Deloitte.

Architecting the Cloud, part of the On Cloud Podcast

Mike Kavis, Managing Director, Deloitte Consulting LLP

Title: **Lift and shift? Cloud benefits may be hard to come by**

Description: For a variety of reasons—technical debt, application or platform preferences, or simple familiarity—many companies hesitate to refactor certain applications to be cloud-native. Instead, they prefer a “lift and shift” strategy that ports their applications to a cloud infrastructure without optimizing the code for performance in a cloud environment. In doing so, they may miss out on some of the benefits the cloud has to offer such as increased performance, ease of maintenance, scalability, and flexibility in data access and storage. In this episode, Mike Kavis and guest, Deloitte’s Sudi Bhattacharya, discuss some common drawbacks of not going cloud-native with legacy applications and typical benefits companies can realize if they refactor at least some of their code to be more compatible with cloud-native coding practices. This wide-ranging discussion also touches on machine learning (ML) and AI, and the potential benefits of leveraging the cloud for ML, AI, and advanced analytics at scale.

As referenced in this podcast, Amazon refers to Amazon Web Services and Google refers to Google Cloud Platform

Duration: 0:23:51

Mike Kavis:

Hey, everyone. Welcome back Architecting the Cloud podcast, I'm your host Mike Kavis, Chief Cloud Architect at Deloitte. Today on the show I'm joined with good friend of mine, Sudi Bhattacharya, and Sudi's a managing director in Deloitte's cloud engineering practice, mostly known for his excellent skills in the data analytics AI/ML space, which is going to be a big part of our topic today. So, welcome to the show, and why don't you tell us a little bit about your background?

Sudi Bhattacharya:

Thanks, Mike. Really happy to be here. My background has traditionally been legacy data warehouse, ETL, building these systems. I've done other jobs beyond consulting. I've worked as product manager in Microsoft, but last five years I have taken my legacy knowledge and upgraded my skills to cloud and mostly doing this large-scale migration or greenfield implementation of big data systems on the cloud, AWS, GCP, Azure.

Mike Kavis:

Cool, so since you mentioned legacy knowledge and converting it into modern cloud knowledge, one of the questions I wanted to talk about is very often people move – I won't say very often, but I've had a number of clients who, when moving to the cloud, they kind of just use a cloud as a data center in the sky and they take their old tools and their old database and ETL technologies to the cloud. And one of the questions I wanted to ask you, because we've both seen this, we've both been on clients where this has happened or where we replace those technologies, what are some of the tradeoffs that people get when they don't go cloud native or they don't use GCP's Big Courier, Amazon's Red Shift or some of the tooling that Microsoft has and they bring their old stack to the cloud? What are some of the things that they don't get out of that?

Sudi Bhattacharya:

So, that's a great question. I face that quite often. Customers, for a good reason, they are used to a stack. Let's say they have an ETL stack, they have a reporting stack, and they have been doing this for five years, six years, ten years sometimes. They have trained people, a lot of comfort. So, the moment they're now thinking about moving to the cloud, they want to do their ETL in their own stack. They want to do their reporting in their own stack. And usually, more often than not, that's actually not a good choice or a good decision. And you exactly mentioned it. Then you're just treating cloud as your data center, because what happens, the first problem that happens is these tools are legacy tools, although they are really actively trying to be more cloud-friendly, but they're not truly cloud-native tools.

So, you have to install them, you have to manage them yourself, you have to scale them, and that's a complete management overhead that you are not reducing and you are actually, in many cases, increasing, because these tools don 't necessarily integrate that well with, let's say, the logging infrastructure that cloud offers, the monetary infrastructure that cloud offers, and it's almost impossible in my experience, I have done multiple of these where I've taken this legacy tool and tried to automate this with the other cloud-native part of it, and that becomes almost impossible because they were not built – this installation process, or the job scheduling process, are not built with automation in the back of their mind, so you have to intervene, you have to put in a password, they ask you questions. It was such a nightmare experience to try to integrate this in the cloud-native stack, so my – those are the main reason. My recommendation is on this there is a really, really good reason, there's some special core capabilities, something that nothing else offers.

Mike Kavis:

Yeah, a couple examples I've run through, a particular client had a heavy, heavy investment in an ETL tool for data ingestion and all that, and they're like we just want to continue to leverage this tool. And they were moving to GCP in this example, and Google has some managed services in this space. The auto scales, falls over to different zones and regions, and all this stuff, and they just didn't want any part of it because of this investment. And I was saying, well, I get it but why don't you – when you do new, greenfield, why don't you move to this new world? Because all this work they have to do and all these licenses you have to pay just kind of go away. So, I understood their choice there, but I was kind of trying to push them to let's just do net new, the new way because ten years from now, you're still going to be using this tool that's even going to be more outdated and there's a lot of stuff you could take advantage of. So, I see that a lot, but the bigger part I see is the actual database engine choice. We have some third-party packaged wrapper around Hadoop, and we just want to bring it to the cloud. And you kind of mentioned some of it. I think people underestimate how much it takes to manage and patch and retain that stuff, where when you're using stuff as a managed service, some of that goes away. So, what are your thoughts, just focusing on the engine and the capabilities in the cloud, versus some of the stuff that they're bringing with them?

Sudi Bhattacharya:

Yeah, so that's an interesting question. I think if you take this company that takes the Hadoop distribution and layers their manageability tools, additional functionality, additional capability, and then they make it available either on-prem or cloud, of course if you actually install – if you actually take such a tool which is really built for on-prem clusters and take it on AWS or GCP or Azure, it's extra work, so you're not really getting the benefit of the managed – so-called managed service from the cloud. But at the same time, I think what I have seen is some of these products have tools that come with them, that is not offered in the cloud-native package.

So, for example, Cloudera Impala. If you are used to Impala, and if you have queries and Impala is really doing a great job for you, that's when people try to push back and say I'm used to that framework. And the argument against that is of course now there are compatible tools in the cloud that are fast, that can actually respond to your large queries against terabytes of data very, very quickly, but may not. So, that's why I think the real answer is that there has to be a use case, there has to be a reason, a business case, something that forces you to use Cloudera that let's say EMR on AWS doesn't offer or Data Proc on GCP doesn't offer. Then it makes sense because its business value is what we are going for at the end of the day, it's not the technology. But if there is no real reason, if they're apples to apples, I think it's perhaps unnecessary to, just for familiarity, to go with a cluster – a Hadoop cluster that's really built for on-prem and try to install it on AWS or GCP.

Mike Kavis:

Extending that a little further, one of the biggest advantages I've seen just from database choices is that, in the cloud, there's a separation of the data and database engine. So, for example, in AWS, you put the data in S3 and you can use different database engines to access that, where in the old days, everything was kind of tightly coupled. So, talk about some of those advantages and what some of those capabilities are, what's the difference when you start separating the data from the database engines?

Sudi Bhattacharya:

Right, so that's a great point. So, before I jump into that, there is this concept I want to just mention. So, the idea behind all this is data locality. So, let's say you have a distributed system, you have these nodes off a distributed system. You have disks attached to them, and you have processing attached to them. So, your processing will be fast if you have your data sitting in the same node where the processing is happening. If your process is actually going and accessing data in other nodes, that's essentially called shuffling, and when you do shuffling, that's when the whole performance starts degrading. So, even if you have data directly attached to your nodes, shuffling can completely slow you down. And the other – so the idea is to keep the data as close to your processing as possible, right. But that's valuable only in the case of when you have, let's say, you need really sub-second, 100 millisecond, 200 millisecond response time, or some crazy metric that you're trying to fulfill. Most of the loads, a lot of batch workloads that we look at, it does not have that type of requirement. So, in that case, imagine the scenario where you don't actually have to store the data in an engine. It's out there somewhere and you're accessing it, and potentially your processing will take slightly longer, perhaps, but in most use cases that's fine. But the real advantage is not just that the fact that I've separated it. The real advantage is there are other systems now that can go against the same data. I don't have to take the data and ingest it into my relational database, into my columnar database, into my Hadoop database, into my (Inaudible) database. It's out there on my object store, and all these engines can take advantage of it. At the same time, my data scientists can come and take advantage of the same data.

Now, there is an interesting caveat here that what happens – so underneath the covers, all these object storage, they get replicated. We don't know, and we don't worry about it, and that's how this is handled. The moment the same data set is being accessed by 50 different processing engines, it's not easy. I mean, if you don't replicate, it's not going to perform, because it's a contentious environment when read is happening from multiple sources, so you actually have to replicate. And AWS actually tells you – and most of them tell you that if you're planning to do this type of processing, to let them know so that they can actually maybe increase the replication factor a level higher, do it a little faster. So, I think the separation is a great idea, because of the multiple use of that same data source. At the same time, there are some processing that actually needs that data locality, high-performance computing, for example, is one of those examples where you really need data locality, really low latency queries, high value queries may not benefit from this where you really need the data right next to the processor if you can't write caching and all that. So, those are the main benefits.

Mike Kavis:

Yeah, a couple other advantages is in the on-prem world where you have to go patch one of these software packages, (Inaudible) to do that because the data's connected and if you have like terabytes, the time it takes – the downtime you take in that system is pretty substantial where even though patching is done for you automatically in the cloud, or if you didn't choose a managed service yet you were accessing it in S3 or something, that downtime is a lot smaller because you don't have to wait for your data to (Inaudible). That's my understanding. Is that a correct statement?

Sudi Bhattacharya:

Yeah, yeah, absolutely. So, that patching is really out of your hands. And since it's managed service, they are taking care of it, the cloud provider is taking care of it, and that's a big time saver, obviously. And then to add to that another twist that I think I'm seeing in the industry, so let's talk about this is (Inaudible) Spark, so Apache Spark, it's a distributed-processing framework. It runs on Hadoop, can run or any other cloud system also. So, what is now happening is it's actually people are taking this whole idea of this containers, putting Spark on container and wrapping it around Kubernetes, which is an orchestration engine around containers, and then Kubernetes comes with this rolling upgrade policy where you could take Kubernetes pods and upgrade them automatically, right. So, that's kind of – I see that as the next level of evolution that you are actually going into a completely managed container orchestration system and having this distributed data processing systems on them. So, Kafka, Spark, these are all at the beta stage I would say, but people are experimenting with these (Inaudible) framework on top of container (Inaudible).

Mike Kavis:

Yeah, cool stuff there. The next question is funny. You did an internal webinar for Deloitte a couple weeks ago, and one of the things you did – you kind of did some table setting at the beginning because, as you say, AI, artificial intelligence, and ML, machine learning, are kind of being used interchangeably and don't really understand the difference. And the very next week, I'm at a summit, and someone asks the speaker the difference between those two, and he didn't really have a good answer, so I said I've got to get Sudi on here, and I think that's important enough. These were a bunch of architects in the room and a lot of people were struggling. So, what – in your mind, what are the differences between AI and ML? And you used some pretty good examples to explain it in that webinar, so I thought it would be good to explain that to our audience here.

Sudi Bhattacharya:

Yeah, sure. Definitions are always challenging, so everybody has – there is no official definition and different people look at it differently. I've heard people say that everything is AI and ML is a small part of it, everything is ML and AI is a small part of it. So, maybe they're right, maybe they're wrong. So, that's – but that doesn't help anybody. So, the simple way I understand this is – so if you look at the word machine learning, there is an expression machine learning, there's a word machine in it. So, essentially there are these tasks, the predictive tasks, understanding patterns in data that humans can't do. For example, I have all this data about your buying patterns, so you are going to Amazon and you are actually buying different books, different stuff. I have lots of information of it, lots and lots of information, right. Now if I want to decide on what I'm going to recommend to you as a next possible action, a recommendation and what will you buy next, what can I offer you, what can I offer you that will induce you to make that purchase. So, that's sifting through a lot of data that a human being potentially can do, but it would take us a long time. So, this task where you have to sift through a lot of data to actually identify patterns, or assign a probability score of you doing something, is not a human-friendly task. So, those are ideal for machine learning. Machines can do that a lot better because the mathematics is known.

We can do the same thing; it would take us maybe one whole year to sift through petabytes of data, but – and we'll make mistakes. A machine can do that much faster. So, another good example is when you actually try to predict, let's say, churn. So, you – in telecom industry, I did a long time ago a predictive analytics model to really which users are highly likely to change provider, and then what can we do so that we entice them not to change, what kind of promotional activity, offering, et cetera? Same ideas, we have the behavior, we have the demographics, we know who they are and their pattern, and we know also test data that what kind of people have switched or changed. Our models, machine learning models, learn from existing data and then using that knowledge, predict the behavior. So, human beings can't do that very easily. So, that's my definition of machine learning.

Artificial intelligence, on the other hand, also has the clue of the meaning in the two words themselves. It's essentially just the reverse in some sense. We can actually do things that an artificial intelligence is trying to do very easily. Handwriting recognition, for example. So, we can do it very easily. Voice recognition, we can do it very easily. Image recognition, we can do it very easily. Machines can't. So, machines actually, for example, just to – I mean, now obviously they're improving, but to go back ten years, we could still do that. But then we did not have enough computing power and enough scale to do this in a cost-effective, inexpensive way. Chat bots is another example where you are talking to another person. Of course, we don't need any guidance. We can chat with another person. But machines, you can very easily flummox a machine. So, one thing I will mention here, I will throw in a little bit of deep learning.

So, if you think about AI, so it could be rules driven also, so let's say I have created a 50 kind of branch rule that if the other person on the end of the chat bot says X, I respond as Y, and I have, let's say, 100 rules, but beyond that I don't know what to do. So, these are rules-driven chat bots and that's how they used to be going back some years, but now we have AI-driven chat bots, so where you are actually using deep learning for the chat bots to tap into and then you become a little bit more intelligent, and that's where you're using advanced technology like artificial neural networks. CNN converted to neural networks—a whole bunch of new technology that are being used to do that. So, those are, again, going back to the basic definition, problems that human beings can solve very easily, but it's hard to automate if you use a machine and then you're using more and more powerful technology like deep learning (Inaudible). That's what is in my mind about machine (Inaudible).

Mike Kavis:

Yeah, the machine learning part is fascinating to me. So, I came out of the grocery industry many, many years when you would get retail data and grocery and pharmacy and we would look at your loyalty card and figure out what type of shopping behavior you and your family had and we would try to target you with the right offer at the right time, and the thing was I would build those algorithms, but they were on known patterns. We would take huge data sets and give it to these – they weren't called data scientists then, but these data specialists who would then run all these SAS programs and do all this analysis on a hypothesis they had, and they would come up with, we think this is a way to target, and then we would go build that and send the right coupons out. And now both Amazon and Google have an API to here's the recommendation API, just call it. And I literally had a team with a lot of people who spent years developing this stuff and now it's an API. And it's because they're letting the machines figure out the patterns, and it's just, you know, people who resist this stuff, they need to hear stories like that. Here's a company whose bread and butter was figuring this out, and now there's APIs for it. It's just amazing.

Sudi Bhattacharya:

Exactly, right. So, if you talk to any practitioner in AI/ML, they will say that the models, the algorithms, even the understanding that one could – can do this existed a long time ago. I mean, literally 20, 30 years ago. It's the technology that has actually now enabled us to not just do this at scale, but do this quickly, embed this into actually the end application, that's where the edge AI and edge ML concepts come, so where you are making the AI/ML sit next to the user, it's not somewhere far away in your data center. So, that's kind of the trend.

Mike Kavis:

Yeah, it's amazing I mean, going back to my example, I mean before we even built any of that, we spent 18 months standing up at the data warehouse. So, there was 18 months of pure cost with no business benefit just to get the data and the technologies in place to even start analyzing, where today you just bring your data set to the cloud, choose your engine and you're off and running. It's just amazing how this shifted. I think – and this whole space, data analytics, AI, machine learning is, in my mind, the number-one use case for cloud, because what is more elastic than this stuff. What's your thoughts on that?

Sudi Bhattacharya:

Yeah, yeah, absolutely, right. So, actually that's strategy of Google also. So, they are actually leading with data, leading with AI, leading with ML because this is – and now almost every company has done what you just described. They have spent their 18 months and they have data. They have maybe – it depends. I mean, some people have high quality, some people have low quality of their data, but there's lots and lots of integration now, but without that scalable elastic infrastructure, without – with the separation of compute and storage, with the ability to run these models on really high-performance CPUs like tensor processing unit or even GPUs, that has completely changed the playing field so if not anything else that's definitely a big driver for going to the cloud.

Mike Kavis:

Yeah, we could go all day on that topic. I appreciate your time on this. That's it for this episode of "Architecting the Cloud." To learn more about Deloitte or to read today's show notes, head over to www.deloittecloudpodcast.com. You can find more podcasts by me and my colleague, Dave Linthicum, just by searching for Deloitte On Cloud podcast on iTunes or wherever you get your podcasts. I'm your host, Mike Kavis. Thanks for listening and we'll see you next time on "Architecting the Cloud."

Visit the On Cloud library

www.deloitte.com/us/cloud-podcast

About Deloitte

As used in this podcast, "Deloitte" means Deloitte Consulting LLP, a subsidiary of Deloitte LLP. Please see www.deloitte.com/us/about for a detailed description of our legal structure. Certain services may not be available to attest clients under the rules and regulations of public accounting.

Please see www.deloitte.com/about to learn more about our global network of member firms.