Deloitte.

Architecting the Cloud, part of the On Cloud Podcast

Mike Kavis, Managing Director, Deloitte Consulting LLP

Title:	**DevOps value stream issues? Check for time thieves and invisible work**
Description:	Despite advances in tooling and the promise of DevOps, companies often struggle to design and deliver applications on schedule and on budget. One reason is that there are often unplanned, or invisible, activities and bottlenecks that steal development time and slow down their value stream. In this episode of the podcast, Mike Kavis and guest, Tasktop's Dominica DeGrandis, discuss what Dominica calls the Five Thieves of Time that can create bottlenecks, slow down an organization's value stream, and cripple the development process. Dominica lays out her solution to remove bottlenecks and catch the thieves—to help speed up the value stream and deliver better applications to the business.
Duration:	**00:24:18**

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 the Architecting the Cloud Podcast, where we get real about cloud technology. We discuss what's new in the cloud, how to use it, why to use it, when to use it, all that good stuff, and we do it with the people in the field who do the work, day to day. I'm Mike Kavis, your host and chief cloud architect over at Deloitte, and today I'm joined with Dominica DeGrandis, author of, "Making Work Visible," and has the coolest title ever, called Principal Flow Advisor at Tasktop. Tell us about that, that cool title. I like that.

Dominica DeGrandis:
Thanks, Mike, and thanks for having me. Yeah, my title had been, you know, Director of Digital Transformation, but what I really have done all this time, for the last several years is work with organizations to optimize the flow of their work across value streams, helping them make it visible, helping them optimize it, and with an emphasis on measuring that using flow metrics. So, I'm thrilled to have flow in the title, and it's really just about helping our customers optimize the value that's actually moving through there.

Mike Kavis:
Yeah, and you grew up in release engineering, right, if I remember from a talk a couple years ago?

Mike Kavis:
So, obviously flow is pretty relevant when it comes to release engineering. So, let's talk about your book. So, as an author myself, most people write technology books because they experience pain and they have to figure out how to solve that pain, and a lot of times a book comes out of that. I was just wondering if that's what drove you to writing this book called, "Making Flow Visible."

Dominica DeGrandis:
Yeah, well, it started because I was teaching classes on, you know, Kanban for DevOps, or Flow for DevOps, and people said, like, "Where's the book? You know, where's the handouts for this workshop?" And there wasn't any, and so I created it originally so I could hand something out to students in my workshops. And then it sort of grew and grew over time, and I also created it for myself. So, instead of having to go to one of 30 books in my bookcase that included some of this information, I wanted sort of to gather it in all in one place so I could just take one book with me on the road, or you know, one PDF on my laptop and be able to find what I was looking for. You know, what kind of queue was that that Don Reinertsen talks about? You know, things like that.

Mike Kavis:
Yep, pretty cool. So, tell us a little bit about the book. For those who are going to read it, what are we going to get out of that?

Dominica DeGrandis:
Well, it's centered around the five thieves of time that I believe if we could acknowledge, bring some visibility to, and measure, we could really improve how we work and be able to actually get some work done, because most people are so overloaded. There's a complete imbalance between the capability we have to do the work and the amount of demand and requests that come to us. And so the five thieves of time basically are unplanned work, you know, Thief Unplanned Work – this is the foundation of interruptions that cause us to not get what we – our most important work done, because we have a hard time saying no to five-minute requests, and nothing is usually five minutes. And then Thief Conflicting Priorities just come from, Team A's priority is not Team B's priority, yet we have a dependency on Team B. That's a conflicting priority. We spend a lot of time with that, which sort of leads into Thief Invisible or Unknown Dependencies, which we see a lot when we break our organization down into lots of small teams. You know, there's bound to be dependencies between them that cause a lot of angst.

And then Thief Neglected Work – this is the important work that often just doesn't get prioritized because it may not be considered revenue-generating work, things like fixing technical debt, or improving internal team processes that can help us move faster. And then last, but certainly not least, the ringleader of all the other thieves is Too Much Work in Progress, because we have too much work in progress if we have conflicting priorities and we say yes. You know, we have too much work in progress when we only plan for planned work and we don't plan for unplanned work. So, the other thieves all play into us having our plates just overloaded and not being able to get our work done on time, or even during working hours. It's almost like Sunday's the new Monday now – Because Sunday is when we have uninterrupted time to focus long enough on our complex work to actually complete something. It's why people sometimes do their best work at ten to midnight at night because, you know, the kids have gone to bed, and now they've got uninterrupted quiet time where their brains can think deeply about solving the problems they're trying to solve.

Mike Kavis:
Yeah, and you did a lightning talk about saying no, and you mentioned, you know, someone says I've got a minute, and that minute is never a minute. And while you were saying that, I was just thinking a couple Fridays ago I blocked 8:00 to 12:00 because I just had to get stuff done. And I probably got three of those calls and I took them, and next thing you know 12:00 was gone and I didn't work on a single thing that I'd blocked time for. So, yeah, definitely know what you're talking there.

Dominica DeGrandis:
Yeah. Yeah, it's hard, but I like to say if we're going to get the work of our life done, we have to ferociously protect our time. And this is true more for our creative maker people I think than for management. Creative people – designers, developers, writers, we need extended periods of uninterrupted time for that creativity to get into the flow, right, to get into the zone so we can focus and actually get something done. Or maybe if it's not completed, at least it's to the next point where we can sort of tie a bow on it and put it on a shelf if we have to go to a meeting or something.

Mike Kavis:
Yeah, so in your book and a lot of your talks you talk about, you know, the flow of work and making work visible, right, which is the title of your book. So, what are examples of work that's in the value stream that's not visible to us? It's eating up time, but we don't realize it. What are some examples of those?

Dominica DeGrandis:
Yeah. I'd say the first one is unplanned work. It's not in the backlog, right? There's no unplanned work in the backlog, and it doesn't make it into triage or investigate. If we're lucky it gets added to done. You know, that's why we have work, that the cycle of time was maybe one day on it, because it didn't get put into the system until it was actually done, but a lot of times it doesn't get entered at all. Things that we think are only going to take us a few minutes or less than 15 minutes tend to be invisible. And depending on the nature of your demand, that could be a large chunk of work for many people. "Oh, this will just take me 15 minutes," or, "This will just take me 30 minutes. I'm not going to create a work item for this. It's going to take me longer to create a work item than it is to do. I'm just going to do the thing." And that may or may not be a problem, but if you have a lot of those, if you've got four or five of those going on every day, then that's going to consume a lot of your time. So, unplanned work is definitely one of the things that rates high as being invisible.

Another invisible item is feedback. You know, we might have test in our workflow or validate, but what about the feedback? Like, "What do you think about this? You've got five minutes. Can you let me know? Could you take a look at this blog I wrote?" "Could you take a look at this new product canvas we're developing? "Can you please –we need your review on this item today. We thought it'd be okay for it to slide out longer." So, I'd say feedback is also invisible.

I also think we don't have a lot of visibility on dependencies and the impacts from those dependencies, because that work may show up on somebody else's workflow, especially for organizations that don't have good visibility across their whole product value stream. Maybe it's in a different value stream, but it impacts you because of that dependency. So, I think those three right there tend to be invisible.

Mike Kavis:
Yeah, I often see a person, or a group of people, being a dependency because for whatever reason they're the only expert on the topic, so everything goes through them. And those people often have a hundred other things to do, but they're always being sucked into review this, give their feedback of that. How do we get out of that? I see it everywhere. There's always this expert, the poor person works 24/7/365 and becomes a bottleneck for a lot of stuff; nothing can go forward without this person's approval. How do we get out of this?

Dominica DeGrandis:
Yeah, yeah. Well. As far as how to get out of that, I think that creating a skills matrix and having people self-rate themselves on their knowledge and skills and experience is quite helpful. And then you turn it into a heatmap so that it becomes evident on who has this level of expertise in the company and who is learning that. And you use different colors to bring visibility to that so that we can see where we're at risk. Oh okay, Brent is a master at half of the skills that we need to have, and the rest of the team, they either need somebody to pair with to do the work, but they can't – they're not quite in a position yet where they could be an instructor or teacher. They're not a master at it yet. So, I find that to be quite helpful, and those skills that are lacking, where the heatmap is showing lots of orange or whatever colors – yellow, whatever colors you're using, that those skills then become part of a dojo that we're training people on. You know, we see a lot of dojos now where we have technical tools and skills and languages that we want to bring teams up to speed on and different ways of working, but what about offering in the dojo new knowledge and skills that we need more people to have? And so that's one way, is a skills matrix. I think another thing that's kind of a hot topic right now is the fallacy of the full stack engineer. I was so honored to be in a group that wrote a paper on it. There's a whitepaper that just came out in August. It's called, "Full Stack Teams, Not Engineers." And we did some research off of stack overflow, and the number of developers who consider themselves full stack now has grown – in the last six years it's doubled pretty much, up to 60 percent now from 30 percent. And then we started to ask, well, what does the full stack mean these days? Like, what is the full stack now? And when we looked at that, and we looked at the vastness of all the knowledge and skills that people would have to do, we recognized that not really that many people have all that knowledge. Those that do, they're not going to be available, right, because they're probably at Netflix or something.

And so the guidance of the paper is to consider building up that skill set across your team, so you have a full stack team, and DevOps isn't necessarily relying on one person to do it all. We know that leads to a lot of problems. It leads to burnout. It leads to people being completely overwhelmed. It's not that people aren't talented; we've just got so much work in progress and demand that they don't have enough time to think about how to do something really well. You know, when we're rushed to finish our work, we're getting things out that are sort of half-baked, which in some cases that's okay if you just want quick feedback. But if you're trying to finish something and get it out there for a customer, we would hope that the quality of it is of a certain level that is going to be valued and be accepted. So, yeah, the whole full stack engineer thing I think is coming under scrutiny.

Mike Kavis:
One of many unicorns. Myself, I've been in this industry over 30 years, there's no way humanly possible to ever experience enough of the stack to be an expert on all of it. So, I detest that term and I love the team full stack teams, which is what I always preach. You know, you're not going to hire this one genius; you're going to get a bunch of talented people that cover all the domains and collaborate well and get stuff done. So, what are some of the typical barriers for getting rid of this work or finding work that's not visible, and getting rid of it?

Dominica DeGrandis:
What are some of the barriers for finding the invisible work and getting – well, first of all, we're never going to get rid of it. There's always going to be unplanned work and unknowns, and so I think we're delusional if we think we can probably get rid of all our conflicting priorities and the problems. But just making it visible is certainly going to help us start that conversation on what the pain points are. So, that's the place where I start in my workshops is asking people what prevents them from getting their work done. And the outcome of that is a fairly long list, and just about every workshop there's a pattern of common threads there of what prevents people from getting their work done. And people will say interruptions, for example – I get interrupted all the time. Okay, so if that's a big pain point for you, whatever your pain point is – or, you know, we've got too many meetings, or we've got conflicting priorities. Whatever the pain point is, let's work to make that visible. And so that usually starts an experiment. I mean, the answer to the question is we do experiments. We identify an activity. We set how we know if it's going to be successful. What are some of the measures to put around that? And so if the goal of the experiment is to understand why we have so many interruptions, let's do an experiment.

I did this with a company a while ago who complained about so many interruptions. And I said, "Okay, for each interruption, just scribble a few words about the nature of that interruption and let's put it up on this physical board by your desk." And so they started this Kanban, to-do, doing, done, and every time they got interrupted, and they defined what an interruption was. For them they were co-located for the most part, so it was somebody who walked up to their desk or did a shoulder tap. And they also included any kind of – they were using Yammer at the time, but it could be any kind of Slack – "Hey, got a minute?" thing, and they just wrote down what the nature of that interruption was and put it up there. We said, "Well, just try it for a week."

So, this experiment after one week that showed 92 interruptions, most of those interruptions were product managers, or project managers, asking for status. They wanted to know where's my thing, because they had no visibility on where their thing was. And that drove some discussion which resulted in some decisions to insert some more states in their Kanban, in their workflow design, so that those product managers could get visibility on waiting on design, or design done, or waiting on analysis, or analysis done. Not that they had to insert wait states for every single work state in their workflow, but for them, a lot of the work got stuff in working on validate, waiting on validation. And so we inserted a waiting on validation state in their workflow, and now we could start to measure how long do things sit in validate? "Oh my gosh, that's a long time." So, then we worked to improve that. So, does that answer your question?

Mike Kavis:
Yeah, yeah. It reminds me – a long time ago – But there was this place that loved status, right? So, they had a new leader and the leader says, "We're going to be transparent and we're going to create this monthly status." Well, unfortunately there was six levels of management to get down to the person doing the work. So, at the end of every month there's this big rush to bubble up work – first level of management, then the next and the next and the next, and there was all this busywork. And then, like, four weeks later the leader who wanted this would tell us what we already knew a month ago, right? "Here's the status!" We're like, "Well, we gave you that." And it was just all this busywork, and you know, to spit back what we already knew was kind of funny.

Dominica DeGrandis:
Yeah. This is the problem with what I call the Red, Yellow, Green Report. It's obsolete by the time it's printed, and we call it the Watermelon Report because everybody knows it's red on the inside, but it just looks green on the outside. And this is why it's so important to automatically see where the work is at in your tools instead of having to manually create a status report, which is usually opinions. In my opinion, those are opinions on what we think, and often they're written to be politically correct if there's not enough psychologically safety in the organization for people to say what they really feel. But we're using our tools, which we would hope would be the source of the truth for where the work actually sits – well, it's in this wait state, or it's in dev, or it's waiting to be released, whatever – and we make those states visible through dashboards and reports that people can go see where their work actually sits, that, I think, is a great way to get visibility on it.

Mike Kavis:
The Watermelon Report – I love that. I have one place I worked, it was always trending green, which means it's really, really red. I agree with what you said. I worked on a project where, you know, everyone wanted to know status. It's a very important piece of information, but we finally got everyone to say, "Here's all the information in the Kanban board. If you want to know something, go there. Every Friday we're going to do a demo. That's your status. Everything's in there. Risk is in there." And finally, we got away from the traditional waterfall reporting that just consumed everyone's time and got everyone angry anyway. So, yeah, totally agree on that.

So, here's an observation I had from DevOps days. I've been to I think all but one – and I've been talking about this for a while. DevOps to me has really been about removing bottlenecks, right? And when we first started this, it was CICD, right? It was the builds. It was the deployments, the infrastructure. And then we started looking at testing and automating testing. Then it was DevSecOps. And, lately it's been compliance and governance. But I saw that great talk on architecture, and this is where I've been struggling with all this time.

We're not talking about architecture, and finally we are, and I think architecture is a huge bottleneck, especially if it's not done right. And I think what I've seen is the companies that have been doing this for a while, some of these other bottlenecks, they've tackled those, or at least made them less painful. And now what I saw from a few companies is now they're focused on architecture finally, and as an architect, you know, tears are running down my face when I'm hearing this. But how often do you see that a lot of this technical debt or a lot of this crazy process or a lot of this dependency on that one person who knows it all is the result of a weak, or not well thought-out, architecture?

Dominica DeGrandis:
Yeah, we see it all the time, and that's why there's such a drive, I think now, to loosely-coupled systems and microservices, is to break down these pain points in the architecture that inhibits and prevents people from being able to, say, do annual outage releases. Well, we can't do that because of this dependency on the architecture. What I do see now, though, is a heavy dependency and bottleneck on UX design, and I think that's why we're seeing that ratio of UX designers to developers increase.

Mike Kavis:
So, what's the bottleneck?

Dominica DeGrandis:
Well, as applications now, and the way that consumers purchase goods, the design of the website and the app—is so important. I mean, if you go to a site that's badly designed, how secure or comfortable do you feel purchasing something from there? Sometimes I'll bring up a website and it's still got, that bright blue background with, purple letters or something, and it's like, I don't think so, and I'll go look for another.. But often, developers are working on wireframes. “Where's the wireframe?” “Oh, well, it's because our one UX designer or developer has conflicting priorities and is working on all this other stuff.” And so when we look at the ratio of developers to designers, we see that gap getting closers and closer at places like, you know, Atlassian and Microsoft, and in our company, too, at Tasktop. We've hired more UX designers, and now our throughput and our flow time is improving drastically.

And I think the same could be said for architects, too, or at least the way that our architecture is designed. If we give it the attention it deserves and maybe more eyes on it, and more acknowledgment that it – the amount of technical debt that is built up, it inhibits our ability to deliver customer value as fast as we'd like to.

Mike Kavis:
Yeah, that's an interesting topic, especially the UI piece, because UI is becoming voice and motion and AR and VR. We could do a whole podcast on that, but that whole world's changing as well. So, that's about all the time we've got – great conversation. I appreciate your time. So, where can we find some of your content? I know you've got a lot of YouTube out there. You're on Twitter. What's your Twitter handle?

Dominica DeGrandis:
Twitter is @DominicaD, and my content is sort of all over the place. It’s on my list of things to do – I do have a website. It's DDeGrandis.com, DDeGrandis.com. And I used to be pretty good about getting all my content out there. It's one of the things I've got planned for during the slower time towards the end of the year, is getting all my content out there. Most of the latest is going to be up on YouTube, on the Tasktop site, or with IT Revolution. That's where that full stack teams whitepaper is at, is on the IT Revolution website.

Mike Kavis:

Cool. Well, thanks again and I appreciate your time. You can find more podcasts by me and my colleague Dave Linthicum by searching for Deloitte On Cloud Podcast on iTunes or wherever you get your podcasts. I'm your host Mike Kavis. If you'd like to contact me directly, you can reach me at MKavis@deloitte.com. Thanks for listening and we'll see you next time at 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