Architecting the Cloud, part of the On Cloud Podcast

Mike Kavis, Managing Director, Deloitte Consulting LLP

Title: **Reduce IT complexity and build better apps with AIOps**

Description: AIOps is being embraced by savvy companies to help them build—and operate—better apps, faster. In this podcast, Mike Kavis and Constellation Research's Andy Thurai discuss how companies can use AIOps to reduce IT complexity and deliver better products. Andy cautions that the onus is on the enterprise to assuage employees' fears and get management buy-in for a successful AIOps implementation.

Duration: **00:33:03**

Operator:
This podcast is produced by Deloitte. The views and opinions expressed by podcast speakers and guests are solely their own and do not reflect the opinions of Deloitte. This podcast provides general information only and is not intended to constitute advice or services of any kind. For additional information about Deloitte, go to Deloitte.com/about. 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, and welcome back to the Architecting the Cloud Podcast, where we get real about cloud technology. We discuss what's hot in the cloud, but most importantly with the people in the field that do the work each day. I'm your host Mike Kavis, chief cloud architect over at Deloitte. And today I am

joined with Andy Thurai, VP and principal analyst at Constellation Research. So, before we started this, Andy tried to give me the right pronunciation, but my Greek tongue don't roll that way. So, tell us what your last name really is.

Andy Thurai:
Well I mean, it's a little bit harder with the phonetic sound. The actual pronunciation is more like Thurai, Thurai, Thurai with the TH sound. But I mean, Thurai is an acceptable form. I have no problem with that.

Mike Kavis:
Well, we'll roll with that. *[Laughter]* I didn't even try to attempt it. My tongue just don't move like that. So, we've known each other for a long time, and—well, first tell us about, you've kind of got a new gig. Tell us about that. But we're going to focus on AI/ML type—AIOps today. But tell us a little bit about your background and what you're doing, what you're up to now.

Andy Thurai:
Yeah, so the funny part is, which I didn't realize for the longest time, if you notice my last name, that's Thurai with an AI. *[Laughter]* So–

Mike Kavis:
I'm going to change it to Thurai-ML

Andy Thurai:
There you go. *[Laughter]* So, I'm a VP and principle analyst with Constellation Research, a research firm particularly specialized in covering the AI and ML areas. So, my work involves two facets of it, as we talked about earlier. One is how do you use AI or ML to improve the operational efficiencies, the IT operational efficiencies, whether it's AIOps, CloudOps, observability, all of those areas. That's one side of it. And the second side is to—the first one obviously appeals to the IT operations folks, IT executives those kinds of folks. The second one is more of how do you use AI and ML to improve the—any of the business work and applications, right, whether it's computer vision or how to use tech to generally approve whatever app you have. So, that's—kind of it's the same thing, but it's two different things if you look at it. So, I over both of those areas, and I'm working on some of those reports and I'd love to chat with you at some time in how we can work together.

Mike Kavis:
Sure. And speaking of those reports, I've noticed a few blog articles that took a lot of action on my network recently so, we're going to talk about some of those. So, you recently wrote one called, "The Number One Reason AI/ML Projects Fail in Your Organization," so—and you mention things in there about the amount of data they have, the amount of—even though they have a lot of data, they don't have some of the key data that they need, and all the complexities like—I kind of joke. Are we entering MDM projects again? For years and years and years we were like, "We need this master data management," and I don't know if any of those projects ever worked, trying to get all their data into a common nomenclature and common place and cleaned up. But regardless, I said a lot. Tell us about the number one reason why you're seeing AI/ML projects fail.

Andy Thurai:
First of all, thanks for having me on the podcast, and I see that if I become popular in your feed, then I get calls. So, I'll make sure that I'll tag you all the time. *[Laughter]* So, that's an interesting point, that you picked that article. This is—again I'm sure you have the same thing when you talk to some of the enterprise-level executives, right? A lot of companies think their AI/ML projects fail maybe because they don't have enough skilled folks who could do that, not enough data scientists, or they're not picking a proper algorithm, or they're not using the right cloud. But my personal view is a lot of those projects, majority of them fail because they don't have the proper data or they don't know how to use the proper data. That's why I wrote that article. I mean, there's a series of articles I wrote, a few of them. If you go to my website, TheFieldCTO.com, you could look at that as well.

But look, at the end of the day, you and I both agree that the existing entire IT infrastructure, the whole thing IT—when I say infrastructure I'm not talking about just the database, and compute, and servers, and whatnot; I'm talking about the whole, all the way up to app level. It's all built modeling for a compute economy. That's what we have from '70s, '80, '90s, and 2000s. But now things have changed to a data economy. You are trying to make meaning out of a data economy and you are trying to move as fast as you can to make decisions. The entire thing of whatever you want to call it—the pipeline of application development—is built geared towards the compute economy, not towards the data economy.

That's the thing that I was trying to explain. And obviously if you double-click on that, the enterprise client, when you go and talk with them, "I have a ton of data." And when you deep-dive into that, when you start doing analysis, you'll find out either they don't have the right data, or they have data blank spots. I mean, if you are trying to achieve, for example, total observability, if you monitor and measure 80 percent of your—especially when you have a visibility application, 80 percent of your application you measure properly; the rest 20 percent you don't. How are you going to get a total visibility? And that's just one example. That same thing goes across the board. So, your data collection is a problem. Your data management and managing the blank spots is a problem. And more importantly, with the distributed applications, with the cloud-native applications—I'm sure you would appreciate this—data instrumentation is also a problem. That's another whole topic—I could talk the whole day.

It's—one thing I will point out, though, having noticed that digital-native companies don't have this problem, it's a majority the legacy companies, or the so-called, that I want to be hybrid companies have this problem. Born digital-native, cloud-native has none of these problems. They don't have hybrid issues. They are not bogged down with other issues. Their business generally is—again, I'm generalizing here too much. But they are set to solve one business problem. I mean, Uber, if you look at it, they are solving one business problem very specifically, right? Everything is built around it. Their KPI is I want to get this guy, Mike or Andy—he's standing at this block here in New York. Figure out whenever the hell you want to do. Get him his ride as soon as possible, at the price that works out well for both of us, right? They hone in and focus and all hands on deck to solve it.

But with the traffic enterprises, like, they are all over the place. What am I trying to do? You were talking about it earlier. They have like 40 different KPIs with 40 different teams. You don't even know which one you're going to solve first? *[Laughter]*

Mike Kavis:

Well, the cloud-native guys have their problems as well, so don't be fooled. And some of that, we were talking about this earlier, they take an approach: let's build enough just in time. And that works great until they reach a level of success, and then just enough isn't enough, right? So, what happens a lot is we start mandating this architecture, and lo and behold, ten years in, all of a sudden we're legacy, and it's like, "How did this happen? How did we just become like everyone else?" Well, everyone else started someplace, too, right? I mean, scale, and scale, you're a victim of your own success if you're not pragmatic about how you scale, so—and then you have business priorities versus technology priorities. Regardless of what kind of company, you still have those problems.

Andy Thurai:
Right. But here's the thing, though. Yes, I agree that digital-native or cloud-native companies can also have this problem. But they don't hit the problem until they become truly hyperscale. When you say hyperscale, especially with your point about building just enough architecture just in time to solve that issue, you might not face the problem if you have maybe a million users. For digital-native users, a million users is nothing, right, or a million requests a minute or so is nothing. If you go into a hyperscale of having—serving, I don't know, five million customers at the same time, that kind of scale, when it comes in, that's when the problem is.

But also, one of the things you ought to keep on mind—all those digital-native companies, they are kind of born out of necessity, that they are trying to solve a problem where someone else has failed. In the case of Uber, taxi company failed, so I'm trying to give you a new way to facilitate doing these things. I mean, the same thing existed before. I'm trying to solve the business problem in a different way, in a digital-native way, a better way. So, if they also fail, why would you have to use them? I could be using the same thing, what I was using before. Why would I want to go to you, right? *[Laughter]*

Mike Kavis:
Right, right.

Andy Thurai:
So, they have to—it's a question of survival. So, they've got to figure out a way to fix it right away. So, yes, they run into problems, but they fix it fairly quick.

Mike Kavis:
So, another reason I see, whether it's AI, ML, or name your technology—data's always a big one, but people and process, right? People are built structured around the way we used to do things, and now something new comes. So, what are some of those problems you're seeing as some of the larger companies, not the unicorns, right, are trying to embrace AI and ML? What are some of those barriers you're running into?

Andy Thurai:
Oh, boy. How long do you have for this podcast? *[Laughter]* Well, so a ton of them, right? And the top few that come to mind—I would say the first one would be sort of—how shall I put that? A cultural shift maybe is the right word to us. First of all, look, everybody's afraid they might lose their job. If I'm going to use AIOps, observability, or AI and ML to solve something, then I'm going to become outdated, the machines will take over, and I'm going to lose my job. The job security, the mindset—that becomes the number one issue for a lot of these folks to embrace this technology, especially the old school guys who have been there for a while, particularly in the IT Ops area, right? They have been there for a while. It's not like you've got a new group of guys to solve this issue. They have been there for a while, and then if you try to implement these solutions, there's always this friction saying that, "Hey, you know what? I'm going to lose my job? What are you going to do about it?"

So, one thing—also the onus is on the enterprise as well. They need to make it clear, whatever we're bringing in is to assist, not to take over your job, right? Because right now, look, I mean, you've been to some of the war rooms that are held (Inaudible) stuff like that, particularly with the multi-cloud things. How long do war rooms take? And how many people show up for the war room? I mean when you have a collaborative war room—not a physical war room anymore. You'll have a Slack channel or whatnot. I mean, random people get invited. Seventy people get invited to a Slack channel. It goes on for two weeks. You're trying to figure out what the hell are we going to do. And when you have a distributed architecture across the place, you don't even know where to start, what to fix, right? So, if you can show to the organization that war rooms, instead of taking like a week or ten days of 70 people involved, you can bring it down to, you know what? If something happens, I'll tell you either before or during the time it happens, right?

If you do that, then you will have the secondary issue that I see, secondary issue—or second issue, rather, most people have a problem with, which is the executive buy-in. The executives at time don't get it. But if you can show them the business problem—look, you get 70 guys gone for two weeks, not doing anything else, solving this one specific issue, which might repeat itself at a later time or some other issue with come up, but I'll show you a way—again, unicorn status, I'm saying that, "Okay, I'll solve it within a minute," which may or may not be possible. But if you could show them, "I can reduce about 90 percent of the time taken to solve"—and particularly one of the use cases, as you well know, is noise reduction, right?

Mike Kavis:
Right.

Andy Thurai:
With distributed systems, you produce a ton of noise, the noise to signal ratio's too much. If you show them, "I can eliminate 98, 99 percent of the noise effectively, your signals pick up very quickly, and then I can help you solve," that's another thing that'll help the organizations, having executive buy-in. And the other thing which I see which is kind of important is—how shall I put that—more like a skills shortage kind of thing, meaning so you have this—so you have the ITOps and the ITOps group area. Let's talk about the operational side of things. It's the kind of a combination of a data scientist, data analyst, and a business analyst combination—need to look at all the data and figure out—basically you need to experiment, which means you should have all the data to experiment to come up with a result.

Same with any of the business applications, too, right? And in addition to that, they should also have operational skills, plus after they experimented, they made something to work. That needs to be productionized, which means you need to have data engineers. Having all of them to solve your problem, whether it's for the operational side of things, or even for the business side of things, skill shortage is another major issue. There's a reason why the employment level is so high or shortage of people, right people in that industry skill, and then upskilling IT Ops folks.

But all of this said, the two issues that—one of the issues we already talked about and we touched on earlier. One is a data issue, right, which also needs to be solved, plus the existing IT processes, because you are doing certain things in a certain way, but it may not work when you go into the new one. So, I mean, there—I'm pulling apart for you things I do, but again, like I said, that's a huge topic. I don't think we'll do justice by just answering one question.

Mike Kavis:
Yeah. So, there's this presentation I did for a client a while ago. We were talking about observability and chaos engineering, and I was kind of explaining the why, which is—that same why could work for AI/ML. And I was talking about two things really, the complexity of our architectures now, where everything's a distributed architecture. There's so much data coming from different machines. The machines come and go. And then we're going to microservice-level components. So—and I used a graphic which I saw in one of your presentations. I think it was a Netflix chart–

Andy Thurai:
Oh, yeah, yeah, yeah.

Mike Kavis:
And I had a few of those, one from Netflix, one from Twitter, and one from Facebook. And it just showed a visual of what was going on in the environment, and my point was human beings can't do this.

Andy Thurai:
Absolutely.

Mike Kavis:
It doesn't mean we don't need human beings, but the humans need assistance from technology, from AI, from ML. So, your job's not going away; it's just changing. And there's no way in the world that you can look at that monitor and come up with what I need to do next. So, it's not it's going away. Your job's changing, and it needs to be supplemented with a lot of information. And if you don't use these tools—I've seen this a lot, where companies move their dot-com or some key service to their cloud and their SLAs go way down. And people say, "Yeah, the cloud's no good." No, the cloud's pretty good. The problem is you brought your old tools and your old methods to a highly-distributed, complex world and it doesn't work.

Andy Thurai:
That's exactly right, that point which you made. By the way, I will take no credit for that diagram. It's not mine. It's obviously just—I used it from Medium. But in any case, you're right. That picture is, what, outdated by at least four years? I think they have doubled their services, but it gives you a visual picture, and Netflix and Uber are the top one or two or top five, I would say, when it comes to number of services. And the last check in that diagram—that's four years ago—they had about pretty close to 800 microservices, right? Imagine—and again, they're all interconnected. If some service goes down, everything else goes down. And then—so I was listening to this presentation, and they were suggesting—which also I talked about in my article—on an average they do thousands of changes daily, daily, which means some services may not be affected, but some of them would have about three to four, or even 20 changes per day. That's the whole idea of agile development.

So, if you change a specific service five or seven times a day, how the hell are your operations guys supposed to know what changed and what's affecting it? One change might break—I mean, it might work when you do it in unit testing. In and of itself, it might work. But if it's connecting to about 30 or 40 other services, maybe there's some combination that breaks. And then guess what? All hell breaks loose if that were to happen. That’s the problem in itself. The agile development environment, fixing things as you go in the smallest possible microservice, made the life of the faster time to market of the functionalities and services, and made the development guy's life a lot easier, made a mess out of operations guys. That's what one executive-level VP told me who shall remain unnamed. But he said, "They are making our life miserable," which is true.

Mike Kavis:
Yeah. I mean, complexity kills, right? And when you're balancing complexity with speed—and I think I wrote four years ago that agility is the new goal, right? I mean, you could be sitting there for 30 years on top of the market, but some no-name comes in and is delivering value daily that takes you three, four, six months a year, they're going to chip—and they don't have to take your whole business. They just need to take that service away from you, right?

Andy Thurai:
Yeah, yeah.

Mike Kavis:
And some other competitor could be taking another service away from you, and over time your value goes down. So, it's the world we live in, where the expectation is that value is delivered quickly, so there's a tradeoff. So—and this—this is going to turn away from the AI/ML conversation, but a lot of that—the reason why it's taken us long to deliver for so long is we have so much control over things like security, governance, compliance. And the reason why these companies deliver 50 times a day is because they've built software-defined infrastructure, or that stuff's baked in. So, the developer doesn't have any say in the plumbing anymore. The plumbing's delivered as a service-by-platform team that's been blessed by a security team, so we don't need 28 meeting reviews.

We know if you're using this API it's good. And I'm deploying not a monolithic app. I'm deploying sometimes a config file, a config change sometimes, a microservice. So, it's a different game, the vast—there's pros and cons of everything, like you put a drug into your system that cures this but it creates that over there, right? There's a side effect. The side effect here is complexity to do distributed systems allows you to get stuff out faster, but on the receiving side of it I can't get my arms around this stuff. I need artificial intelligence, machine learning, and I need to discover it early, that it's trending bad. I can't wait till it's bad and the house of cards comes down. I need to say, "This trend looks bad; let's go do something with it." So, the people worrying about their job, it's like, no, you've got more to do than you've ever had. It's just different.

Andy Thurai:

So, that's an excellent point, right? So, particularly the point about fail fast, right? The digital economy is about fix as small as you can and fail fast, but the forgotten aspect of that is fix fast. That's where the problem is. If you fail fast, you should be prepared to fix fast. You should know what went wrong, why it went wrong, fix as fast as you failed. But if you failed fast but if you can't fix fast, you're screwed. *[Laughter]*

Mike Kavis:
And that's what I'm seeing. When I mentioned they migrate their dot-com to the website and they build fast and deploy fast but they never brought in the Ops team, and they don't even know what's built, and it fails, and it's like, "Uh, I don't even know what to do," right? So, yeah, we see a lot of that. So–

Andy Thurai:
On that note, if I may, this is where some of the AIOps and applying AI and ML to the Ops efficiency teams are helping the teams rather than hurting them, right? Because you have read, I'm guessing, the *Visible Ops Handbook* by Gene Kim, right, the famous book?

Mike Kavis:
Yeah.

Andy Thurai:
So, one of the points he makes in there, which also I quote in my article, about 80 percent of the unplanned outages happen because of changes, the fail-fast culture, fast-food culture, right? So, if you have—you could call it AIOps; you could call it whatever you want; I don't care. But if you have a mechanism that, when something goes wrong, if your system is able to figure it out, saying that, "Hey, you know what? This particularly thing was working until two hours ago; this changed, went through that," assuming you integrated with the CI/CD pipeline, the whole nine yards, right? "This went into effect about 30 minutes ago, and then after that we have started seeing these things.

Therefore, there's a possibility this is the one that caused it." And then if there's a way to hit a rollback saying that, “You know what? Why don't I roll that back and see if that problem goes away?” And that was easy in the old days because, as you said, the releases were six months, three months apart. If you do a release, the whole application can be rolled back. In the microservices, technically it should be easier to roll back as long as you know what you broke. If you have thousands of changes, if you don't know what you broke, how are you going to fix it?

Mike Kavis:
Yeah, the rollback in the new world is, I just destroy and bring back the old. The rollback in the old world was, “Oh, God, I've got to touch this box, pull out what I put in.” And then you break 50 more things. That's why we only want to deploy twice a year, because it's so painful.

Andy Thurai:
There you go. But again, the Blue-Green and Canary environments are supposed to fix that if implemented properly as well.

Mike Kavis:
Yep. So, let's move on to another article where you wrote—looking at my notes here—that, "AIOps has a data (Ops) problem." So, tell us about that theme.

Andy Thurai:
So, you are picking apart all my articles, aren't you? *[Laughter]*

Mike Kavis:
Well, to come up with questions.

Andy Thurai:
All right.

Mike Kavis:
I go, "What are these guys talking about? All right, that's what we're going to talk about."

Andy Thurai:
Well, so, we talked about some of this earlier, that the data, or the data pipeline, that we have now is built for the compute economy, not for the data economy. There in itself is the problem. Matter of fact, actually your own survey, your Deloitte survey—I think I refer to that in the article—that you guys did, it's a LOB Insight survey, one of those things you guys do with your lab. The conversation with 2,700 LOB owners, there's no clear winner, but the clearest number one priority, what'd they have it listed? You should know this. It's “Fix the data, data pipeline for the AI and ML,” because every single one of them, they've figured out that's the problem, right? Basically, what I think should be done is more of having a DevOps equalizer for the data pipeline of AI and ML, right? So, I mean, there are multiple issues in there. I don't know if we can cover all of it, but I'll tell you a couple of them that I'm seeing as a problem, right?

The first one is particularly when you build microservices, especially when you build them all containerized, right, when you drop containers, instrumentation becomes a bigger issue. Right now, there's a lot of friction between the developers who want to use what they feel comfortable in building for collecting the metrics and data and whatnot, mostly opensource, like Prometheus and whatnot or Fluently and what have you. And the Ops teams, especially the centralized Ops teams like you were talking about, the platform teams, they have their governance and monitoring tools, the whole nine yards, et cetera.

And shockingly, if you look at them, the ones they use are the old, so-called application monitoring tools, which collects the metric from a very high level, whether it's RUM tools real-user monitoring tools, or the DEM tools or application or monitoring. Okay, so it tells you that something broke, but if I—going back to the Netflix example, if I have 700 microservices, one of my whatever the app, the business app broke, and if it touches about 30 or 40 microservices. How am I going to know what broke? So, that data collection, that will help me to lead to that level that is very important, so which means the instrumentation, in my mind, is number one. And organizations need to figure out how to efficiently instrument your—whatever is the new service you're

deploying, right? I mean, Prometheus has become a de facto standard, as we all know, for cloud-native, right? A lot of observability, AIOps systems, still can't handle it. I–

Mike Kavis:
So, are you talking about instrumentation? You talking about instrumenting your code, to leave like cookie crumbs in your code to be able to trace where it's going and those types of things? Or are you talking about instrumentation more around the infrastructure? Or are you talking both?

Andy Thurai:
Kind of both. For example, if you were to collect metrics and other information of (Inaudible) you need Prometheus. And then if you need logging into that, cloud-native logging, then you've got to use a different tool for logging. Or if you need to do tracing from an operational level, then you've got to use a different set of tools. But there are some solutions that don' require all three of them. They'll figure out a way to tell you. But regardless of what, you need to have a mechanism in place that the system has to tell you if something were to happen. You should be able to pinpoint out, not only to the level of which microservice it is, which API it is, or even which line of code if possible, right—imagine the power of that.

If something breaks, as soon as you know what went wrong, within a matter of seconds not only are you alerting system—whatever you have, whether it's Pager Duty or Genie, or Slack, whatever, right? Not only will it tell you something broke, but the problem is in this particular line of code, somebody put in. And it doesn't alert 200 people, as it would normally do. It would just send it to the guy or girl who's responsible for fixing it. So, if it comes to me, I'm the subject matter expert. If I can fix it before even anyone finds it, imagine the power of that, right? That's not taking the jobs away, but it's enabling the right person to fix that, right?

Mike Kavis:
Well, imagine alerting the person before it's broken, so there's–

Andy Thurai:
Yeah, right? Ultimate nirvana.

Mike Kavis:
So, we're seeing consumption of way more CPU on this service than before. Danger—go check it out. I mean, I think that's where this space is trying to go.

Andy Thurai:
Well, so the CPU—I want to make a quick comment on that, too. It shouldn't just tell you, "This is using a lot more CPU." It should also figure out, based on seasonality, or a daily trend as well, right? For example, if it's Black Friday or the Amazon Prime Day now, because of that, it shouldn't come and complain that, oh, it's too much CPU usage. Well, you're expected to have that usage, right? There has to be a baseline measure and trend analysis. So, the AI/ML, where it shines is it doesn't just complain that there's too much usage. It complains there's—one of the things that somebody told me stuck to me. Not all anomalies are incidents, right? An anomaly can happen. It's outside of the range, which is fine, but you need to figure out is that really an incident. And if it's an incident, is it causing problems to anybody? That AI/ML system, if it's set up properly, should be able to figure it out and tell you. And if you can figure that out, if there is a problem, then you should fix the problem. If it's not, fine. Okay, you use 80 percent of my CPU. That's fine. Keep using it. I'm fine with that, right?

Mike Kavis:
Yep. Yeah, well, and the other point there is it can say, "I'm seeing behavior that looks very similar to a behavior we saw three months ago when this thing occurred. It may not be an incident, but maybe you should go look at it." You know, those types of things.

Andy Thurai:
Well, not only go and look at it. How about this? "Three months ago, this happened and you did this. It fixed that. So, take a look at it. And also, by the way, someone else changed this one, which also leads to that." So, when you put all of it together—I mean, right now you'd be surprised, the Ops teams, how many systems they need to log in to get a complete view. I wrote a piece on that as well. There are three to four systems you need to log in to figure out if there is a problem, or what the problem is. And then you've got to go into one or two systems to figure out what the fix is. And then after you figure the whole thing out, assuming you're not getting into a war room or—you've got to log in to another system that'll automate that. But imagine if you had a system that not only figures out something went wrong.

There is a precedent that this—last time this is what happened and you did this to fix it, and then if I'm having dinner or sleeping, and it wakes me up and says that, "Hey all this happened, but by doing this, it might fix. Do you want me to run out a code?" If I look at it, I would say 99.9 percent of the time I'll click yes to see what happens, right? And it'll fix it for you. I mean, if it's a low-risk environment, it depends on—you also have to do a risk analysis of whether or not you'd want to auto it. The system probably would be able to fix it. I mean, the least of it is assigning additional memory or CPU or whatnot, but there are other things that auto system can recommend or suggest or fix it for you.

Mike Kavis:
Yep. So, we're about out of time. We could do a whole other podcast just on that topic.

Andy Thurai:
Which we should. *[Laughter]*

Mike Kavis:
Which we should, yeah. But—so where can we find—you mentioned a bunch of articles you wrote. You know, you blog a lot. Where can we find this content? Where can we find you on Twitter?

Andy Thurai:
Sure. So, you can follow me on my Twitter handle, @AndyThurai. That's Thurai with an AI.

Mike Kavis:
Of course. That's your new tagline, right?

Andy Thurai:
Yeah, right? Or you can connect with me on LinkedIn, and also I blog a lot, I write a lot, and talk a lot on my website. That's TheFieldCTO.com. Again, that's TheFieldCTO.com.

Mike Kavis:
All right, well, appreciate your time today. That's it for today's episode with Andy Thurai. To learn more about Deloitte or read today's show notes, head over to www.DeloittePodcast.com (This link doesn't work) where you can find more podcasts by me and my colleague David Linthicum just by searching for Deloitte On Cloud Podcast on iTunes or wherever you get your podcasts. You can always reach me, Mike Kavis, at MKavis@Deloitte.com or @MadGreek65. We appreciate your time 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.