EPISODE 382

[INTRODUCTION]

[0:00:01.1] JM: A decade ago, a Microsoft developer might have been defined by the fact that they build C# applications on Windows. Today, a Microsoft developer is just as likely to be writing JavaScript for Linus. The company has repositioned itself to focus on cloud services, SaaS products and enterprise artificial intelligence. Jason Young and Carl Schweitzer host the MS Dev Show, a popular podcast about Microsoft developers and technologies.

On their show, Carl and Jason explore the rapidly expanding marketplace of services on Microsoft Azure and they talk to experts about how these services are built and what they're used for. I attended the Microsoft Build Conference this year and enjoyed talking to Jason and Carl there as well as afterwards. Also, at the Expo in the Microsoft Build Conference, I was just amazed by the new technologies that the company is building and how successfully they have pivoted to cloud services. It's quit fascinating to watch the rise of the cloud. It's such a competitive and open market for several giant corporations that have overlapping, but also some disjoint concerns and disjoint customers. It's a market that I really enjoy watching and one way that I get information about it is through the MS Dev Show. I enjoyed talking to Jason Carl on today's episode, and I hope you enjoy it too.

[SPONSOR MESSAGE]

[0:01:41.2] JM: In the information age, data is the new oil. Businesses need data and there's no better data than real time data, which is why Amazon Web Services built Amazon Kinesis; a powerful new way to collect, process and analyze streaming data so that you can get timely insights and react quickly to new information. Here's the thing, websites, mobile apps, IoT sensors and the like can generate a colossal amount of streaming data, sometimes terabytes an hour. That, if processed in real-time, can help you learn about what your customers, applications, and products are doing right now and respond right away.

Amazon Kinesis from AWS lets you do that easily and at a low cost. With just a few clicks, you can start sending data from hundreds of thousands of data sources simultaneously. Loading in real-time, it lets you process and analyze the data and take actions promptly. All you need to know is SQL. Kinesis also gives you the ability to build your own custom applications using popular stream processing frameworks of your choice. With Kinesis, you only pay for the resources that you use. There are no minimums, no upfront commitments. To learn more about kinesis, just go to kinesis.aws, and let's get streaming.

[INTERVIEW]

[0:03:18.8] JM: Jason Young and Carl Schweitzer are the hosts of the MS Dev Show. Guys, welcome to Software Engineering Daily.

[0:03:25.1] SD: Thank you. Thanks for having us.

[0:03:26.1] JY: Yeah, nice to be here.

[0:03:27.3] JM: Yes. I made a resolution recently to do more shows about Microsoft and Microsoft technology because I don't deal with them on a regular basis. I'd never programmed in C#. I just haven't interacted with that ecosystem very much. I come at it very much from a kind of an outside observer perspective. What does it mean to be a Microsoft developer today?

[0:03:54.3] JY: The world is definitely different. You mentioned C#, but you don’t have to do C# at all to be a Microsoft developer. The world is a totally different place now. If you look at VS Code and how popular that's become, that's been causing a lot of people to take another look at Microsoft technologies, “Hey, maybe Microsoft isn't so evil after all.”

Then things like TypeScript as well. Then if you do look at something like C#, that world is changing as well with .NET core and some of the great work there and being able to kind of get the best of both worlds, being able to get a super mature language that has a ton of features running on a super reliable platform where you can get great support and you can run it across all different operating systems. I mean I think any previous assumptions that any of your listeners would have, I would ask that you kind of throw those away and take another look. That’s kind of the short version.

[0:04:53.0] CS: Also, look at some of the newest, popular kind of desktop applications out there. You mentioned VS Code, but there's also Slack. Those technologies that are written in Electron. Those are JavaScript applications that are written in this chromium shell that you can also use the desktop app converter to package those in an APPX and deliver via the store. You don't have to use all the traditional models that Microsoft has had in the past and. They're really opening, really kind of accept whatever you want to do.

[0:05:27.5] JY: Exactly.

[0:05:29.5] JM: The Microsoft company culture has certainly changed over the last decade or two. How has change propagated through the community of people who develop for Microsoft technologies?

[0:05:46.0] JY: Yeah. I think Carl and I are going to have a little bit different perspective here. Me, being a Microsoft employee; him, working for a company that does a lot with Microsoft technologies. I mean internal in the company, there's just been a dramatic shift not just with the technology I talked about before, but things like Linux. There's a substantial base of Linux virtual machines running on top of Azure. If you look at even Windows 10 now, Linux is integrated with that. We're really embracing all of these other technologies and it's really causing people, developers in the community to sort of rethink their assumptions. I think there either is a group of people out there that they want to stay within the Microsoft ecosystem.

What's great now is I think kind of the non-Microsoft ecosystem is sort of part of the Microsoft ecosystem now if that makes sense. You can start to use these other technologies and you don't have to necessarily wait for Microsoft to innovate in a particular area. You can start using these third-party database products on top of Linux, for example, and run all of that on Azure.

For me, that's been the biggest thing is just opening up that world of possibilities. Also, Microsoft culture changing in a way that's making those things work well together instead of trying to rebuild something that already existed, only doing that where that makes sense. First, the default being, "Hey, this product is popular in the marketplace. How can we just use that and extend that or work with that, integrate with that," that type of thing. Carl?

[0:07:32.7] CS: Yeah. Looking at it from kind of the outside looking in, there's different pockets of Microsoft have different levels of interactivity. There's some that put out their code on GitHub where you can check it out, even submit poll request and take it in. What I think is even more valuable is a lot of the discussions that are happening, like you can go to .NET team on GitHub and look at their discussions and planning for the next version of like C# or the .NET framework and have input. You can say, "Hey, I need this problem solved," or "You guys are looking at this. I don't think that's a problem or I think you're looking at it the wrong way," and you can have a lot of impact on the future of Microsoft products, ask somebody who's not directly associated with Microsoft.

I think that part of the openness is taking Microsoft in different ways, whether it's directly open sourcing things or just being a little bit more open to the community. I think different teams are dipping their toes into it in different ways. I also look at user voice. Microsoft has really adopted user voice to gauge feedback. If there's something, a product that you really want to give feedback on, that's another great way to get your voice heard.

[0:08:51.2] JY: It's interesting. If you look at what I'm working on today, like literally this day, it is Docker integrating with IoT edge, our IoT edge platform is open source and it runs in a number of different languages, so you can run Node.js. In this case, the partner I'm working with, they want to do .NET Core work because they already have a .NET Core codebase. We're running .NET Core in a Docker container that will get — .NET Core itself is open source, and it will get deployed to an open source framework that runs on the edge to enable these IoT devices.

You get the strength of all that open source technology and the ability to even, like Carl said, submit feedback and also submit poll requests, but you also get the power of the Azure cloud and you get the power of things like identity at scale from Microsoft. I don't know. It's just super exciting for me to be able to just use the technologies that I think are the best and sometimes those are the — Our Microsoft product, sometimes they're not. Guess what? I'm going to use the thing that works the best to solve the problem at hand just like everybody else.

[0:10:03.0] JM: Yes. I saw you guys it Build, and Build was exciting. There's so much interesting stuff released and the Microsoft Azure cloud offerings are a pretty interesting. It does make me wonder sometimes like, “What is Microsoft best at today? What is their core competency?”

[0:10:24.1] CS: From my point of view, I think they’re still best of what they’ve always been best at, building a platform. If we look historically, that started off with Windows and that was really dominant for many years. I think it's really been transformative seeing Microsoft flip their core competencies to be cloud base. When you look at how Amazon started their cloud, they initially just took their extra infrastructure that they had and put it out there for sale or for rent or whatever and that kind of started the cloud phenomenon.

When you look at Azure's approach, they kind of took the other ways, like, “How can we make this cloud a platform?”

[0:11:02.2] JM: By the way. I think that story about Amazon, that’s accurate not accurate.

[0:11:06.1] CS: Either way. Even if it’s not accurate, that is their core competency from the beginning is been a very IaaS heavy approach. Even if the story is wrong, the core concept is still good. They have a very strong IaaS story. As you start going into the platform bits, that’s where it gets a little bit weaker. Microsoft started with the platform being stronger and they’re continuing to grow that as well as having a good, now, infrastructure approach. I think that Azure's cloud is a lot more interesting than Amazon's cloud even if it is in second place right now by usage.

Microsoft has thousands of developers are that are really good at creating tooling, creating developer experiences. I think leveraging what they're good at is making Azure just really exciting place to be right now.

[0:12:00.7] JY: Yeah, I think that really hits on it, developer experience. Pretty much everything you said really boils down to developer experience. There's the tooling and there’s the mindsets and all of those things, but then there’s the global distribution. I don't want to sound like I’m marketing and talking about the number of regions and things like that. I’m not going to get into that. Again, developer experience, like developing PaaS services on top of Azure that make it so that it's as easy as possible to to deploy your code.

If you look across the services, that's always what it's about. I have a problem that I want to solve and how do I do that? You saw that initially with some basic platform services and, now, overtime, you see more focus on particular verticals. If you look at the IoT strategy you'll see that around some very specific scenarios but while the underlying generic platform still exist. In any case, again, it's how can I allow the developer to be super flexible but at the same time provide them with some tools that are really going to accelerate their development.

Then the other thing, if we're talking about Azure specifically, is that there are only a handful of companies on the planet that can afford the type of scale that AWS, Microsoft, Google, can achieve. The thing is, you have to spend —I don't know. I don't even know the exact numbers. You have to spend 20, 30, 40 $50 billion to build a cloud at this type of scale and you have to start at least five years ago. There's just no other way around it. I mean those are some of the skills there.

Again, I think everything boils down to developer experience and not everything is perfect and stuff comes up over time. As far as — Especially on the enterprise side, starting with that audience and understanding how that audience works and then starting to spread that out and focus on other verticals and other types of developers, that's really where I see the strengths lying.

[0:14:06.0] JM: What were the platform as a service products that Azure started with?

[0:14:09.9] JY: What they started with? If you look at the super early days, the easiest thing was deploying NET Code, and those were super early days and things were pretty difficult there. If you look at things like web apps for running your code, even today, if you want to deploy a web application, you can do it for just a few bucks a month and being able to deploy your Node.js, your Java application, your.NET application, Python, those types of things. You can deploy that super easy. You can actually put the code into GitHub. Azure will go pick up that source code. It will build it. It will deploy it automatically for you. That's kind of the core there.

Earlier than that, there were, and they still exist. There's web and worker roles. Now, there's Azure functions which it would be similar to something like AWS Lambda. Obviously, AWS came out with that one first. Then if you want to know kind of where Azure has innovated first, I would say on the Azure IoT side. If you look at the Gartner Magic Quadrant, the completeness of vision, I think Azure is quite a bit ahead there on the IoT side. Not to say you couldn't do the same things on AWS, but I think there's a pretty good advantage on that side.

[0:15:27.5] JM: What are the advantages? What are the specific technological gaps that AWS has that Azure has patched up on the IoT side?

[0:15:38.0] JY: That's a really good question. If you look at where we were — I kind of jump in — I don't look at AWS on a daily basis, so it's really hard for me to do a specific competitive analysis on the spot. If you look at — What I can tell you is at Build, not this year, but last year. We're talking a little over a year ago. I help run some of the IoT labs, and the experience I thought was phenomenal. What we had set up was at each station we had a Raspberry Pi 3 device and then we had a laptop there that had Visual Studio installed. It was basically an hour long session where you would set up an IoT solution. What you would do is you'd create a new application, you'd add a few references in for the client device libraries. Then with basically one line of code to send in the connection string, you provision the device and set the connection string and that connection string was specific for that device for security reasons.

Then what you would do is write that code in Visual Studio and then you could actually hit run, execute. Visual Studio would actually connect to the Raspberry Pi 3, it would deploy the code for you. You could actually step through the code and view the real-time variables and that type of thing. Being able to do two-way communication with a device with basically one line of code for each operation, one line of set up, one line of code two send telemetry data from the device, one line of code to receive data from the other side. Again, it always comes back to developer experience. That experience was super clean and you could take somebody who maybe wasn't even a .NET person who didn't really fully understand IoT solutions. You can have them look at that code and fully understand that.

We’ve just been building on that experience past then. Again, I’m not specifically on the IoT team, but they've added a lot of great device twin functionality and there's a lot of enterprise, user feedback that has gone into that and it's really been moving fast and maturing fast and it’s been great to see the progress there.

[SPONSOR MESSAGE]

[0:17:58.0] JM: Bugsnag is an error monitoring tool that enables developers to identify, prioritize, and replicate bugs in a time efficient and enjoyable manner. Automatically capture errors and diagnostic data in any app. See errors group by root cause and focus on fixing errors that have the greatest user impact. Instantly track the performance of each application release with interactive dashboards. Automatically capture a stack trace for every error and get automatic breadcrumbs for crashes so that you can easily reproduce and fix any error.

To-date, Bugsnag has processed over 10 billion application crashes from thousands of top technology companies. To get started, go to bugsnag.com/sedaily and create an account. It takes three minutes to get up and running, an Airbns, Lyft, and Shopify are already using Bugsnag for their bug tracking and crash monitoring. Try all the features for free at bugsnack.com/sedaily.

Thanks to Bugsnag for being a sponsor of Software Engineering Daily. It's much appreciated.

[INTERVIEW CONTINUED]

[0:19:16.1] JM: We’ve moved from a place in the past, I think, like 20 years ago or 10 or even 15 years ago. I'm not sure where you want to look at the timeline, but we’ve gone from a place where IT was more of a zero sum game to a place where it’s a much more positive sum game and there are lots of different players. There's not a one-size-fits-all solution for most of the problems that you would want to solve. Do you think Microsoft has been able to adjust to that new world properly where it’s not zero sum and you don’t have to be focused on the competition?

[0:19:56.5] JY: Yeah, absolutely. I guess that's a lot of what we've been talking about, is having all of these things work well together. People will come to me and say, “I wish you guys could do this,” and it comes down to like, “Hey, we would have to have a monopoly to do that,” and I think it's nice having multiple players in the space to kind of keep everybody honest and to watch that innovation come up from a couple of different places. That's the world that we’re living in now.

It’s not Microsoft dictating terms to the world. It’s the developer community. It’s the users that are really making the demands and have been empowered, “Hey, we want this stuff to be easier,” and then you have all these companies competing over that and you just see it working out better for everybody. I think the battle right now is in machine learning as an example. Microsoft is investing. Actually, there was a significant reorg focused around artificial intelligence and machine learning. Really, that's just kind of a result of the things that we've been doing anyway.

Obviously, Google is a big player there as well, but that battle is really heating up, because we have a lot of industrial partners that are they’re already able to collect a lot of this data. It's really a matter of getting the data to the right place, throwing a whole bunch of compute power at it and then coming up with insights. Being able to figure out when equipment will fail. Being able to figure out — There was one recently figuring out from the sounds of something in a pipe, like where the weak spots on the pipe.

There’s situation after situation where you can use this existing data and use machining learning against it, and there's so many opportunities right now that I don't even think if you look at Microsoft. Amazon Google, and even if you bunch in a whole bunch of other companies there, there’s no way to even keep up on the opportunities. There are far more opportunities than any company can even execute on. I think everybody is winning in this right now, I think. There are some mind-cheer things going on, but it's really as fast as we can come up with functionality and as fast as we can throw a course at a problem and throw smart people at a problem. We’re able to really improve the world, improve organizations’ ability to execute and be efficient.

[0:22:21.8] CS: When you asked your original question, I was thinking back, what really has changed a lot in the last six years? For me, especially when you talk about IT as the blurring of the line of what IT and developers are with the concept of dev ops. Microsoft had really kind of stepped up pretty quickly in providing a lot of the tools that people in those kinds roles need in their VSTS or VS online online products that they have.

Six years ago, we had the VSTS on premises and that was it. It was essentially your source control and your tasks and that's it. Now, that it's really been involved not only into an online service but one that fits the IT minded role as well. It's not just a tool that developers use to get their jobs done. It's something that the entire team uses to coordinate, work together, not just build the code but to coordinate its release and manage it in production as well.

I look at that, that there are specific things. Jason was talking about a few other things, but I think there's some very specific things. You look at some of those trends at the time, and Microsoft not only has kind of foresaw other tech trends and met them, but they also saw trends that they necessarily weren't pioneering themselves, like the dev ops one and really came in early was a good set of tools to fulfill these evolving and new roles.

[0:23:49.5] JY: Yeah, that's such a good point. Donovan Brown from Microsoft, he has this great presentation. He goes on stage and he wants to show off the dev ops pipeline and he actually surveys the audience and says, "You guys pick the language and you pick where you want me to deploy it to." They'll pick like, "I want to deploy a Ruby application to AWS," and then he will, onstage, build out that dev ops pipeline. I think that's just an awesome demo, first of all. Second of all, it just shows it's not some guy with PowerPoint slides saying, "I'm going to deploy C# to Azure and show you how easy it is." It's much more impactful than that.

[0:24:31.4] JM: Yeah. Studying this cultural shift of Microsoft has been really interesting to me. I was focused a lot — Actually, last summer, I was reading several books about Microsoft and the antitrust case that happened in the 90s. I was actually living in Bellevue at the time, so I was thinking a lot about Microsoft and you walk around Bellevue and it just feels like Microsoft town, basically, and you talk to people about Microsoft all the time. It was just very interesting to kind of studying while living in Microsoft town.

One of the things that I walked away from, I actually did a show with a lawyer who had written a book about the case against Microsoft back in the 90s, and I walked away from an interview with kind of feeling like maybe the antitrust case was a little unfounded. Have you guys looked into it in detail? What are your feelings about the antitrust case? Do you think is overblown? In retrospect.

[0:25:37.2] JY: Yeah. My only comment on it, first of all, I think that was a completely different company. It was a complete different company. I don't even really think that I work at the same company that that was back then. I would say that it’s completely different people and a completely different culture. That being said, me, as a consumer, and I’m sort of speaking like not as a Microsoft employee now. What frustrates me is I use all different types of devices. I use an iPhone. I have an Apple Watch, I have AirPods. I use everything. I use Mac OS as much as — Well, I would say less than I use Windows these days. I try to use everything.

If somebody tells me a piece of technology is cool, I'm going to try it. I’m a technologies before I’m a Microsoft employee. It’s really frustrating for me using something like an iPhone where you click on something in it once opened a browser, and guess what? It has to be Safari. You don't get a choice. The same thing happens with almost all of their built-in applications. They hold such tight control over it and they really dictate to the user, like, “You will use your phone in this way.” I try to use Chrome and it works on there, but then, again, when I click on a link or something from a different application. Guess what? You’re going to Safari. There's no choice.

If you want to click on an address and you’re in your calendar, it's going go to Apple maps, for example. They get favorable treatment on everything. I use Ways all the time, but guess what? When my phone is locked, I don't see the navigation. If you use Apple maps, everything becomes magical and it works totally different.

From that perspective, I find that really infuriating and it is because I look back at like what happened to Microsoft and I don't know if it was fair or not. I can’t be the judge of that. But I will say just knowing that that happened and that Microsoft was penalized for his, it is frustrating now seeing other companies exhibit the same behaviors that Microsoft was punished for and nothing happens. Again, I’m saying that as a consumer. That’s just my personal frustration.

[0:27:44.8] CS: Whether it was founded or not, I think one of the key things is you have people that kind of live through that timeframe. There's a large amount of them that just have an opinion that I didn't pay attention to it, but Microsoft went through a lot of flak for what they did, so they must be evil. They've never updated their perspective. Whereas if you look at immediately after all the consequences were handed out, whatever the fines and legal things were, Microsoft went out of its way to not be that company for quite a few years. Even under the same Bill Gates and Steve Ballmer administrations, they do their darndest to not repeat those mistakes.

What's happen since, not only did they kind of reverse course on whatever it was, but when we had that starting towards the end of Steve Ballmer's period, he started transitioning the company and such has really championed this — Like we talked about earlier, this openness, this accepting of all of these different technologies. Whether or not it was fair that what happened then, it does have a lot of impact on how people think of Microsoft. A lot of those people, what they really need to do is they need to open up their eyes and their ears and say, "Hey, Microsoft is doing something different. Let me see what they're doing." Then make a new opinion.

[0:29:03.2] JY: Yeah, because Microsoft, I feel like they're the ones that are the most open. The ones that I think are playing the most fairly. I'm sure you could find an example or two where you're like, "Oh, what they're doing over here, I'm not really a fan of it." Overall, it is really like focusing on both privacy, on consumer trust, and then also focusing on being open and being honest and even things like roadmaps and showing those. Yeah, I mean I really like kind of where we're sitting right now as far as positioning. Yeah, I wish people would just take a second look. If you want to form the same opinion, then so be it, but just do yourself a favor and take a second look.

[0:29:46.0] CS: Different teams are at different stages of their transition as well. You look at the web and cloud teams, they're very far along in how they interact with the public and are open. You look at the web community calls. As an MVP, we get access to inside information, but they put some of the MVP calls out on Google Hangouts for everybody to hear, not just the MVPs.

I think that's really huge when the public can have that much access to information. Whereas there's other teams like the Windows team, they come from a lot more — Historically, we wrote this software for three years and here's the next update. They're making the transition on their cadence and they're still changing on how they communicate. That something that they're still the progress of changing and evaluating and see what works for them.

[0:30:41.2] JM: Yeah. So bizarre. Jason, you made the point about people who have not updated their views about Microsoft since getting convinced that this is an evil company. I remember I was in college and I had multiple professors say “Oh, this is the evil empire. This is a terrible, evil company.” I just kind of assumed, “Okay. Sure. They are my professors. They teach computer science. They must know what they're talking about.” Then studying in more detail, what you find is, “Okay, this company was just ahead of everybody. They were so far ahead of everybody that they dominated the cutting-edge and, yeah, they set some defaults and you could frame their default setting as being anticompetitive, as being inconvenient for the consumer in favor of Microsoft’s business,” if you take an extremely short-term view or if you’re Netscape and your biased towards taking that view.

You could also make longer-term arguments that those browser defaults were, “Oh! The same arguments that Apple is making today,” as you pointed out. I think it was shortsighted. The decisions, the legal decisions that were made were shortsighted and they were based on kind of perception of Bill Gates, perception of this company that was dominant, but they didn't take into consideration the fact that there were alternate path the you could take. Different companies could be built. You could build an operating system from scratch, like Linux did.

There was the idea that, “Oh, that is not necessarily a threat to Microsoft, because Microsoft has their networking stack locked down and Linux computers aren’t going to be able to interface with them.” Who knows what would have happened if there wouldn’t have been legal pressure. Maybe Microsoft would have been doing stuff that was more convincingly antitrust, but I don’t know. It’s just so weird when you analyze it in more detail and you see,” Okay, was this company really doing something that was inconvenient for consumers?” and just the fact that they got their name dragged through the mud has had such a such a bad outcome for them.

I think companies like Uber — Sorry. Companies like Google are learning from this in terms of optics. I think that's why Google's perception, they have tried so hard to keep a clean perception.

[0:33:07.4] JY: Yeah, and failed in many regards. They’re known as the company that's selling your privacy.

[0:33:13.0] CS: I think that the other thing that makes that really hard too when you talk optics is, all of us, we live in the US and we have a certain way of looking at things. When you start moving to the EU, they have different values. Different things are core to their being, so what we may think of as may be a very small issue, that could be a very important thing to those kind of people. In moving out through the world, there's many more different kinds of cultures.

It's not just navigating that in one area. These are global companies. They have to tiptoe and maintain their branding and goodwill everywhere, and that is a very hard thing to do as well.

[SPONSOR MESSAGE]

[0:34:00.4] JM: Simplify continuous delivery GoCD, the on-premise open-source continuous delivery tool by ThoughtWorks. With GoCD, you can easily model complex deployment workflows using pipelines and you can visualize them end-to-end with its value stream map. You get complete visibility into and control of your company's deployments.

At gocd.io/sedaily, you can find out how to bring continuous delivery to your teams. Say goodbye to deployment panic and hello to consistent, predictable deliveries. Visit gocd.io/sedaily to learn more about GoCD. Commercial support and enterprise add-ons, including disaster recovery, are available.

Thank you to GoCD and thank you to ThoughtWorks. I'm a huge fan of ThoughtWorks and their products including GoCD, and we're fans of continuous delivery. Check out gocd.io/sedaily.

[INTERVIEW CONTINUED]

[0:35:01.8] JM: You guys work with a lot of enterprises. I think this is why Azure is the second most popular cloud, is that a lot of companies feel comfortable with Microsoft technologies, and Microsoft already has its sales clause into the companies who bought enterprise Windows licenses, so it's becomes easier to kind of transition them to Azure. What's the process that you're seeing enterprises taking as they moved to the cloud?

[0:35:33.2] JY: Actually, I’m going to disagree with your statement there. We do have — It’s interesting way of saying it, we have our sales clause in there. Here’s the thing. Here's the here's the thing that we’re learning over time. You can't go to companies and say, “Hey, you should buy some Azure. This is the thing you need to solve your problem, so you should buy a whole bunch of it and use that Azure thing.” That doesn't work. You can't sell the cloud like that.

You used to able to sell box product like that. You could sell office, and you still can. There’s a cloud- based model for Office. For those products, you go in there and you say, “Hey, how many uses do you have? What kind of term do you want?” Those types of things, and that was really the old model.

For Azure — And this is why my team exists. For Azure, things are different. What we found is the best way to get people to adopt Azure and really understand those things is to develop alongside the developers at those enterprises and it's really like a completely different model than what we used to have. Again, I mentioned today doing this Docker, and I won’t list all the technologies again, but this is for a hackfest that I'm doing with a partner next week. They have a set of problems that they're trying to solve.

We’re not really even thinking about Azure at this point. We're trying to figure out how to solve their problems. They're coming up to Redmond. We’re going to throw a bunch of smart people at it from our side. They’re sending up a whole bunch of smart people. We’re going to figure out what tools we can use to solve this problem, and I suspect —I should say, we already know that, ultimately, the purpose that we’re collecting this data for is to gain insights out of it through machine learning and through business intelligence tools, and that will make their customers more efficient.

Again, I just kind of want to reiterate, the way that we’re doing that is working side-by-side with their developers, figuring out the best tools to use and that's how you get people to use Azure. It makes them want to use it instead of a salesperson saying, “Hey, you need to buy some of this to solve your problems.”

[0:37:40.3] CS: That works really well with, especially the PaaS offerings that are at Azure. You have to get in there and get in there in the trenches with it. When you look at the IaaS, you can kind of take a little bit more of that box model approach too because you can look and you're like, "Okay, okay you have so much server capacity and you're running the following applications. If you virtualize that and just kind of shifted the cloud, you'll save X. Therefore, blah-blah-blah."

You can sell differently depending upon which product it is. When you look at kind of the early cloud, what's sold a lot was those kind of IaaS, the VMs. As we see people understanding the cloud more and understanding the different capabilities that it has, they're he realizing that PaaS is where they want to be. It's understanding how do you develop for it? What are the best patterns and practices to use in your apps to make them work better in the cloud?

[0:38:31.8] JY: Yeah, it's such a good point.

[0:38:34.1] JM: When you talk about PaaS, what kind of services are you talking about? Because when you guys were illustrating the differences between AWS and Azure, it sounds like you have an opinion that Azure is more focused on the PaaS. What exactly are those PaaS offerings you're talking about?

[0:38:54.2] JY: I can give a really simple example of that. I think one of the best examples would be look at SQL Server. I work with really large global enterprise partners, and what Carl said was step one is that they take their replication as is and they just run it in a VM and Azure. I think it's really boring and this whole thing of like you can save money by doing this, in a lot of cases it's not even true. I think if you're trying to move all your stuff as is and trying to save money, you're probably going to have a bad time. The effort you put into it versus any potential saving that you're going to have is going to be minimal.

In that case, what they would end up doing is they'd run their software in the virtual machines and then potentially they'd also run traditional boxed SQL Server in a virtual machine, which is perfectly fine. We have a lot of guidance for how to do that, but the pattern that I see with a lot of partners I work with is, really, the next phase is really the reason that we're all doing this. Instead of using SQL Server installed in a VM, you use SQL Azure, the PaaS service. You go out there, you say, "Give me a database, and boom, SQL Azure within seconds will provision you a database. You can put your data in there. You get the exact same interface to it. You connect to it exactly the same way. Obviously, it's a different connection string. If you need to scale that up or down, you just request more resources. You can scale it up or down as it's running.

Then we'll do things like replicate that data to other data centers. We'll automate the backups for you. We'll do all of those cool things. Then there's also a pattern that I see in regards to the databases like that where, traditionally, the software might be sold to a hundred different customers and, basically, the install it on prem and they each get their own SQL Server instance locally, and that's just what they're used to doing.

Whereas now, when we re-architect this, we make the application multitenant and we can still give each customer their own SQL database. But we're basically putting it into an elastic pool so that the partner can basically pay for the amount of overall capacity that they need and then it kind of gets divied up between each person and then if there are some spikes or whatever, it doesn't really matter at all, just evens out in the end.

The end result is that you have an application that it's far more reliable, it's far quicker to deploy, because you're probably going to have some kind of a dev ops pipeline where you're making changes and you're getting those in a production quicker and you're using future flags and that type of thing, but then you're also just adding redundancy. You have multiple services that are geo-redundant. The reliability of your application goes up, plus the accessibility goes up.

There's a whole list of advantages to going that route, and that's what I'm doing with most of my time during the day.

[0:41:53.5] CS: At the end, when you're looking at PasS, you're only caring about the services that you're caring about. In this case, Jason talked about SQL. You're only caring about the database. You're only caring about the schema, the table structure, and so on and so forth. You no longer have to think about, "All right. Now, I need a server to run it on. That needs an operating system. I need to install SQL Server itself. That needs a license. I have to talk about EA agreements." You don't have to manage any of that anymore when you start using the PaaS offerings.

[0:42:21.6] JY: Jeff, were you aware that SQL Server now also runs on Linux?

[0:42:25.4] JM: Yeah. I knew that, and I also knew about the PaaS SQL offering. Google has cloud SQL, Amazon has Amazon Aurora. Everybody has their PaaS soffering, and I think that's fine. I think it's great. It doesn't really seem like a differentiator. It seems like we're basically in place today where we've got almost feature parity among the clouds and it seems like they're fairly interchangeable at that point.

[0:42:58.1] CS: I think if you look at component by component, like if you're just looking at SQL or something like that, or if you're just looking at the web hosting that might be available or just like an event hub or service bus pieces, you could like, "These are all kind of equivalent. It's who's a penny cheaper here or there." I think where it gets really interesting in Azure is how easy it is to combine all of these into a full-scale solution.

In a lot of cases — You mentioned Google, but Google doesn't have everything that Amazon and Azure have, but AWS and Azure are a little bit more compatible or feature equivalent. When it comes to just kind of tying those pieces together, it's been my experience that it's so much quicker to do that with Azure. Either they put it directly in the portal where you just have to kind of connect things there, or even with the Visual Studio IDE. It's nice just being also like to open up a console app and then once you have your app running, you can right-click and publish that as a web job up in Azure just by entering in the right code. That just makes it so much more convenient.

Then it's in with your source control with all of your other — Your web apps, your database, all of that can be managed via Visual Studio in one spot. As a developer, I can use my one tool. I can stay productive all day long and kind of just keep adding and adding and adding as I need it.

[0:44:30.0] JY: Yeah, could you build any application, any of these cloud platforms? Absolutely, yes. We're not going to — Carl and I aren't going to sit here and lie and say, "Azure has feature X, and feature X is the thing that nobody else can create and that's the whole reason you want to be in Azure." That's just not the case.

Like Carl mentioned, there are equivalent services for the most part between all of them, and if there are missing on one cloud provider, guess what? It’s probably on the roadmap and it’s probably going to show up at some point.

Again, I think it comes back to — For me, it comes back to developer experience and the ease of deploying those things. There are certain feature combinations that do work together really well. If you look at some of the announcements from Build, if you are running .NET and you're running it in Azure and you're running it in Azure website, which is really an easy path forward. Like I mentioned, you can just throw that code in GitHub. You literally just point your website at it and, boom, you’re up and running with a deployment pipeline.

[0:45:33.5] JM: You're talking Azure, like the Azure benefit, is you have this well-integrated experience.

[0:45:39.0] JY: Yeah, I was actually going to get to the big benefit there. In this particular case, what you end up getting is if you're using Visual Studio, you are able to actually set a breakpoint in the cloud in production. It's not a breakpoint that it will actually stop it, but it'll take a snapshot. Then you can actually step through the code locally and diagnose production issues. You're in this world-class local IDE connecting to Azure and connecting to your .NET application that were easy to deploy.

You could demo the entire thing end-to-end in 15 minutes, from creating your code to pushing it into GitHub, deploying it, and then being able to actually debug in production. I think there's a lot of value in that.

[0:46:25.5] JM: Do you know if AWS and Google have that same capability? Do you know what the integration across services story is with those services?

[0:46:35.7] JY: Obviously, I specialize in Azure. I don't know every detail of that, but my understanding is the future I talked about in particular is like a differentiating feature at least for the time being. I don't think Amazon has any kind of IDE out in the marketplace, for example. If they do, I don't think they have that comprehensive tools that all work together. I could be wrong.

Again, I'm sure they have some other scenarios where it’s kind of the same case where if you’re using X, Y, and Z, those technologies will work together well. If you look at — Especially with our enterprise customers and that the massive .Net codebase out there and even like Node.js now, you look at those code bases out there and the number of people that are —Visual Studio customers. They’re going to have a great experience in Azure, and AWS it's going to be a little more difficult.

[0:47:28.7] JM: Do you think that the future that users want is the highly integrated cloud experiences. For me as a consumer, the goal would be to have really good easy ways to mix-and-match between services in the cloud. I'm not sure if —

[0:47:48.6] JY: No. They want it all. Absolutely. If you want a virtual machine and you just want to do everything yourself and kind of stay out of my business, I want that to work great. If you want to use that with SQL database that's a PaaS service, I also want that to work great.

At the same time, if I do pick a set of technologies that make sense together and are commonly put together, I want them to work well. There's PaaS services in Azure, for example. There's streaming analytics, for example, which will read from event hubs automatically. You go in there and you say, “I want to read from that.” You can have it read from other locations. Same with Azure functions. Maybe functions is an even better example, which again is kind of like AWS Lambda, if your listeners are familiar with that.

If you're using Azure functions and there's a defined set of inputs and outputs, you can write your own connector to write to anything, literally, anything that you can possibly dream up, but at the same time if I want to read from event hub, for example, which is my ingestion point where I’m receiving this pipe of data, I don't want to have to take this connection string and use some kind of client library and blah-blah-blah. I want to have the ability to do that, and you can, but if I know that I'm using both those technologies, there's definitely this desire to just say, “Just wire those up for me automatically, and I'm happy to have the choice to do it, but please make my life a little bit easier if you can.”

[0:49:24.9] JM: Cool. All right. I wanted to talk a little bit about the MS Dev Show, because we are all podcasters. I know we're up against time, but maybe if people are curious about learning more about Microsoft technologies, definitely check out MS Dev Show. I listen to it on a semi regular basis. Maybe real quick, what are the best and worst aspects of hosting a software podcast?

[0:49:47.9] CS: I'll say worst is just like — It comes down to a non-technical thing; scheduling, getting everybody's calendars to line up and emailing everybody and getting everybody just agree, like, "We're going to meet here on this time. Oh, yeah, and it's his this time zone too. Is that Daylight Savings? Oh, crap! Hold on." It's that thing. That's the worst part.

The best part for us, at least for me, is something that actually doesn't happen very often, but when we go out to conferences and user groups, people know us and that's really cool. We get to have a lot more conversations with a lot more people than we normally would. Before we started doing this, I would still go to conferences and user groups and I would go out and reach out to people and have these conversations. Now, I'm finding more and more people are coming to me. It's really cool just being able to interact with the people that listen to our show.

[0:50:38.9] JY: Yeah, for me the best thing is all of the great people that we get to talk to, people — It's pretty much the exception if somebody says no or kind of defers to somebody else to come on the show. As a rule, anybody will accept and say, "Yes. I will come on and talk about something."

We can really go to the authoritative experts on particular technologies and get them to come on the show. For me, I use it as a learning experience. The worst part is, is just all of the business overhead. I did want to mention too, we cover Microsoft technologies, but we also cover — We only say that we cover anything that a Microsoft developer would be interested in. We will cover Node.js. We will cover any kind of third-party stuff.

We had an episode on Electron Shell for Slack. I'm just kind of looking through here right now. We had one on dev ops. Mesosphere, Docker and containers. We had one on a balloon that went up into space. Things like that, even like career superpowers. We cover a whole bunch of different things.

Really, our niche is if you were a developer that uses any Microsoft technologies, whether it's Visual Studio, whether it Azure, whether it's Windows, or even TypeScript, or VS Code, any of those users, like we want to expand your horizons and teach you something new. That may or may not be a Microsoft technology.

[0:52:03.9] JM: Yeah. You also have some different segments, like the news and some other stuff that you explore. Anyone who's curious about that, if you can't get enough software engineering podcasting, then check out the MS Dev Show. Thanks for coming on Software Engineering Daily, guys. It's been great talking to you.

[0:52:20.9] JY: Great talking to you. Thank you so much.

[0:52:22.4] CS: It's been a pleasure.

[END OF INTERVIEW]

[0:52:25.9] JM: Heroku's operational experience lets teams focus on what's important, maintaining application health and providing an optimal experience for end-users. Listen to our podcast with Andrew Gwazdziewycz from Heroku's engineering team to learn more about the importance of application health and best practices for monitoring application user experience.

This episode aired on February 28th and you can find it on the Software Engineering Daily website. You will also learn about Heroku's metrics platform architecture and how it laid the foundation for auto scaling. This was a fascinating episode, and if you haven't heard already I hope you tune into it. Thanks to Heroku for being a sponsor of Software Engineering Daily.

[END]