Deloitte.

For Cloud Professionals
November 2019

Architecting the Cloud, part of the On Cloud Podcast

Mike Kavis, Managing Director, Deloitte Consulting LLP

Title:	**DevOps agility requires shift in focus to product value streams**
Description:	Despite the rise of agile software development methods and DevOps to organize IT teams, software delivery is often still impacted by the very bottlenecks DevOps was meant to alleviate. As a result, many companies still struggle to get to the cloud faster and innovate. In this episode, Mike Kavis and guest, Tasktop Technologies founder and CEO Mik Kirsten, discuss how companies can accelerate what's possible with the cloud and build a culture of innovation. Mik's solution? Focus on flow rather than milestones. Shift from individual projects to delivering products by re-architecting product value streams, removing bottlenecks, and measuring the success of software development and delivery by the business value it provides.
Duration:	**00:23:54**

Operator:
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. Welcome to Architecting the Cloud, part of the On Cloud Podcast, where we get real about Cloud Technology: what works, what doesn't and why. Now here is your host Mike Kavis.

Mike Kavis
Hey, everyone. Welcome back to Architecting the Cloud podcast, where we get real about cloud technology. We discuss what's new in the cloud, how you use it, and why, but we talk to people in the field who are doing the work. So, we get the real story here. I'm your host, Mike Kavis, chief cloud architect over at Deloitte, Today I am joined by Mik Kirsten, CEO and founder of Tasktop Technologies, and, also, author, which we'll talk a little bit about his book coming out. Tasktop Technologies provides enterprise software solutions. The company offers test, project, portfolio, and service management solutions, as well as product development, integration, training, certification, and opensource solutions. Mik, thanks for coming to the show. Tell us a little bit about your background. Tell us about your book, and what kind of drove you to write that book.

Mik Kirsten:
Sure. Great to be here. Thank you, Mike. My background has been, first, as a – I spent a decade as a developer just writing a whole lot of code. I started realizing through that journey that the way that developers tend to be disconnected from what we now call product value streams, was actually kind of the biggest impediment to my own personal productivity. The fact that there's this constant flow of issues, and features, and security fixes to make on the projects I was working on, and that my coding environment was really disconnected from that whole – all of those different planning, tracking, and workload systems. So, I first set out to solve that with – by creating a series of popular developer tools that would integrate these issue trackers like G-Run into developer workflows. And then, I actually founded the company Tasktop to solve this problem at infrastructure level.

So, I realized, okay, this is a bit deeper than just the fact that we've got multiple tools, and all these best-of-breed tools. There needs to be this federation of the end-to-end value stream so that information can flow, and people can collaborate all the way from – to the business side, ideation, through planning, portfolio management, development, testing, deployment operations, and so on. And that's really Tasktop. It's this tool that connects the entire end-to-end value stream. Now, the interesting thing that happened is as we were deploying this tool at large enterprises over the last five years, we started getting a ton of insight into how these value streams are structured. You know, one of the key realizations I had was that the ground truth of how software is built, so the equivalent of the factory floor, is actually those tools within these organizations.

That's where information flows. That's where the collaboration happens. And the amazing thing is it's all captured. It's all there. We receive this amazing lens of it. We actually collected data on 308 of our customers' value streams, and we started gaining some real insights. That was one of the key pieces of emprical information of data that went into understanding what was wrong, where the disconnects are, and how misaligned these -- basically, these product values streams are, these values streams from these organizations, the software architecture, and the organizational structure. So, seeing how much of a bottleneck this was to digital transformation, to really getting the benefits of agile and DevOps actually inspired – was one of the key inspirations to writing the book, to writing Project to Product, and to help organizations think differently at a business level, at an organizational structural level, in terms of how they should operate and manage their software value streams.

Mike Kavis:
Yeah, so get into the book a little bit. We're trying to shift people's mind from thinking about individual projects, to thinking about products. So, explain the thinking behind that, and the tell our audience, what's the theme of that book?

Mik Kirsten:
When working with these large organizations, I realized something that was initially just completely foreign to me, right. I spent my life working at software companies, open-source projects, startups, and I realized that the way that we thought about an organized – and really at a business-level—managed software delivery because for the last over a decade now, I've been an entrepreneurship and a CEO of a software company. No longer writing code. But the way that myself, my peers, my colleagues, the tech giants, thought about and managed software delivery was just completely different to what our customers were doing, these large enterprise organizations were doing. I realized the difference was that the managing principles in terms of what was done, how it was done, how was it organized, how people were assigned, were project management principles.

Those project management principles, it turns out, they simply don't work if you want to innovate, if you want people to get really good at the technology stack they're working on. If you need a fast-moving value stream specific to cloud, where can you do blue-green launches and so on, and there was just this mismatch. Companies were hearing about how to become innovative. The IT leaders I was listening to, and CIOs wanted to drive their companies forward. But the organizing and management principle that was in place was actually working against that. So, another way to look at is these large companies are trying to be agile, but they're really just doing agile development.

They're not actually thinking agile end-to-end, or they're not able to implement. Even if they're thinking it, they've not been able to implement agile end-to-end because, in the end, you have these waterfall, these very iterative – less iterative, but very stepwise processes, to the left of agile delivery, and often to the right. So, to the right, DevOps practices have helped solve a lot in terms of deploying, but it's still not enough. If you've got any kind of manual gates on that side, or if you've got bottlenecks up stream. Or I think the bigger thing if you're using project management, and basically tracking activities and costs rather than what the whole Project to Product movement is about is measuring value delivered through your software value streams, through these product value streams.

So, the realization for me, and really the core thesis of the book, is that if you're measuring project milestones and activities, you have no idea if you're moving faster or not. You are measuring the wrong things. These are proxy metrics. If you're just measuring deploys per day, you have no idea if your investment in IT is actually producing more. All you know is that you're deploying more often a day. It's not telling you what you're deploying, how much value you're adding to your customer. So, this is really the main shift is have us identify, at the level above agile, above DevOps, product value streams as an organizational construct. And then to measure, not activities, but the flow of business value through those project value streams, and then also measuring and correlate that to actual business results. That's really the whole Project to Product shift is to change how we organize, and measure, and manage software delivery.

Mike Kavis:
Yeah, that's great. It's funny, I've been asked to present internally here at Deloitte on agile and DevOps. My presentation that's coming in a week is more bashing agile than it is promoting it. But not that there's anything wrong with it, but my point in this presentation, and I've got this slide where it's showing the agile manifesto. And I highlight in almost every sentence, "software developer", "software developer", "software developer", like where's the ops, right? There's no talk about the ops. Where's the customer? They don't talk about the customer. Then I bring up the safe framework. Yeah, we've integrated DevOps and all this.

Where's the ops? It's still a hand-off to ops, and on, and on, and on and the point is that there's doing agile, and there's being agile, right. Another example I always like to use is we were doing a development for a client, a 90-day development. It required some new tools, and the procurement process was six months. Well, that's not agile. I don't care what methodology you use, you know, that's part of the value stream. And, too often, people think of DevOps and agile as dev and ops. It's really the whole value stream. That's kind of, I think the point that you've just raised.

Mik Kirsten:

That's exactly the point. Right. It's the doing agile versus being agile point. Exactly. I think what's happened is there has been such a push in these organizations to move towards doing agile, basically, that it's the implementation of agile, a) it goes against some of the principles of agile, which were meant to be the leading principles around end-to-end flow, but, b) it absolutely has been an overly developer-centric movement. So, I'll just give you an example, a very concreate example that's brought up in the book, but I think it's a very powerful one, where we analyzed the product value streams of one of our customers. These were more length of projects, but that's fine. you need to start somewhere. At the time, they were making the shift to thinking in terms of product, and are most of the way they're now.

But we looked at the end-to-end flow time because – by the way, the Project to Product book is really a vehicle for the flow framework, which has these four flow metrics. One of them is flow time. How long it takes from a customer's point of view, from a business's point of view, to deliver value. Not from the agile team's point of view, not user-story open; user-story closed, but all the way from inception to code running, right, through operations. We measured it, and the average time was 120 days. This was a shock to the CIOs of this organization because they thought that having deployed agile things would move much faster. Having done some of the principles of the DevOps, things would move much faster.

But they were actually not – no one was really measuring the time end-to-end until they saw this. And when they saw this, this was the amazing thing, the amount of time spent – because were able to measure how much time was spent on each tool working on, let's say, a feature or a security fix or something of that sort. The amount of time spent in the development was two-and-a-half percent of that 120 days. So, around three days. So, this obsession with agile and – by the way, of course, like everyone else, and to your point, everyone at that point thought that their main bottleneck was they didn't have enough developers, and they needed to hire more developers. They needed better developer talent. It really is very often the gut is telling leaders is they just need more developers. This was data that proved exactly the opposite. They could have double the developers, that's realistic for them to double developers because it's a large organization. But if they could double the number of developers, things would not move any faster, right, because of the fear of constraints. And like everywhere else, like you're saying, Mike, it's the same thing, right. Things are thrown over the fence to ops, even though the people are saying DevOps. Things are thrown over the fence onto the agile teams. These agile teams have these insanely long wait states. And, sometimes, like you said, it could actually be they need to procure a piece of software. So, unless you're actually looking end-to-end to and measuring, looking at things end-to-end, you're failing on both the skirt of agile and of DevOps, right, so of flow and feedback stuff. So, that's what we learned.

Mike Kavis:
And that's why the measures are important. If all you're measuring is velocity, that just means people are doing something fast. It doesn't mean they're producing any value. It doesn't mean the customer is happy.. I was on a team one time where the leader was complaining that our team's velocity wasn't the same as another's. I'm like, well, that's like comparing my blood pressure to my wife's. She runs cold. I run hot. Those are our normal. You can't compare. it's comparing apples-to-apples. You need to compare our velocity against what it was yesterday, not against any other team. But at the end of the day, velocity doesn't mean anything, right. So, what are the metrics that you lean on in this flow framework that people should be looking at?

Mik Kirsten:
Yeah, so I think that's the key question. I want to be clear about something. Measuring actual, the actual end result of value delivered is hard. So, I've been struggling my whole career through my research, and so on, to come up with meaningful measures of productivity and software delivery. And the flow framing does something really interesting in terms of measurement. It doesn't go and measure, like let's say, technical, that's an important thing to measure. It doesn't go in and measure the source or the base. There's no analysis of what's there. It measures the amount of work flowing through the product value streams. It does this by forcing the organization and the business and technology side, to identify this four flow items, which encapsulate all business values.

So, features, defects, risks, and debts. Features are new business value. Defects are quality fixes, outages fixes and so on. Risks are security compliance, privacy, and so on, and debts are technical debts, be they on the infrastructure side, on the development side, maybe even if we can say they're on the procurement side in some cases, as you've mentioned. But the key thing is, then, all work items, all issues, all problems, incidences, and changes, everything is mapped into one of these four flow items. You start tracking the flow of these items, and you actually do get; you get flowtime, you get flow velocity. So, I'll get back to why the velocity is this, absolutely as you said, this double-edged sword. I'll get back to why it's relevant in this in a moment. Flow efficiency.

So, you see the comparison of wait states versus active work states on this item. And then flow load. How much basically work in progress. How much you're overloading your value streams. And the key point of this is what you're doing – I should mention one thing in the flow framework about product value product streams is they're meant to be a level above the agile team level or the ops team level. They're end-to-end value streams, which means they mean to be meaningful to their customer because the flow framework is all about what's pulled by a customer. They could be an internal customer. It could be the HR department. They could be a set of developers in the company needing some more APIs.

But you've defined the customer, and you're measuring what you're delivering to that customer by what they're pulling through those product value streams. So, we're measuring the flow of work, and then basically correlating those to business results. Because, again, it's not like a scaling framework. This layers over like the scale data framework, where you actually do it and give prioritization. It's really truly meant to be end-to-end, which it's agnostic of how you're doing planning of the items. You may be using more project management still. You may be using more OKR. You might have product management tools. But it's tracking things end-to-end. And then you're able to compare exactly as you said, Mike, what's happening with that product value stream.

So, one example that's very easy that we see over and over in our customers is if you put too much flow load by overloading the feature backlogs of those teams, flow time will go up, how long it takes, and low velocity will go down. So, if you're not taking into account the fact that the flow items are all mutually exclusive and comprehensively exhaustive, meaning you do more risk fixes, you do more work on, say, something like GDPR, you are going to get fewer features out. Then you're actually, again, ignoring flow load or a key component of flow load, and you're getting less done.

So, the whole point is that if you look at the value stream end-to-end, and they tend to be composed, by the way, between ten and 100 people, right. It's one in ten feature teams. It's all the operations people. You look at the thing as a whole. You look at the UX designers as part of the product value stream. You start being able to see, well, if we do more of this, did flow velocity increase or did it decrease? Comparing between product value streams, like you said, it generally useless because it's apples and oranges. But comparing how investing in tech debt reduction, investing in some infrastructure automation, how – what effect that had on flow velocity is actually super meaningful. So, that's the whole point of these flow metrics.

Mike Kavis:
Yeah, and one of things about the value stream, you know, often, it spans multiple groups with different incentives, right. It creates a lot of bottlenecks at the end offs. And the companies that really are growing fast, they actually re-architect those value streams. But that's hard, right, because that's people in process and org structure. So, what do you recommend to people when they have a value stream, and there's room for improvement in the process or organizational structure side?

Mik Kirsten:
To quickly answer is once you start doing this, once you start measuring flow, you will be architecting and re-architecting the value stream. But you'll be doing that, again, with meaningful data. And here's where flow efficiency is so key. You'll see that this particular product value stream, that developers are waiting on wireframes, or making widgets themselves, or opening the shop. Huge sort of inefficiency, potentially. But you see it. And you see it not within just one team complaining, but you actually see it stalling out many teams. You'll actually see that a bunch of feature work, or a bunch of defect work is held up on APIs that don't exist in a value stream that you depend on in some kind of shared service, or platform, or customer views.

Again, it forces you then to invest in that. So, the whole point is you start measuring flow that allows you to start seeing the bottlenecks. And then what you do is you either re-architect the value stream or invest according to the bottlenecks. Because it could be a staffing investment. It could be a deployment automation investment. But you're basically seeing where to invest based on this. I'll give you just a quick example. You know, we saw one of our bottlenecks was actually a set of APIs that were being too slow to be made in one product value stream. So, we moved that whole API team into the actual customer-facing product team for two release cycles, right, for two quarters. That worked. And then we can move them back if now we need that API to be used more broadly, which was the original intent.

So, you really start investing where the bottleneck is, and you're re-architecting, as you said, at potentially at a tool-chain level by looking at how your delivery pipeline is working; a staffing-team level, not by, again, doing this project-management madness of allocating people to multiple teams, which makes no sense. Our whole philosophy is that each person should be on no more than one product value stream. Or by making investments in actually how you structure the product value streams, themselves. So, being more horizontal, more customer experience-oriented, system of engagement-oriented, or be more platform ready, and so on. So, it's a journey, but the whole point is that you're actually basing the journey on whether that last experiment that you took, or that last initiative that you took, resulted in better flow or impeded flow.

Mike Kavis:
Yeah, and I think the one factor that makes all this work is trust, right. I mean, a lot of organizations won't allow that type of flexibility to talk about, hey, we're going to move this team over here. They're going to do this for two quarters and move back. So, trust is a big factor. You know, how have you seen companies overcome some of those hurdles?

Mik Kirsten:
Well, it's a big problem, typically, and I absolutely speak frequently as a big problem. So, the best advice I have right based on what we've seen, and the book does go into this, probably not enough, but is that the product value streams need autonomy. Ours have, effectively, budget-level autonomy, right. So, if they choose to move a team, or to staff up one team to basically keep another team at its current size, they need that autonomy. They need to be given a business direction and strategy, right. There needs to be a north star. That needs to come from the company, from leadership, and so on. That could be some kind of revenue result, bringing some cool new thing to market, whatever that is.

But then given that north star, they're responsible, for example, for their own flow distribution. So, the flow distribution is in, let's say over the course of this year, how much are we going to invest in tech debt reduction versus feature delivery, because is all tech debt reduction should be actually in support of future feature delivery. That's the only reason you should actually be investing in architecture. Of course, one of the big impediments we see is most large-scale IT value streams have their enterprise architecture misaligned from the product and the customer experience. So, the key thing there is that teams need that autonomy.

And at the higher level, above the product value streams, that's where we have the bigger decision made. For example, do we move this agile team, this feature team, from this one product value stream to another? You know, that'll often take an executive or a senior vice president or something of that sort. But within those product value streams, they need autonomy to maximize their own flow, and to understand their impediments and to fix them, and to continually improve, daily improvement for those impediments.

Mike Kavis:
Yeah, fascinating stuff. I've had so many conversations about changing the way we think, whether it's from an ops perspective, a build perspective, a product perspective. It's such a theme the last couple of years, and I think it comes from we're moving to the cloud, and we focus so much on the technology, and it took us so far. And now we're really looking at the human aspects of all this. Books like yours, and some of the others coming out, do a really great job of trying to drive that home. Cloud is an organization change. It's a process change. It's not just running microservices.

Mik Kirsten:
No, exactly. And it's got great aspects to it, right. These amazing technologies that we now get to leverage of containers and microservices. They really work well in terms of supporting flow and getting fast feedback in terms of deployment. And the issue is that every company out there needs to be investing in these things. Because if they're not in the bottleneck now, they will be at some point, right. You need to be able to go to cloud native, especially for product value streams, where you need that fast feature feedback, where you're trying to innovate on a customer experience and so on. But what we're seeing is that it's not enough, right. If you're only thinking at the feature team level, the agile stream level – oh, this team needs to be able to deploy to this environment; it's insufficient.

Back to your point, Mike, on the whole agile manifesto stuff, the main point I'm trying to get across with Project to Product is that you need this organizational construct that's bigger than the agile team. Those are these product value streams. And then to make sure you optimize the flow of those. A key part of that is allowing the key ones, especially the ones that need to optimize for fast feature flow, to deploy cloud native. But, again, if they've got these upstream bottlenecks because they can't get an authentication to technology that will make sense for your platform because they've been waiting three months on it, you're not leveraging anything from cloud, right. You're still stuck, and you've still got the old ways of working with a new backend. I think the key thing is measuring flow just shines the light on that. Organizations that are basically making good progress in terms of moving to cloud are trying to see that it’s just like agile. It’s not sufficient if you’re not looking at the end-to-end flow.

Mike Kavis:
Great conversation. Tell us where we can find you on Twitter, where we can find your book, and if you have any slide shares or blog posts, all that goodness. Where can we go find all this stuff and learn more?

Mik Kirsten:
Yeah, @Mik_Kirsten on Twitter. Just Google Project to Product. It’s Project to Product dot-org, where all the information about the book, the framework, and everything else. So, you’ll find everything from there.

Mike Kavis:
Great. Make sure everyone looks that up. It’s a really good book. I read it myself. Awesome stuff. So, you can find more podcast like this by me and my colleague Dave Linthicum just by searching for Deloitte On Cloud Podcast on iTunes or wherever you get your podcasts. Go to www.DeloitteCloudPodcast.com to get the show notes on this podcast. That’s it for today. I'm your host Mike Kavis. If you need to contact me, it’s MKavis@Deloitte.com, and or you could always find me on Twitter @MadGreek65. Thanks for listening and we'll catch 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