accenture

EPISODE 1: SERVERLESS COMPUTING

PODCAST TRANSCRIPT

Barbara: 00:00:00 – Welcome to the first episode of Trending Topics in IT: A Deep Dive into Today's Emerging Technologies, a new podcast series on cutting edge technology developments and sponsored by Accenture and AWS. I'm Barbara Call, Senior Director of Content Operations and Strategy with IDG. I'm joined today by Miha Kralj, Managing Director, Cloud Architecture at Accenture, and Mike Deck, Principle Solutions Architect, Serverless Partners at AWS. Welcome, gentlemen.

Miha: 00:00:32 – It's an honor to be here, Barbara.

Mike: 00:00:34 – Yeah, very glad to be here today.

Barbara: 00:00:36 – Great. So today we're going to talk about serverless computing and how it's different from today's server-based architecture and the associated business benefits. So I want to start with defining serverless computing. If each of you could take a minute or so to clarify what's meant exactly by serverless computing and how is it different from today's server-based architectures?

Miha: 00:01:00 – All right. Let me try to take the first cut here. I would say serverless is a cloud-inspired execution model where provisioning and operations of that required infrastructure is fully and completely automated. So despite its name, serverless of course still runs on servers. It has the virtualized, the containerized service underneath it, but the core point is that people don't interact with those servers anymore. All of those infrastructure tasks, like provisioning and scaling and cleaning up, all of that is done by machines in a completely automated lifecycle. That's very much the core of the serverless underpinning.

Mike: 00:01:40 – Yeah, I think that's a great kind of summation of what serverless really is. When we at AWS talk about serverless, we really look at kind of three core characteristics to what makes a service serverless. Those would be no servers to manage, as Miha kind of pointed out this idea of that there's no operating system to install or patch or anything like that. Also the scaling aspect of it, the fact that scale is managed for you or is done in a way that's defined in terms of the actual capacity of the application as opposed to trying to think about things like CPUs and memory and these kind of server-based concepts. And then last would be sort of automated high availability. So all of the serverless kind of components of the overall serverless platform that we have, have built-in high availability that's not something you need to design or really think about in your applications. You just sort of get that out-of-the-box. And so I think those three kind of components are really kind of the core defining aspects of what makes something serverless.

Miha: 00:02:42 – Let me also help here. I often hear interchangeably used serverless and FaaS or Functions-as-a-Service. That is just partially correct. At least when we are looking into the complete solution design and that's I think quite relevant for people to understand. Serverless requires FaaS as one of its components. It requires that computing part to have a solution, but serverless then also requires persistence, it requires management of API endpoints, and so on. So serverless is slightly larger and we'll talk

about that a little bit later, but FaaS is a crucial component for serverless, but it's not the only part.

Barbara: Okay.

Mike: 00:03:25 – Yep, that's a great point.

Barbara: 00:03:25 – Yeah, that's good. Okay. Thank you for that definition. I think that's really helpful. So the other thing I've learned about serverless computing is the fact that its event-based approach to computing and this enables organizations to be more responsive in faster ways. Could each of you talk about that and explain sort of why that's the case?

Miha: 00:03:46 – Sure. So Event-Driven Architecture, EDA, is nothing really new in the computing world and we tried to do it for quite a while, I would say from the 2000s or kind of somewhere around there. One of the core issues of true event-driven was mostly, how do we capture events, how do we react to those events. What serverless brought to the market is much – we usually talk about reactive event-driven architecture. So what serverless enabled us is that new architectural styles. So serverless, as I said before, it's an execution platform that finally enables the architectural style where we can completely change the way how we are looking into solutions, number of units that that solution needs to have, and then complexity of interactions between those units.

00:04:40 – It also drives, with all of that event-driven approach, every little component is extremely independent and very, very well decoupled. So speeds to deployment in serverless is way faster than more tightly coupled systems that we did before, but also costs to deploy of those event-driven independent units in serverless is going dramatically down. So that whole combination that you can listen to events and react to those events once they happen and you can do that in that very elastic, very scalable manner. It really allows developers to stop thinking about how is my system actually working and it allow them to create the reactive style of applications where all you really needed to do is to consider what do I need to wait for, what events am I listening for, and how do I react into that.

Mike: 00:05:34 – Yeah, absolutely. I think that's really kind of the core of it. And one of the really kind of key benefits that I think that this general event-driven nature provides when you're working within the context of a serverless architecture is this idea of eliminating waste from an actual infrastructure cost perspective. So if you think about kind of the majority of the time spent by most of the servers that are out there in the world and in any datacenter, most of them are spending a lot of their time sitting idle listening on a port waiting for a request to come in or waiting for an event to happen. And whether your server is active or it's just sitting there idle, you're still having to pay for it in one way or another, whether you own the physical hardware yourself or if you're paying for a virtualized instance in some sort of cloud platform.

00:06:26 – With serverless that whole model is kind of turned on its head to where now you literally are only paying for the compute time that you're using. So if your application goes through periods of low utilization and periods of high utilization, as a developer you only need to worry about how do I handle each individual event and then your infrastructure, your actual costs to run that application only accrue when the code itself is running in response to those events. And so I think that definitely kind of providing this mechanism for developers to think about these very highly cohesive individual components just write the code in terms of, how do I handle each one of these individual events and kind of how does data flow through that system. And then giving them the ability to deploy into a platform where now all of the scaling and elasticity concerns and kind of cost optimization aspects of it are completely managed for them, just provides a much richer kind of environment to be able to quickly develop very highly optimized systems.

Barbara: 00:07:31 – Okay, great. It's really helpful. So those are a lot of the benefits we're talking about sort of down in the trenches and what the developers are doing. If we take a step back and maybe take it a step higher, walk me through the various business benefits that are associated with this kind of model. I understand agility to innovate and obviously there's some cost reduction, but tell me more about the business benefits.

Miha: 00:07:37 – Well, we track about 25 different benefits and a value of serverless, which we usually then try to, through the value realization try to track are clients getting the value out from it. I won't list all 25. Let me give you just the top 4 that I think that are super crucial to understand.

00:08:16 – First one you already explained, it's the lower execution cost. Very much the serverless, the way how AWS is measuring it, it's called the gigabyte seconds, so you're kind of trying to pay for amount of RAM that you are getting within seconds of execution time, which is a completely new way of thinking about the cost of execution.

00:08:37 – Second one is completely automated scalability. That's extremely valuable component when you are thinking about that just-in-time provisioning. Before serverless you always needed to put some automation together to automatically scaling from let's say 0 to 50,000 users concurrent when they're asking to get to your website.

00:09:01 – And the third one would be that there is no infrastructure skills required anymore. If there is a patch for your operating system, you don't even get an alert in a serverless environment. You don't need to think about the networking and you don't need to think about addressing and routing and VM sizing and all of the things that we kind of always took as for granted that you needed to do that kind of the ground work foundations even before you put code on it.

00:09:29 – And the last one, which I believe is the most important is easy access to cloud innovations. So serverless, for example, Lambda on AWS is the most ideal platform that somebody can very quickly experiment with all of those other complex, modern cloud services, that all the software innovation is now launching in a cloud first. I don't know when was the last time that I got the DVD and somebody told me this is the latest software innovation coming from some software company. And if you want to try that latest innovation, serverless is ideally positioned because you can so quickly put those proof of concepts and minimum viable products together.

Mike: 00:10:09 – Yeah, absolutely. I mean I think Miha really kind of stole my thunder there, I feel like in terms of those are very much the core value props that we've seen sort of time and time again across a number of different customers. The way that we talk about it a lot is kind of removing this concept of undifferentiated heavy lifting. So again this idea that you don't have to have this infrastructure expertise and you don't have to spend engineering cycles thinking about how are we going to automate patching, how are we going to automate provisioning, and how are we going to keep an eye on how many servers we have and think about capacity planning and all these kind of things. When you sort of remove all of that extra overhead out of the process of building and deploying and operating systems, you give your engineering organization a lot of cycles back to really now start innovating on behalf of your customers and thinking about ways that they can deliver differentiated business value instead of the very undifferentiated types of innovation that historically, I think IT organizations have had to focus a lot on.

Barbara: 00:11:14 – Excellent, okay. Before we continue our conversation with Miha and Mike, I want to say a few words about our sponsors, Trending Topics in IT: A Deep Dive into Today's Emerging Technology, reports on emerging enterprise technologies and is presented by cio.com in partnership with Accenture and AWS.

Now back to the show.

00:11:38 – So welcome back. This has been great. So one of the most complete and production-ready serverless platforms today is the AWS serverless platform. What can you tell us about the platform and how it works? And maybe we'll start with Mike this time.

Mike: 00:11:53 – Sure. So the AWS serverless platform is composed of a number of different kind of components. So Miha made the great point earlier that a lot of times people kind of conflate this idea of Functions-as-a-Service or sort of serverless compute, which is the AWS Lambda service in the AWS world, with kind of serverless in general. But in fact there's a number of other components that go along with that, so things like responsive data sources. So NoSQL databases like Amazon DynamoDB, as well as the new Serverless Aurora, which is a relational database that allows you to run in a serverless capacity that we announced, it reinvent, and will be coming out shortly. As well as orchestration and state management tools like the Step Functions service that we have that allows you to orchestrate in multiple Lambda Functions together and manage sort of how state moves across those different functions. As well as application management integration components like API Gateway for managing HTTP connections and some of our messaging services like SQS and SNS, as well as Kinesis.

00:13:03 – So again I think the key to it is that there's all of these various different concerns that you as the developer have when you're building an application. And I think the serverless platform is really about providing these very feature-rich manage components for each one of those different concerns that can then be sort of glued together, if you will, with Lambda Functions and with this kind of custom business logic layer that executes on top of the serverless platform.

Miha: 00:13:34 – Perfect. On top of that, what Mike said, we kind of when we say serverless platform, let me first try to be vendor-neutral and then deep dive into AWS. For a platform, we see that you have to have six blocks of functionality that executes really well so developers can actually take their codey[sic] thing and put it onto that cloudy thing.

00:13:57 – The first one is the compute part, which obviously with Lambda it's serving as a FaaS in AWS. Then we have persistence, which Mike already mentioned, and AWS offers amazing set of fantastic persistency options from NoSQL to kind of a relational with Aurora to caching to all of the other options that persistence needs to have.

00:14:21 – Third one is the API with API Gateway or anything that will incur together the HTTP calls. Fourth one is monitoring, which is super important in case of a serverless platform because in a decoupled world you need to monitor all of those different events and interactions and reactions in order that you properly track and debug application. The strength of the monitoring is super important as well.

00:14:48 – Then the fifth one is security, which of course people are thinking what does that do to security and risk posture if you are decoupling as a serverless? And extremely granular and transparent security model of AWS actually takes all of those worries away. And the sixth one is integration. We rarely see 100% pure serverless solutions, so it's extremely important to integrate into something that, I dare to call maybe a hybrid serverless type; that you can use a little bit of a traditional compute and traditional application model, some architectures, with a lot of a serverless on top.

00:15:27 – So just to recap; compute, persistence, API, monitoring, security, and integration, all of that together. AWS offers extremely mature, well-tested, battle-tested type of services that it's very easy to stitch together that you can actually get a very, very well performing serverless platform.

Mike: 00:15:48 – And actually one additional note I'll make that I didn't mention originally, but I think is also important is looking at kind of the

meta concerns of how do I actually build my software and get it deployed. And kind of the entire toolchain around continuous integration and continuous deployment and things like that I think is another key point, which is really kind of manage both in terms of sort of our first party services like CodePipeline and CodeBuild and CodeDeploy, along with the serverless application model, which is an Open Source specification for how to define and deploy serverless applications in a declarative model.

00:16:27 – But we also have a very rich kind of community of third-party partner solutions looking at monitoring tools like SignalFX and Dynatrace and Datadog and New Relic, as well as on the CI/CD pipeline side looking at CloudBees and Codeship. And then you also think about some of the other sort of serverless frameworks that have come out such as serverless.com and Apex and kind of all of these pieces as well. So I think certainly, yeah, there's an entire additional community of new tooling that's being constantly developed and innovated on that helps developers in terms of managing these applications, not just the actual core runtime components, which are obviously very important in and of themselves.

Barbara: 00:17:16 – Okay, great. Thank you. So we have time for one last question here. Can both of you give me sort of what are the one or two key takeaways that CIOs or IT leaders need to keep in mind as they think about serverless computing?

Mike: 00:17:32 – So I can start off with that. And I think the way that I typically kind of describe this and really sum it up is that any of the real kind of value propositions, any of the reasons that you see for moving to the cloud in the first place are really amplified by serverless. So if your goal in moving to the cloud is to reduce your procurement timelines from weeks or months, which you experience when you're running your own datacenter, down to minutes running on top of EC2 or in a virtualized kind of compute environment, serverless takes that away entirely. There's no more procurement at all. Capacity is essentially available to you right out of the gate.

00:18:13 – Similarly, if it's about HA and fault tolerance and you want to be able to run active/active across multiple availability zones and across multiple datacenters, moving to the cloud makes that so much easier and cheaper than it was when you were running on-prem. And going to serverless kind of ratchets that up again, just making it super simple. You get it out-of-the-box. You don't even have to kind of worry about designing for it in the first place.

00:18: 38 – So, yeah, I think the key takeaway in my mind, kind of at that high level, is just thinking in terms, okay, the real sort of value and benefits that drove all of our customers to the cloud in the first place, I think really serverless can be thought of as kind of the next evolution of really amplifying those benefits even further.

Miha: 00:18:57 – So when I'm looking across the portfolio of our clients, and Accenture serve some relatively large clients in a Fortune 500 portfolio, I'm astonished that I would say almost 80% of those organizations are already playing internally with serverless. They're already kind of having a small project or minimum viable project to build something to learn and actually try to understand what is that next new paradigm that is lurking over the horizon. My belief is that there will be a flashpoint somewhere in the near future where all of those small pilots are going to turn into a full production scale systems, and it will happen really fast.

00:19:46 – So if there is one key takeaway for _____ here is, if you are not playing with serverless today, you should because probably your competition is doing that already, even though they might not advertise it. Maybe they are not using it in the publicly facing solutions. But the knowledge accumulation of serverless is very obvious and very clearly happening within organizations. If you're looking at amount of engineers that are interested in Lambda sessions, let's say to reinvent and others, there will be a moment where suddenly huge amount

of systems will start to popup in the production environments on serverless. Get ready for that.

Barbara: 00:20:29 – Great discussion. Thanks. Really nice conversation. And thanks for listening to today's podcast, Trending Topics in IT: A Deep Dive into Today's Emerging Technology, reports on emerging enterprise technologies and is presented by cio.com in partnership with Accenture and AWS. Don't miss future episodes by subscribing to the IDG Tech Talk channel on SoundCloud and on iTunes. For Accenture, AWS, and IDG, I'm Barbara Call.

END