Deloitte.

On Cloud Podcast
August 17, 2022

The Deloitte On Cloud Podcast

David Linthicum, Managing Director, Chief Cloud Strategy Officer, Deloitte Consulting LLP

Title:	**Observability: What it is, how it works, and why you need it now**
Description:	As cloud adoption increases and multi and hybrid clouds become the norm, operations become more complex and monitoring performance becomes more difficult. In this Knowledge Short podcast, David Linthicum talks about how the concept and practice of Observability goes beyond traditional systems monitoring to help IT teams observe, gain insight into, and maybe even predict what's happening in their cloud ecosystem to better manage increasing complexity.
Duration:	**00:19:49**

David Linthicum:
Welcome to this Deloitte cloud podcast Knowledge Short exploring a specific topic related to cloud computing. This is a short tutorial talking about real-world concepts in the emerging world of cloud computing. I'm your host, David Linthicum, cloud computing author, speaker, and managing director with Deloitte Consulting, and this is Observability.

So, Observability is really kind of a term that emerged over the last few years, and it's a mixture of analysts and vendors that really kind of glommed onto it. And it's really something that's important, certainly as we're moving into the next generation of cloud computing, dealing with complex architecture such as multi-cloud, where we have lots of things that are going on at the same time, and we need to have some sort of a scalable way to observe behavior and react, auto heal, and the ability to optimize these architectures.

Observability, in term and concept, has evolved over the years. At its core, it allows monitoring of internal states with systems using externally exhibited characteristics of the systems, and being able to predict the future behavior of those systems that are monitored, using data analysis and other technologies. So, that's a fancy way of saying that while we used to just monitor the current state, the real-time state of existing systems, and kind of wait for some light to turn red and us going off and fixing the system, it's more holistic than that. In other words, we're analyzing historical data, we're analyzing existing data, we're analyzing as-is and real-time data, and trying to figure out what's going to happen next, and therefore, we can proactively prevent issues.

Now, Observability, while it's often applied to operations, and certainly cloud operations, now we're seeing it applied to cost analysis, cost governance, cost monitoring, and also security. Security Observability is a thing that's starting to emerge because people realize that we need to take monitoring to the next level to really kind of get into a scalable position where we can maintain these systems ongoing. So, by monitoring what has happened, enterprises can reactively fix issues. So, whereas Observability helps with predicting issues before they become issues, hence help build a proactive enterprise. Ultimately, we tie automation to Observability, so we not only can spot problems but proactively fix them before they become problems. It's possible to build hyper-automated self-healing enterprises that fully understand what's happening currently, and will happen in the future, with the systems under management, how to predict and respond to likely outcomes. It's very important.

So, Observability is important because it takes monitoring to the next level. It's important in the management and monitoring the modern systems, and certainly complex distributed systems which really are what multi-clouds are, especially because modern applications are built with a more agile and faster velocity through the widely held best practices, because bolting on monitoring and management after applications are deployed, really doesn't work anymore because applications don't run standalone by themselves.

They're connected to databases; they're connected to core internal systems. They may be distributed across different platforms, and therefore just monitoring what a single application instance is doing only tells you not even half the story, maybe just a fraction of the story, maybe a hundredth of the story. So, Observability kind of takes this concept to the next level. It provides us the ability to have insights that go way beyond monitoring, insights which allow us not only to look at what's currently going on, but as I said earlier, and more importantly, the ability to predict future states, predict issues, and solve issues before they actually become issues. That's the core thing.

Part of the notion of Observability is the importance around modern instrumentation that needs to be implemented to better understand the properties and behaviors of an application. This includes performance and dealing with underlying complexities and distribution, and the ability to deal with dependencies that are really a part of those applications. Keep in mind that applications typically have many dependencies. They can have very raw and primitive dependencies such as the IO system of whatever native environment that they're on—cloud or not—and the ability to have dependencies that are loosely coupled—databases, other systems that are providing information, things that are not working along with the application—really stop the application from working.

So, therefore, we need to not only understand how to monitor a particular simple instance of how something is running, but how all these various components are linked together and how they're orchestrated to, in essence, drive a healthy application. And in doing so, have an understanding of as to how we're going to look at historical information—monitoring data around that application, around that database, around that system where it's hosted. And then look at the future dependencies and the future risks of things happening—the IO system going bad, security breaches likely occurring, and really not just monitoring something that only tells you half the story.

But it's the ability to see holistically not only how that application works and plays well with its own little ecosystem, and how the ecosystem works and plays well with the application and how everything really kind of jives together to create this interdependency of systems that have to work together to keep the application, keep the systems, and keep the IT operation healthy and running and changing moving forward.

The idea here is that things are becoming more complex, therefore traditional ways of doing monitoring that we've been doing for the past 30 to 40 years aren't necessarily going to be able to keep those systems up and running the way we want them to perform. Therefore, Observability provides those insights and takes monitoring, management, operations, security, governance, things like that to the next level.

So, a common question that I often get around Observability is monitoring versus Observability. Basically, monitoring has certain attributes that are related to passive consumption of information for systems under management, leveraging dashboards typically display the state of a system, and we've all seen those dashboards for years and years and years different management packages, things like that. Even back in the old mainframe days, they had a control system or a network operation environment and team that would look and monitor the current state of the system, and they were basically reacting to issues that occurred.

In other words, something went red on the console, and they knew they had to go out and fix some issue—IO systems, network outages, network routers, storage system failures, things like that. It's more tactical in nature. Monitoring is purpose built for static systems that really don't change much over time. We're monitoring IO of storage systems in very much the same way we did 20 years ago. We started to perfect the monitoring and management of various distributed systems. We're looking at application performance, we're looking at processor performance, even rudimentary things such as power consumption and temperature of particular systems and how they all interrelate one to another. So, we're moving from the tactical to strategic.

Observability takes a very different approach. It leverages the ability to gain an understanding proactively, again providing insights that not only see where you've been but where you're going. Very important. It asks questions based on continuous understanding or hypothesis. In other words, it's not only looking at the current way in which we're observing the characteristics, historical data, and therefore predicting future data, but also changing the way in which we're doing Observability to improve it moving forward. And, so, certainly when you have AIOps, artificial intelligence operations, and that's a whole set of tools that are out there that are able to, in essence, implement Observability if you're talking about what technology actually exists to do this, that's it.

Then we can ask additional questions and improve the processes very much like a human being would, so it's living, breathing, thinking, and understanding that we have to continuously look at different dimensions and how we're going to observe the behaviors of all these systems individually and holistically and subsystems to improve upon their performance and reduce the number of outages, hopefully to zero. They're purpose bult for dynamic system platforms with widely changing complexity. So, we may go from 1,000 cloud services under management to 3,000 cloud services under management, and the idea there is we don't want to triple the amount of people around and triple the amount of tooling around to monitor those systems. We're trying to do something in a much smarter way.

Therefore, Observability provides us the ability to expand these systems. We're able to deal with complexity through abstraction and automation, for instance, and the ability to dynamically continuously improve, as we just mentioned, on how Observability works. Also, it's the ability to identify what if in trending applications to predict the possible future failure. So, in other words, what's going to happen if this IO system goes down, and how many applications will it affect, and how do we indicate whether an IO system is actually running into issues?

In other words, if an IO system is starting to generate errors, how much time do we have to fix a hardware problem, or basically if you're in a cloud, moving it from one virtual server to another. And, how do we stop the applications that are dependent on that IO system, say a storage system, for example, or a database, from failing because of some sort of underlying storage, and can we do this without stopping processing? And that's a key thing, so we need to look at where the applications are trending. We're able to predict failures that are moving forward, and, more importantly, we're able to proactively stop them from failure.

You have to remember monitoring and management is typically very rudimentary today, very primitive. In other words, we wait for things to happen, and we go fix them. And while that seems good enough for many of the enterprises out there, it's not going to be good enough if you're solely dependent on your system that's becoming your business. If Observability can eliminate that or reduce that, that's going to be fairly significant. Kind of keep that in mind.

So, why is Observability so important *now*? Why am I doing a knowledge short podcast just on the concept of Observability? Well, the core principles involve applications and infrastructure monitoring operations management that are very familiar to us. The nature of how new applications are built, how they're architected or designed has changed over the years a great deal. Cloud-based applications are depending on multiple servers often, RESTful APIs and microservices are going to be part of it, and so in other words, we're dealing with complexity on top of complexity. Not only are cloud-based systems, certainly multi-cloud systems, complex unto themselves and have lots of moving parts, but those parts have parts, and those parts actually have parts. So, we have to break them down and monitor them at the level we need to monitor them.

The other thing is that serverless and service-based applications, which is pretty much what we're building today and certainly container-based applications, Kubernetes-based applications are a part of that. They have little visibility into the underlying infrastructure. So, in other words, they run themselves and they really are dependent on these native services being there, whether it's services that are native to a particular container and those actually may call out to a native server on the particular platform that the container is hosted on, or in some cases Kubernetes. And, your ability to kind of keep everything running is really assumed to be something that's never going to stop by the folks who build and deploy the application.

They don't have visibility into the platform that they're running on, and, therefore, if the processer's going bad, or if the network device is failing, or an IO system is running into trouble, they have to have the ability to fix that proactively. And, so, what we're doing is saying, okay, they don't see the native infrastructure that they're running on, and, therefore, we're going to create a native infrastructure that's as resilient as we possibly can using the concept of Observability but certainly implemented using technologies like AI ops and other application management systems that are leveraging the Observability concept.

The other thing is we have highly distributed applications that are based on things such as containers and microservices with the ability to scale up and scale down in a matter of seconds. Keep in mind that we like using cloud computing because we're only going to pay for the resources we're using, and so we can call up and allocate and deploy as many virtual servers as we need, and we can remove those virtual servers and therefore reduce the billing rate. So, if we're getting into what we call web scale where we're supporting millions of simultaneous users accessing a particular application, that's really kind of another level of play in terms of how we're ability to make these things dependable.

So, if we're scaling up and scaling back, Observability provides us with the capability to proactively monitor how those things are going to affect the underlying systems, how they're going to affect security, how they're going to affect governance. And then having a more holistic view of all these various systems and how they're tied together, but more importantly, how their behavior now is living up to optimizing the capabilities of the applications that are supporting the users, and more importantly, moving forward, how issues are going to be spotted in advance of us having those—having to endure those issues and fix them proactively. Very important.

Hybrid and multi-cloud architectures that are dependent on multiple providers, in house and external, that are complex to keep track of are really one of the larger reasons that we're moving to Observability. You have to remember that complexity is really the downside of something like multi-cloud. So, if we're moving into a complex distributed architecture, multi-cloud is an instance of that, because in many instances, we're moving from a single cloud deployment where we have 1,000 services that are under management in the service catalog that developers and users are able to allocate as they need it, whatever best-of-breed technology they need to leverage, with say 3,000 and 4,000 servers that are now under management. And, so, you have a bunch of choices, and developers and solution builders will leverage those choices.

They'll leverage the AI system that they want; they'll leverage the databases that they want; they'll leverage the application development and deployment environments that they want. And, so, therefore we may go from, say, 800 services under management to 10,000 services under management. When you look at how all these things are interdependent, one to another and also the abstraction of these services and the ability to leverage services within services, and really just kind of the fact that we're giving the developers and the innovators in the company, which is very good, the option to leverage any number of services to solve their issues. And if we're missing one that they need, we'll onboard that system.

We'll go from four clouds to five clouds, because maybe that one cloud has a particular service that they feel is going to be a game changer for their particular solution. So, of course, we could put our foot down and say you can only leverage services from this particular cloud provider. That's not necessarily going to be optimized for your business. If they're able to leverage technology that's able to make their solution a game changer, that goes right to the bottom line of the business. You've got to remember businesses are being evaluated by their ability to be innovative in their marketplace—not as much how quickly they're able to sell something, or how quickly it'll move into a market as much, but the ability to kind of move into market with different innovation techniques.

So, multi-cloud, complex distributed systems, the fact is that we're not giving up our legacy systems as quickly as we thought. We're still maintaining those data centers, and we're moving into many more moving parts. And by the way, we can't increase our operations budget to support this new direction as to where the systems are going, so, we have to figure out another way to do it. Observability becomes a way to do it, and then we have different tools and technologies that really kind of implement the concept of Observability.

Keep in mind most monitoring and operations tools such as AIOps, they claim a role in Observability in some respects. This is really kind of difficult to figure out because if you're moving into Observability, you're going to find the different tools and technologies are going to define Observability differently and do it differently. So, my advice to you is figure out what Observability means to you. In other words, what are the value points and what do you need to do within your enterprise as far as keeping systems running, keeping systems secure, optimizing environments, optimizing cost spending, thin-ops operations, things like that.

And then figure out the tools that are able to live up to the way in which you're defining Observability in terms of the core attributes or roles that you feel will be more valuable to your business. So, Observability, like another buzzword, is getting tossed about lots of different places. You see it on TV commercials now. Certainly, a valuable point, but you've got to keep in mind that this comes down to your requirements and backing the appropriate technology into your requirements and really kind of living up to and looking at the values that Observability means to you, specifically. That's very important, probably the most important lesson I think we're going to talk about during this podcast.

So, anyway, I urge you to learn more about Observability. You can check out my blog on infoworld.com. you can go to Deloitte.com, search for Observability. There's a bunch of work we've done out there including research reports, things like that I think you'll find valuable. And continue to listen to this podcast because we're going to cover Observability a great deal moving forward. It's going to be part of the underlying infrastructure, and for good reason. It's very important.

If you enjoyed this podcast, make sure to like us, rate us, and subscribe. You can also check out our past episodes, including those hosted by my good friend, Mike Kavis. Find out more at deloittecloudpodcast.com. if you'd like to contact me directly, you can e-mail me at dlinthicum@deloitte.com. Until next time, best of luck on your cloud journey and stay safe.

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.

Visit the On Cloud library
www.deloitte.com/us/cloud-podcast