Want better DevOps? It comes down to people and leadership

For years, IT organizations have struggled with siloed development and operations, long lead and cycle times, and poor operational quality. DevOps was born as an attempt to address those pain points, but companies often have problems shifting to DevOps. For many, it’s often because they view DevOps as primarily a “tooling” solution rather than a “people” solution, and they’re reluctant to blur traditional organizational boundaries to build new team structures. In this episode of the podcast, Mike Kavis and guest Scott Prugh, discuss how to implement DevOps effectively. Scott’s solution is simple in theory, but often difficult in practice: get the right people, get good leadership, view the process as a product—not a project, and integrate governance fully into the DevOps process.
DOES is, that's the DevOps Enterprise Summit. So, welcome to the show, Scott. Tell us a little bit about your company, about your background, and we're going to get talking about DevOps.

Scott Prugh:
All right, well, thanks for the intro and I'm happy to be here. So, my background really kind of started hardcore in software architecture and programming where I developed a lot of enterprise systems, systems for startups. And then probably about eight years ago, I started getting kind of more into leadership and really looking at how organizations were structured, what processes they kind of carried out, and really looking at kind of lean system thinking across the enterprises, both agile and lean, and then kind of later on DevOps, and really looking at actually how we create high-performance organizations that can deliver and operate software much more effectively.

CSG itself is one of the largest and kind of traditional SaaS providers. They've been the SaaS provider of customer care and billing for cable companies in the US, so major customers like Comcast, Charter, Time Warner, Dish Network use our software kind of really to support their front office and back office. And we have about 63 million subscribers, cable subscribers in the US, that we manage the infrastructure and the billing for every month.

Mike Kavis:
Well, cool. So, I first saw you onstage, I don't know how many years ago. So, you're a frequent speaker at the DOES Enterprise Summit, the DevOps Enterprise Summits, and one of the things I like about that summit is they have repeat offenders like yourself who, every year, we get to see where the DevOps journey's going. And I've probably seen you three or four times now. So, before we start getting into the specifics of some of the things you've talked about, what were the drivers that led you to say, "Hey, we need to embrace these DevOps concepts that are coming out of these thought leaders, coming from all these other companies"? And what led you to say, "We need to make a change and we need to go down this route"?

Scott Prugh:
Yeah, so I mean, the journey was kind of a long one I would say, and it started really in 2012 with what I would say - kind of our agile transformation. We really had looked at - we had releases every 18 months. We had looked at our lead times and they were just really not that great, and we had a lot of traditional, kind of waterfall-based, processes to design, develop, test, operate - you name it, really kind of every handoff in the book building software. So, we originally looked at those and said, "How can we address both cycle time and lead time of improving our delivery?" And that really kind of led us to reorganize around cross-functional agile teams, you know, the traditional design, build, test type teams. And we did that for several years. We saw great results as far as software quality and lead time for delivering the software for our customers. But we still struggled with what I would say kind of is operational quality, and then a lot of I would say friction between development and operations. You know, building software being very different than running it, you would give software to operations; it would take them weeks to get installed. You would have a lot of kind of operational impact and a lot of customer dissatisfaction from those things.

So, we kind of continued to look at that over a couple years, and then in '14, I was fortunate enough to run into the gentleman by the name of Gene Kim, and I was kind of talking to him about these problems, and he became very interested and then since kind of became a really good friend, about how you kind of correct these things. And I was already starting to put in things like, well, what are the operations teams that sit in a different area than the development teams, can we even get them to sit together, to have a standup in the morning together? We were already starting down those paths before really kind of the DevOps movement, but it really kind of started to solidify that, for me what DevOps is looking at is really kind of extending both the agile and kind of system thinking components of really having more cross-functional teams, both build and operate.

And, really, kind of all this came to kind of a head in '16 where we just really could no longer meet the operational expectations of what our customers wanted and I had put a proposal out there, where, well, why don't we collapse the operations folks into the same teams that build software? And obviously you get kind of a lot of objections, you know, that you can't have development and operations on the same team, you know, they have to be separate; they have to have a handoff because of certain kinds of compliance restrictions. And I was like, "Well, no, there's nothing really that says those people can't be on the same team. They can be on the same team; they just have kind of different roles. And we can put all the same checks and balances in through version control, through build deployments, et cetera," that, you know, I think people today know that that can be done, but at the time obviously there was the general - what I would call organizational antibodies that come out and say that you can't do those things.

Given all that, we were still able to convince folks that that was the right way to go to kind of improve. And so in '16 we really made those kinds of cross-functional teams in place, and since then we've put a lot of things on top of that to continue to improve. But it's been what I would call kind of a smashing success and an example of how to improve both the operational quality and then allow the other components and cycle time, lead time, delivery of features, operational impact have all improved significantly since then.

Mike Kavis:
Yeah, and when you get up on stage and you talk about those things, you never have enough time to talk about the lumps you had to take and how hard it was to make those changes. And that's one of the things I wanted to focus on now. I mean, just -- you know, you're crossing organizational boundaries. Usually there's a VP of Ops and there's a VP of Dev and you have to put them in one team, and that disrupts a lot of org structures and ownership. How did you tackle that? And how hard was that, to make that shift?

Scott Prugh:
I mean, that is the hardest problem, and I kind of quote to folks that that kind of quote to folks that I think one is, organizational boundaries represent one of the most difficult problems, because someone's got to lose real estate generally when you do these things. There are examples, and we saw it work on small scales where we would have support of leaders, have their operations folks work next to development folks and kind of co-locate. But at the larger scale, we weren't able to kind of convince people that that was the right thing to do. You know, people have different drivers, right? You know, help desks optimize for first call resolution. Operations folks optimize for the numbers of changes and service requests they can process per day. There's all kinds of, really, what I would call kind of individual, siloed KPIs and OKIs that people have that, in effect, de-optimize your ability to make the overall system improvements. And those are -- they've been in place, they've been in place for years, and they've kind of become both the operating model and the cultural norm. And trying to break those is extremely difficult.
That’s one, the kind of organizational structure impediment, and someone’s got to lose. And then the second problem that I think is a really big one is going
to that next step to kind of combine these things together, it takes a different type of leadership. It takes both leaders that really can work across multiple, I
think, technical domains – in other words, they understand STLC processes. They understand operations. They understand how to build products. And for
organizations that have traditional siloed roles – in other words, they’ve had folks in place for many years that are really good at test, or really good at
operations, or really good at just doing development. It now puts a focus and a stress on having the right leaders that can go across those things, because
it’s not necessarily always going to be the same person that can lead an operations group that can also lead a cross-functional software delivery and
operations group. It is a different and much broader skill set, so those people are harder to find, so I think those two things were a challenge.

Now, you know, we had shown both first with our agile delivery and the improvements there that we had the credibility and that we understood a lot of the
system, basically, dynamics at play that were preventing us from going to the next level. And then second it was an opportune time because of the fact that
we were suffering operationally, really kind of like, "Well, could it be worse if we did this?" And, so, the combination of having established credibility, of
having done this – you know, a bit of it in the agile. And then, second, really kind of getting to a breaking point where it really wasn’t getting better,
expectations were getting higher, which meant – at least to our customers it was perceived as we were getting worse. And those things kind of came to a
head to say, you know, this is something that we’ve got to try to kind of move forward.

Mike Kavis:
Yeah, pain is always a big driver of change, that’s for sure.

Scott Prugh:
Yep.

Mike Kavis:
So, another thing you had to do in all that – you know, it’s one thing to shuffle the chairs and get teams working together, but when you break down those
boundaries there’s a lot of process change as well, right? A lot of these handoffs go away; a lot of these checks should go away. How did you go about
looking at value streams and getting all the right players in the room and going through that? And then, it’s the same type of thing. It may not be a real
estate grab, but some people’s day-to-day job was to do this thing and now we’re going to change it. So, how did you deal with that as well?

Scott Prugh:
There’s a lot of stuff that I think happened fairly easily in the kind of day-90 timeframe. In other words, in the beginning we said, "We’re going to do this."
We, in effect, got managers and leaders together and had a spreadsheet with 600 people and really moved them onto the teams to both build and run the
products. And, for what it’s worth, that was actually one of the easier things. In other words, going through and saying, "Okay, let’s align these people with
the right value streams," was actually not that difficult, fairly straightforward. It took a while for us to get to the realization, but once we got to the
realization, that administrative aspect was fairly straightforward.

The next 18 months, I’ll tell you, was hell, and it was a problem not because of tissue rejection or we weren’t doing the right thing. It was a problem because
we now subjected that entire team to the realities of what it took to actually operate in production. So, in other words, we basically – folks who’d never
before had to experience operating their product now had to understand that, and for them that was pretty extreme. For our leaders, i.e., me and others, it
was an experience of great turmoil, because we were now exposed to operational outages, you know, the reality of talking to CIOs from our customers,
stuff about operational performance and how we weren’t meeting their expectations, you know, getting paged in the middle of the night. All of those things
in the first 18 months really hit myself and my leadership team, and then of course all the folks down which are the most important, pretty hard because
that was really kind of the stark reality of now becoming operators of the software you built.

And then after that – and also during that 18-month period we kind of also started looking at other, what I would call tooling-process, problems. So, when a
company grows up as a set of what I would call process silos, each process silo has a set of tools that help them operate on a day-to-day basis, right? So, for
example, the operations team has some sort of service-management tool. You pick one of them; they’ve got it, right? The software teams have STLC tools.
You know, product management teams have their tools. And when you start to converge these folks together, you now realize that each time is kind of working
out of multiple tools to actually get work done on a day-to-day basis. And that creates a lot of cognitive overhead for those teams as they swivel, but,
additionally, you don’t really have the ability to kind of see the work as a whole, and so you get really kind of a lot of disjointed pieces.

So, we kind of started to embark on – and we wrote a paper on this, on combining work management systems together in kind of a DevOps transformation.
It’s not something you would think of in the beginning, but then when you realize that you’re, in effect, now punishing these teams with the tooling friction,
you start to think about how important it is now to really kind of create that single backlog of work, and now look at how the tools function, and then really
how you need to kind of transform those tools. And that was a big kind of piece of the administrative that we saw that actually had to be changed.

Mike Kavis:
Yeah, that’s a pretty interesting story. I’ve always said DevOps is all about removing bottlenecks, and a lot of times you don’t know what they were until you
move the one in front of it. And that’s probably an example right there, where it took until you got everyone on the same team to realize that there was a
huge bottleneck in the tooling. So, that’s cool, but one of the things I saw you talk about – I think it was two years ago – which was one of the most
intriguing things to me, and it speaks to the concept of this is about moving bottlenecks, is after the first couple years I think the CI/CD process kind of – you
kind of got it, and it’s a commodity – we got it. What’s the next thing? And the next thing was security, and everyone’s DevSecOps and they start getting
that in the pipeline.

Okay, what’s the next thing? And the two next things I saw you talking about two years ago was the shifting of the governance to the left, not so much the
policy definition, but the policy implementation, so you talked about that, and then you talked about what’s next for next year, and you were starting the
journey of looking at tier-one-through-three support. And I never got to hear where you landed on that, so talk about how – you know, we’re moving all
these bottlenecks. We kind of got over the security one a little bit, and now this governance bottleneck’s there, and what you did to shift some of that
accountability and responsibility left, and then what’s your one-through-three support.

Scott Prugh:
Scott Prugh:

You can follow me on Twitter at @ScottPrugh, and then all my presentations from DOES are on the DOES YouTube channel. You can find those there.

Scott Prugh:

today's episode of Architecting the Cloud. Where can we find your work, your presentations, your Twitter account and all of that? Because I really

Yeah, I remember that talk. I was there. It was pretty powerful, but I love this whole conversation because I struggle with a lot of folks who DevOps to them

Scott Prugh:

You can follow me on Twitter at @ScottPrugh, and then all my presentations from DOES are on the DOES YouTube channel. You can find those there.
Mike Kavis:
Cool. And you can see today's notes, show notes. They'll be on www.DeloitteCloudPodcast.com, and you can find more podcasts by me and my colleague Dave Linthicum just by searching for Deloitte On Cloud Podcast on iTunes or wherever you get your podcasts. I'm your host Mike Kavis. If you need to contact me I'm MKavis@Deloitte.com, and you can always find me, @MadGreek65, on Twitter. Thanks for listening and we'll see you next time on Architecting the Cloud

Operator:
Thank you for listening to Architecting the Cloud, part of the On Cloud Podcast with Mike Kavis. Connect with Mike on Twitter, LinkedIn and visit the Deloitte On Cloud blog at www.deloitte.com/us/deloitte-on-cloud-blog. Be sure to rate and review the show on your favorite podcast app.

Visit the On Cloud library
www.deloitte.com/us/cloud-podcast

About Deloitte
-------------------------------
The views, thoughts, and opinions expressed by speakers or guests on this podcast belong solely to them and do not necessarily reflect those of the hosts, the moderators, or Deloitte.

As used in this podcast, “Deloitte” means Deloitte Consulting LLP, a subsidiary of Deloitte LLP. Please see www.deloitte.com/us/about for a detailed description of our legal structure. Certain services may not be available to attest clients under the rules and regulations of public accounting.
Please see www.deloitte.com/about to learn more about our global network of member firms. Copyright © 2019 Deloitte Development LLC. All rights reserved.