is pretty amazing.

Hasan: Whoever has done DevOps properly as a maturity level, I saw they were able to do more ATO. Or, if they are in a highly regulated environment such as financial sector or in the health industry, they have to do the same approval process. There is no way as a developer they can release the code into production. It may happen in one- or two-person type of shop.

Suzanne: This is not Facebook.

Hasan: Honestly, if you are in a compliance-type organization, and if you are serious about your product, nobody would release any code into the production without giving approval. You may call it ATO. You may call it something else, but it has to be approved. They may have their own type of controllers. They have own type of procedural things.

Suzanne: And they have their own standards that they have to comply with.

Hasan: I actually analyzed a couple of compliance like, *Look for the* GDPR. *Look for the* SOAX, *HIPAA*. All of them are looking for auditabilities and reporting and traceabilities. You can get all three elements as part of your DevOps pipeline, which you know it is integrated. You have full traceabilities. Especially in GDPR, you need to really show the auditor how you are taking the private data is part of the pipeline.

How are you going to assure that the developer is not really touching the production? Either you say, *Yes, yes, yes*, but the person will be reliable to say *yes* or *no* to the auditor or they will look at the artifact of the system [and] say, *Yes, I trust it*. This is just a way of looking for the DevOps pipeline.

Suzanne: One question that I know will come up for people that are new to this is, *How much time and effort should I allocate to moving over to a DevOps kind of environment if I have been in a traditional tooling environment*? Because it is a different way of working. It is a different way of setting things up. There are organizational things as well as technical things. Do you have any kind of heuristic in terms of, *Start six months ahead of when you want to have your first deployment*, or anything like that that will help our viewers understand that?

Jose: I couldn't tell you an exact calendar amount of time. The biggest obstacle that we find when trying to get DevOps into, I am assuming, any environment but more importantly highly regulated environments...

Suzanne: Right, HREs are what we are talking about.

Jose: ...is the cultural shift. Now, when you go into an HRE environment, you have this phenomenon of people who are either in military, and they follow orders, or they are in a compliance setting, and they follow process and policy. They have been doing this for a long time, so it is very routine driven based on requirements that they need to follow at the organizational level. The HRE physically itself has its own requirements that you have to follow. We can put all the tooling in. We can make all of the recommendations. If you can't get the culture to shift, it is not going to work. It is just going to fail horribly. One of the things that we do is we sit with them on a project for a while to see how it's working for them and where changes need to be made.

Hasan: Also other things we advise and we work closely with our SEI clients is building up the pipeline along with building up the culture. Let's say we are going to start from repository management, which is the 101 thing in software development think ideas. Get the repository first. Let the people communicate, which is a cultural environment. You have the tools. You set it up, and now you have the cultural pieces, which is communicating the same repository. That is how we start. Then after that, *OK, now we are going to talk. How are we going to talk*? *We have some ideas of the sharing. Let's build out the wiki page*, as an example, or, *Let's integrate the repositories*. Now the next step is, *We don't want to do any work without having any case assigned to me*. *Now, we are going to build up the case management system.* So, building the cultural and building the tool works very well because they are learning when they are building. Also they are understanding about whatever environment they are in. Also, they are finding the solutions by themselves because they are not really getting a somebody else pipeline putting and then changing versus building the pipeline along with building the culture. That is the most effective way of doing DevOps transformation. Build it, work together with the rest of the team members and build together.

Suzanne: What I take away from that is that I don't necessarily have to have a six-month preparation process just to get started. We can look at, *Where are the opportunities? Where is the low-hanging fruit? How can we get some of the elements going in what we have now?*

Jose: It splits in two ways. You have the tooling, which that can be set up, and people can be trained to use that. That can be six months, three months, doesn't matter. For the tooling, you can sort of schedule it.

Suzanne: You can bound that, yes.

Jose: It's the shift in getting the people involved to now accept and adapt to this new process. That takes longer. Like I said earlier, we monitor, and we watch, and we check everything that we recommend to see, *Are you doing this? Show me. How is it working? Do you see it better?* The cultural shift becomes easier when people see benefits. *It is costing less money*.

Suzanne: *Is it taking less time?* That is a big one for people.

Jose: *We are hitting ATO faster. We are deploying faster. We have fewer defects. We are finding them earlier*. When they see those benefits, they are like, *Oh, yes. This is good. That is what we try to follow*. When they don't see it, we revisit it and try to modify.

Hasan: Rebuild and then build the culture together. That's an effective way and also other things like you said how, when they can start. All the stakeholders should understand oral taxonomy about the DevOps. I see a more different type of definition, especially with CI [continuous integration] or CD [continuous deployment]. Spend maybe a week, understand all the terminologies. Set a rule about the communications, and then start building, *OK, what is our problem? What do we have our problem?* So when we do the DevOps analysis for our client, we are talking with them to find out, most of the time we are hearing from them, *Aha, we have a problem here and there*. They are able to identify their own problems. Basically we are guiding them to build up that knowledge, *Let's build it. You have a problem, let's go fix it. How are we going to fix it? Sometimes it is tooling, sometimes you need to spend the time to understand that when we build it, or sometimes build up the culture*. So it is kind of like going ahead…

Jose: I will give you an example of a cultural issue that we came across that has nothing to do with tooling. In some cases to start on a project, you have to get senior leadership to sign off on it. They approve the money. They approve the work. In several cases we have seen that part of the SDLC [software development lifecycle] that group is following, is this, *Prepare the project. Define the project. Put in the paperwork and send it off for authorization, and then start working*.

When it goes off the authorization, there seems to be a lot of confusion up there, different people start to claim, *No, you shouldn't authorize this. I should authorize this. This is not yours. This is mine*. You get this back and forth, we learn this from the developers. This type of back and forth consumes time, consumes the project. It is a cultural shift that has to occur there. It is not a tool. It is not a piece of software. It's talking to see, *Well, why is this occurring in the first place*?

Suzanne: Why do we have this conflict?

Jose: *Is there a confusion into who manages who, who is in charge of what*? Once we had that, it became much easier and much more effective. This is one of the key things in HRE. We deal with a lot of problems that we can solve with DevOps concepts, but we also have to deal with a lot of problems that are solved with non-DevOps concepts.

What we have learned is for every group that we sit with, we end up creating a tailored SDLC from conception to sustainment, post deployment. With DevOps on top and non-DevOps as well, you end up with a tailored SDLC for each group. DevOps alone in HRE won't solve absolutely everything, especially on the cultural side. So we combine the two, and that is something that we learned along this journey.

Suzanne: Much of what we have talked about, we have talked about some other things, but I know that you have recently published a paper at the 2018 International Workshop on Secure Software Engineering. This idea of DevOps in HRE was the topic of the paper, so that is one of the resources that our viewers can look at if they want to know more about what we have been discussing in the podcast.

What is next? We have dealt with DevOps. We have dealt with stakeholders of different kinds in DevOps. We have dealt with highly regulated environments as a variant. What do you see in the research side as things that we still need to research in the DevOps arena?

Hasan: There are a lot of research areas that exist in DevOps, especially looking for the deployment perspective and how can we get containerization concept into DevOps pipeline with a secure mindset. Also, how much we can do architectural automation as part of the DevOps pipeline, because one of the big elements is you may have a great pipeline, you may have a great culture, but how about for architecture?

Suzanne: Yes, if it is a tightly coupled architecture.

Hasan: Right. If the architecture is not compatible with getting it built every day, then how are we going to build the system and take advantage of the build process, automation process, because architecture will be a barrier.

Also, in other research, we have been working on building up the return on investments. What DevOps is building in the DoD? Because we know why industry doe DevOps and they have right data sets, *How much money am I saving? What is the long term [benefit], especially for very complex systems?* That's another research area that we are working on.

Suzanne: OK, so you going to be busy for a while.

Hasan: I have never been as busy in my life since the last couple of years DevOps.

Suzanne: I know. The only time we get to see each other is at conferences and things like this because we are both running around so much.

Hasan: Absolutely.

Suzanne: All right and, Jose, you have been running around too.

Jose: Yes, I run right next to him.

Suzanne: There you go, there you go. I really want to thank you both for joining us today. I love talking about this stuff, and I think our audience gets a chance to learn a lot. We do have a series of blog posts that Jose has been working on. We are going to have those in our Insights area at the SEI. You have a DevOps section in the Insights area on our website. That is a place to look for some other material that relates to what we are talking about.

Hasan: Plus other community engagements. We have been running DevSecOps Days and All Day DevOps. Just follow us on the SEI website.

Suzanne: Yes. I am going to plug All Day DevOps just for a minute. This is a 24-hour period with a bunch of small presentations, electronic. It is all remote. It is a hoot. You can dive in. You just dial up at any point and get in on what is going on the screen at the time.

Hasan: I was awake 24 hours, literally I was awake.

Suzanne: I have not done 24 hours of All Day DevOps, I will admit, but I have done a few hours.

Hasan: Ready for 2019? We can run together.

Suzanne: One day you ran it on my birthday. That is when I only did an hour. I want to thank you both. Thank you.

We will include links to all the resources we've mentioned, the blogs, the papers, and there is more. Just go to the DevOps Insights, and you will get all kinds of things. We will have more discussions coming in the future.

Hasan: Absolutely.

Suzanne: So we will do that.