**RedZone Podcast Episode #83: Why Is The Google Cloud Platform Good For Your Business? – with Mark Mandel and Francesc Campoy Flores**

Bill: Francesc and Mark I want to welcome you to the show today.

Francesc: Thank you.

Mark: Thank you very much.

Bill: Well this is a real pleasure, because I've never done a three-pronged podcast here, so this is really fun. Have you guys ever done three people on, on your own podcast?

Francesc: Three guests, no. We've been four people at the same time, but two hosts, two guests.

Bill: Okay.

Mark: Yeah.

[00:00:30]
Bill:
Well, I'd love to give my audience ... This is a little bit of a different spin that I want to take for today, and I'd love for you guys to talk about your own podcast and how did it get started? And what do you all hope to accomplish from your own podcast? Can you give a little history about that for my guests for today?

Francesc: Sure. I can get started, and Marco will give more detail.

[00:01:00] So basically I've been in Developer Relations at Google for I don't know, five years or so? And for a while I had been thinking about the idea that podcasts are very useful. Like I personally listen to podcasts. They're a very nice way to learn and they're very personal. You can really improve your [inaudible 00:01:12] things and express things the way you really want, which is not the same way with writing.

So I was thinking about making a podcast, but I always thought that it was a lot of effort. And then Mark joined Google, how long ago? Three years?

Mark: Two and a half, something like that? Yeah.

[00:01:30]
Francesc:
Yeah, two and a half. So he joined Google and I was his mentor and once, he was like, "Let's do a podcast." I was like, "Okay, let's do a podcast." So there you go. Now we're doing a podcast.

Mark: That's pretty accurate. Yup, same thing here. I've been a podcast listener for a long time, realized there wasn't one for GCP, Google Cloud Platform, and yeah, Francesc and I were sitting down, and we were like, "We should do a podcast." Like, okay. So we didn't think it would be a lot of work, but we discovered that it was a bit of work.

[00:02:00]
Francesc:
Oh yeah, after the fact we realized -

Bill: See clearly, I got directed to you from the Google folks, so you clearly caught their attention.

Mark:

[00:02:30] Yeah, we've had really good sponsorship internally. I mean, the audience reaction to the podcast has been great. We get even just people on the ground coming up to us and saying, "This is amazing, we love it." Both internal to Google and external to Google, which has been fantastic. So the support from internal has been fantastic as well. So it's been a hugely positive experience, I must say, starting this up.

Francesc: I think that we're in a very privileged position where we have access to people like product managers and engineers and people that are really, really working the products for a very long time, and we are able to get them on the podcast and ask them all of the questions that people have. So it is kind of an easy way for us to just extract all of the information that we want. And apparently other people find it interesting too.

[00:03:00]
Bill:

[00:03:30]
That's great, and I love talking to product managers and I've had a couple on my show, and it's really interesting how deep people are in certain areas and how much wisdom they have. One of the things I want to pretend like I was a CIO, and in many respects, I am, I really want to dispel - I spend much more time as I shared with you talking about AWS and Azure but I really want to understand more deeply about the Google Platform and we just had a guest on talking about the security aspects of the Google Platform but id love for you to talk about what you guys feel is one of the things that the market, that the buyers, really don't understand about the Google Platform that is important. It could be related to containerization or Kubernetes, or anything, but what do you think it is?

[00:04:00]
Mark:
I have answers. Do you want me to come in? I have thoughts.

[00:04:30] I think there's a lot of differentiating things, I think between the Clouds, we all have various opinions on stuff. One of the things I particularly love about Google Cloud is the philosophy or the open cloud or the hyper cloud - within Google Cloud we use this phrase a lot. We definitely have the proprietary type stuff where it's like, you use this, you get really quick to market through this technology because we manage it all for you, but you're sort of locked in. But I think we really talk a lot about the Open Cloud.

[00:05:00] So using open-source things like Kubernetes, such as containers, basically enabling it so that if you want to run on us plus somewhere else, or run on another cloud provider or maybe your own machines, or move from your machines over to the cloud, open source systems such as Kubernetes and container orchestration as a whole really enable you to do that. And we like to think we're gonna be the best place for you to run those workloads, but at the end of the day, you win because of the competition. As a consumer, you win because we're all vying for that arena and that kind of Kubernetes standard which I think is really cool. And we'd be basically leaning on the idea that we think we're going to be able to give you the best experience, to be able to keep you here inside Google Cloud. But we're also going to give you the ability to leave, if you need to.

[00:05:30]
Francesc:

[00:06:00]

[00:06:30]
I definitely agree with that, and following with the thought of trying to make it the best place to run Open Source technologies, we have things like TensorFlow or Kubernetes. And basically the whole idea is, you could run it by yourself if you want to. There's no vendor lock-in at all. You can do whatever you want. What we're gonna do is, we're gonna try to do it, so we're gonna try to minimize all of the bad ops and all the operations that really, they're not part of your business. Like you're doing something cool on top of those tools. Those tools should be there for you, rather than you being there for the tools. So for instance, with Kubernetes, we have Google Container Engine, and with Container Engine, to start a new cluster is literally one form. You click a button, and after a couple of minutes, you get a new cluster. And then from there, all of the operations are automated. A machine goes down, it just got restarted, but also if there is new versions of Kubernetes and everything, everything just gets updated without you having to care that much about it, which is really what people want.

Bill: Is it really designed for people developing their own systems, or would you say that you could move your own workloads directly into the platform? Or Is it more from a development point of view that you see people moving into Google Cloud Platform right now?

[00:07:00]
Francesc:
Yeah, so we've seen both. As far as our podcast, the Google Cloud Platform podcast, we've interviewed many different people, and we've seen people doing, I think it's called - I always find it funny - shift and lift -

Mark: Lift and shift?

Francesc: Lift and shift. It isn't shift and lift [crosstalk 00:07:17].

Mark: Lift and shift.

Francesc:

[00:07:30] One of those where basically they have their own software, running on their own premises, and then they just decide to move it to Google Container A, or Google Compute Engine where you get your instances and you decide exactly what machine you want, how much RAM, how many processors, if you wanted to use, and stuff like that. You really have a very fine-grained decision on this.

[00:08:00] But we've also seen people doing completely the opposite and just building directly on the platform like Greenfield and really using all of the completely managed services like App Engine. So App Engine basically, it allows you to give some code and to say, "Run this, scan as much as you need and just make it work," right? So it's perfect for things like, I don't know, like Khan Academy or even Angry Birds, the back end of Angry Birds was running in there, 'cause they have really huge spikes.

Bill: What's the name of that tool?

Francesc: It's App Engine, Google App Engine.

Bill: App Engine, okay. Got it.

Francesc: So Application Engine, App Engine.

Bill: Oh, I gotcha. App Engine.

[00:08:30]
Mark:
To go on from that, I think that it is interesting ... For a while, one of our biggest episodes was just our episode on virtual machines, right? So Google Compute Engine, if you need a particular instance, you want to spin up Ubuntu or Windows or CentOS or whatever. And while we generally spend a lot of time on the podcast talking about that new, shiny thing, something like App Engine, something like Kubernetes, something like Spanner, all those really cool, special things, sometimes people just need an instance with, you know, Compute and they need to build stuff. That's really cool, and I think that's great.

[00:09:00]

[00:09:30] But then we also have the big managed services that enable you to not have to do a lot of the work, something on App Engine where you just want to run a bunch of, let's say, HTTP end points, or even something like Spanner where you wanna have essentially a globally-replicated data store where you can basically do sequel and relational operations. We have some talk about Compute as the continuum, right? So how much do you want to control, versus how much flexibility do you want, and how much do you want us to manage it for you? So the infrastructure on VMs on one end, where you do everything, but you have all the control and the flexibility you like, to something like App Engine which is much more constrained, that we manage so much more for you. And then Containers and GKE, container engine in the middle and a few other options as well. But as the same thing as those ... Yeah, cloud functions, yeah, if you wanna get into the lambda side of things.

[00:10:00] But then also on storage, where we have everything from [inaudible 00:10:00] Google Cloud Storage, where you wanna store binary stuff, to if you wanna run your own math sequence, this is on VMs, you could totally do that as well, but we have managed MySQL, and then we have things like Datastore. Firestore came out recently, on the Fire-based brand, which is tightly coupled to ours. A couple of managed NoSQL data stores, as well as things like Cloud Spanner and Managed [inaudible 00:10:22] and things like that. So there's lots of ways you could chop and change how much control and how much you wanna have managed and you can make that as a business decision, essentially.

[00:10:30]
Bill:
When the managed service piece comes up, with you talking about that being more of the DevOps orchestration, is that where someone can leverage essentially more of a workflow-based way to manage? Is that, when you guys, giving you more of that capability, helping them orchestrate that, is that what you mean by managed service?

[00:11:00]
Francesc:
Yeah, so basically the whole idea is to go on with the example for storage. Let's say that you want to run a SQL database, right? You could just simply start a couple of VMs and just run MySQL there, right? Now the problem is, you're gonna have to manage a bunch of things. One of them is, well if there's any version on MySQL, you need to upgrade it. You need to make sure that everything is running all the time, if it goes down, how do you manage your replicas and all these things. You need to also configure your backups and all of these things, right?

[00:11:30] If you're a DVA, you know all of these things you really need to do. Now, if you know the DVA and you want to have just a MySQL instance running, you do not need to care about all of these things. So instead, you can use what we call Cloud SQL, and Cloud SQL is simply MySQL and now also Postgres, you can choose. But basically a MySQL or Postgres instance running in the Cloud, fully managed by Google. And this is cool because basically this comes from Google doing it internally.

[00:12:00] Like at Google, there's a lot of things that used to use MySQL, now I'd say not that much. But we used to have a lot of people using MySQL, we didn't want every single thing to manage this by themselves, so we created a managed service where they could just say, "Hey, give me one instance." Right? So we basically just made this available for everyone.

Bill:

[00:12:30] Which I think is really good, because I think from a lot of the innovation conversations I've been having is that the ability to stand up ... The ability to stand up and test and build something on the fly, and fast, I think is what companies really need, to have that capability. I think doing it within their own four walls though is to me problematic, because it's using resources and you've got to fight to have the services allocated for you. So the ability to do it on demand seems very appealing.

Francesc: For me ... Go for it.

[00:13:00]
Mark:

[00:13:30]
I was gonna say, I agree 100%. It can be very difficult otherwise to do that whole test-stage prod type thing in a cost-effective way, when you're dealing with real machines. Especially if you want to test out ... We had a game, Phoenix One at Guiding Kingdoms, they literally just did a lift and shift across over to Google Cloud, and one of the big things they were talking about is, "Now we can test real load on a system that looks exactly like our production system, because it's all virtual. We can build it up, do the load testing, tear it back down, and it's a fraction of the cost of keeping this thing running in real time." And we can also then say, "Okay, what happens if we doubled everything in size? Can we still handle the load where our bottlenecks sit? If we want to do 10 times the load that we normally would have within this game, then suddenly you can start playing these sorts of games, but in a much more flexible way than you would be able to, otherwise."

Francesc:

[00:14:00] And even from the point of view of a developer just writing something really small, like I've had this experience before, where I'm writing, I use a MacBook to develop and I'm trying my things and it works on the machine, I use Docker and it all works, and then I really want to deploy it to production. Creating a new instance is gonna take around one minute, and then I can exit stage into it directly and just try my thing and make sure they actually work, right? There's where you realize that actually, your expectations were not actually accurate. So what you thought would work, it doesn't.

[00:14:30] Being able to do this so fast and without having to ask permission to anyone to have a new instance really changes the way it develops so it's really good.

Bill: So what is Kubernetes? Am I pronouncing it right, Kubernetes? From just a layman's perspective, what is that and what is it compared to, and how would you define it? And then what is it like, what is it dissimilar to?

Francesc: I'm gonna let you go Mark.

Mark:
[00:15:00]

[00:15:30] Okay, that works. So the tagline reads something along the lines of, "Kubernetes is open-source container orchestration at scale". What does that actually mean though? The way I like to look at it, you sort of take the buzz words out of it. The general gist of it is, you have a cluster of machines. So you probably have maybe 3, 5, 10, 5,000. But you have a huge number of machines. You need a piece of software to manage what processes start up on them, where they start up, whether they have discs attached to them, taking all their logs, aggregating it together, making sure things are healthy and available, and if they're not, restarting your systems, deploying new versions of whatever software you have. And basically all the general stuff that you need to manage and run software across a large cluster of machines such that you can scale and deal with information coming in and going out, right?

[00:16:00]

[00:16:30] This is basically what Kubernetes does. There's a bunch of things that are underneath it in terms of like, you wanna talk about Docker and Containers and standards and governance and stuff. But at a very core level, it really just gives you a massive amount of control over exactly what processes are happening across a large cluster of machines, that gives you a bunch of different ways to do that, from configuration files, you can deploy it and tell it exactly what you want it to do. Like "I want five instances of this thing running somewhere in my cluster." At the end of the day, I don't care where, as long as it runs and I can access it, that's great. And Kubernetes will take care of that for you. All the way through to [inaudible 00:16:30] PIs that you can integrate directly into Kubernetes so that you can have very fine-grained control over exactly what's happening, as well. And this is all open-source, which is super cool.

Bill:

[00:17:00] So okay, a couple of question I have. So I just had this happen this week, I've been talking to a CIO and he says "I'm just losing track. All my developers are all in my production code. I have no rules-based access control to parts of the code," so he doesn't have authentication, he doesn't have authorization, none of that. So where does security start for you guys? It's probably a problem that many companies have.

Francesc: So first of all, he has to fix that. That is indeed a problem.

Bill:
[00:17:30] This is not a problem I'm sure, that Google has, or big companies. But this is like small to medium businesses, 100 employees to 5,000, which is most of the U.S. businesses. This is real stuff. This is hard stuff.

Francesc:

[00:18:00] The cool thing about Google Cloud platform is, this is something that is relatively new and it's something that we added because basically our customers were asking for it, right? While the fact that you really want to have, for every single way of authenticating, you want to allow those people or processes to do something specific, and no more than that, right? So there's what we call IAM, Identity and Access Management I think it stands for.

Bill: Sure, sure.

Francesc:

[00:18:30] And basically that's going to allow you to say "This person, they're able to access this specific bucket on Google Cloud Storage, or they're able to manage this instance on Google Compute Engine," right? So you're gonna be able to give very specific rules to what people can do. Now the thing is, you can use exactly the same idea to apply it to processes. So you could say, "I have this process. This process is going to be running on this instance, and this instance on Compute Engine, will only have access to read, only read, from Cloud Storage."

[00:19:00] You're gonna try to minimize the surface area, of how someone getting access to one of the pieces of your system will be able to impact the rest. And then on top of that, the same idea is also available on Kubernetes, if that's the question. Kubernetes also has rule-based access control. Mark knows way more than me about that.

Mark:

[00:19:30]

[00:20:00] I know a bit more. I've not played with it in depth, but yes, Kubernetes definitely has our back authorization, so again it's open-source, built in. If you're using Google Cloud Platform, there're links between our back end and IAM that you're able to take advantage of. So that you do have that fine-grained control over who can deploy and when they can deploy and all that sort of stuff. To tie it into more open-source, which I think is totally valid as well, we also have deep integrations and a continued involvement in another open source called Spinnaker. Spinnaker works in the continuous delivery side of things. So in the way that you were talking about where you said you have a CIO who has those sort of problems, it also sounds like what they may possibly also need is a system such that when a developer build a feature, there needs to be some kind of automated pipeline that A, does tests, and make sure everything's okay, but once everything's okay, automates the deployment part.

Bill: Yes, yes. Wouldn't that be nice?

Mark:

[00:20:30] Wouldn't that be nice? So Spinnaker's another open source project that you can use in conjunction with Kubernetes and Container Engine and Google Cloud. Actually, a variety of products across Google Cloud wherein it gives you lots of pre-built stuff for doing automated deployments and so it's not this one particular developer sitting in a cubicle, hoping to write this config file just right, to make sure things are okay. You have an automated deploy that you're able to set up, and then say, you know, maybe you do it every time. Maybe you do it at a certain time of day. Maybe there's a push button once, some smoke tests, but you have an automated system, so that that potential for things going wrong really gets reduced.

Bill:

[00:21:00] So I'm probably going to use the wrong word, or maybe I'm not even going to remember the word at all. But the managed, there was the controlled, the more controlled piece of the platform, and then there's the more open, free part of the platform where you can do more of what you want. My question's more on the more controlled part, that I guess would have more discipline around building and deploying. First of all, what's the name of the product?

Mark:

[00:21:30]

[00:22:00] I was talking about Spinnaker. Spinnaker is an open source product. If you want to run it - I'm just actually double-checking - but we have a thing called Deployment. Sorry. We have an automated launcher for common things, and Spinnaker's in there. So you have a one-click push to deploy, so you can just go in and fire that up so you don't have to necessarily set it up. But it works with the wider Google Cloud Platform as well as [inaudible 00:21:42] thing ... If you're looking for something that sits high into that, that's maybe a little bit more managed, or maybe even just a little simpler, we also have a product called Container Builder, which is a bit of a misnomer in how it's named. It sounds like it's a thing that builds containers. What it actually is, is a thing that uses containers to build stuff.

[00:22:30] But what it essentially is, is a way to set up a set of basically, steps within a workflow. You can use simple configuration files so you can say things like "When this code goes into a source," let's say GitHub or one of our source depositories, using Git or any version control system, pretty much, or a variety of other hooks, really, "then I want these steps to run." Those steps might be, "Run some tests." They may actually be "I'm just gonna do a simple deploy somewhere," all that sort of stuff. So you have several tools at your [inaudible 00:22:38] to be able to control what happens when you push code into your version control.

Bill: Can you build the security layer in, at least on the access control piece in as you go as well, does that work people through that process, so they're not trying to reverse-engineer it on the back end?

[00:23:00]
Francesc:

[00:23:30]
Something you could do to avoid the problem where you were mentioning, that basically everyone has access to everything in production, is to basically say "No one has access to production. Everything needs to be done through either Content or Builder or Spinnaker or any other way of deploying that you decide to use. And that's basically what we do at Google, really, where what you're going to do is, everything needs to go through the [inaudible 00:23:27] repository, right? You cannot just go somewhere and say, " I'm gonna change this job." No, you don't do that. Instead you go and change the code or you change some configuration. That gets reviewed by other people, and then once it gets merged, then it's going to be deployed to QA and then there's gonna be some tests, then we'll actually be hitting production, right?

Bill: Sure. [crosstalk 00:23:48]

Francesc:
[00:24:00] This is something that it requires organization and a little bit of, what's the word I'm looking for? Basically, you need to be able to tell your developers that this friction is actually useful in the long run.

Bill: It's got balls.

Francesc: Yeah. At the beginning it's gonna be adding friction, but later on it will make it so that your deployments will work faster. Once you automate your deployments, everything is so much better.

Bill:

[00:24:30] We had the same thing in the firewall side. Sometimes we get - not so much in these days, but sometimes these firewalls gets so much Swiss cheese, there's just so many holes punched in them, you just gotta flatten them and basically say "Nobody gets access anywhere." You gotta build the rules from scratch, instead of granting everybody access through the firewall. So it's an interesting parallel.

Mark:

[00:25:00] But the nice thing is, 'cause we have IAN and are backing up stuff, you can still give people access, in terms of being able to visually see what's going on, so if they need to de-bug something or see what's happening, those tools still exist and they're not getting in the way of developers that way, but not to the point where it's sort of like, "I can just change this thing here," it's more " I can still see what's going on." So you don't want to be making it so that your developers aren't able to triage issues if something goes wrong, 'cause software fails. That happens.

Francesc:

[00:25:30] In my opinion, if your developers have SSH access to machines in production, and they actually need it, there's a problem that you should fix somewhere. In general [crosstalk 00:25:23] you should have better ways, yeah, you should have better ways of debugging things. You need a better system to monitor, you need better observability of your system, basically.

Bill: That's a great point. There's a root cause issue going on, and they're trying to have a sledgehammer killing the mosquito. And they don't need it. That's a great parallel.

[00:26:00] So while we're on the programming languages, I noticed both of you - or maybe it was Francesc, had some really interesting material online about the Go Programming - the Go Programming Language, correct?

Francesc: Yup.

Mark: Yes.

Bill:

[00:26:30] What is that? I'm gonna ask a question that hopefully doesn't make me look too idiotic, but if I'm a decision-maker and I'm trying to make good decisions about use of me ... I want to rapidly iterate, I want to develop new tools and products and apps for my business, and I've got this whole legacy, a going concern business so I can't disrupt the core of the business, but I wanna innovate at the edge, could I train up some young kids on the Go Programming language, maybe coming out of school ... What is it and why do you guys like it so much?

Francesc:

[00:27:00] I have two different pitch elevator for you. The first one is for developers, for people that actually program. I'd say that Go is kind of like the soft spot between something as expressive and fun to write as Python, where you just write your thing and basically everything works, plus the performance and security of, I was going to say, C++, but I'm going to go with Java, because I said security.

[00:27:30] So basically you're going to get to a point where you write code that is compiled, like in C++, you get something really fast and probably efficient, plus you get concurrency, which is really important nowadays, 'cause you can't create virtual machines with 64 cores if you're not using [inaudible 00:27:29]. You're wasting money. But at the same time, when you write the code, it is actually very simple. The main goal of the language is simplicity.

[00:28:00]

[00:28:30] The second part of the pitch is, if you're decision-maker, learning Go is super easy, right? The language itself takes, I'd say that maybe will take one day to learn the whole specification and everything that there is in the whole language, which is really good. After a week, you're able to write code easily, and people say that after a month to three months, you're like, a good developer. I'd say it depends on how much code you're writing, and stuff like that but it is true that you learn really fast. And the main point is that through enforcing this simplicity, what you're doing is, you're forcing developers to really go for the most simple solution that works, making your code boring, I would say. I've never seen Go code that I was like, "Whoa, that's so smart. That's awesome," right? While in other languages, I've seen that pretty often.

[00:29:00]

[00:29:30] So now it really goes into a very pragmatic effort on this language to write systems that require things like concurrency, good network stack, and also collaboration across many different people. You're gonna keep it simple, and that's pretty much it. And I think that that's why systems like Kubernetes that we were talking about, it's green and go, it is partly because Kubernetes is going to be running with Docker. Docker was also green and go. But also because you really want something that is able to use all of the cores in the machine, that is able to use them efficiently. So something compiled always works. But also, you wanted to use all of the concurrency and all the networking stack, which is, I'd say, it's really, really good.

[00:30:00] And on top of that, it's open source. For an open source project, you really want people to be able to onboard the project as easily as possible. If you have languages that are for instance very good for networking, like for instance I'm thinking about Erlang. Erlang is a great language for networking. It was designed for that. But learning the language itself is not easy. That's why I think Go is having this really popular phase of basically every single new project ... I have this joke. If it has cloud [inaudible 00:30:04], probably it's green and go, and it's kinda true nowadays.

Bill: So basically, the concurrency you're referring to, is basically that it goes really efficient to be used across multi-processors so to really make great use of hardware, is that one of your points?

Francesc:
[00:30:30] Yeah, you can create a new ... We don't use threads. I mean, there are threads but they are hidden behind them, you don't see them. We use Goroutines, which are like coroutines or like green threads or whatever you want to call them. And you can create easily millions of them and everything works very well. So when you're able to create millions of Goroutines, they are distributed across all of your processors in a very efficient way. You're really using your processors as much as you can, so that is exactly what you want. You want to be able to ... If you have a powerful machine, you should be using it.

[00:31:00]
Mark:
So I'll plug Episode Number 25 of the Google Cloud Platform Podcast. We do go on the Cloud with two of the Go team members, Andrew Durand and Chris Broadfoot. A, it's a hilarious episode, but also we talk a lot about the Go programming language, how you can use Go on Google Cloud Platform, but also the wider implications of Go on the Cloud, just like Francesc is talking about.

Francesc: And also my first time interviewing three Australians at the same time.

Bill: And you're dialing in to this from Australia, is that correct? Or are you in -

[00:31:30]
Mark:
No, I live in San Francisco now, [crosstalk 00:31:31] for the last few years.

Bill:

[00:32:00] Fantastic. Do you see the future of software programming ... If things are getting super simple, then where do you see the direction of programming talent going for the future? Do you see the development of lessening the need for high-end programming talent from a quantitative perspective? Or you're seeing more of a layering in of more creative talent if things are simpler and obfuscated from the code point of view? Does that free them up to do other interesting things? Where do you guys stand on that?

Francesc: I have opinions on this.

Mark: Go.

Francesc:
[00:32:30]

[00:33:00] I think that there's three different things that I see that I'm pretty excited about. One of them is, the languages that are specifically designed for simplicity. Go is one of them. Ruby was one of them too, even though I'd say that Ruby became more into fun, more than simplicity. But the main idea being really enabling developers to create cool stuff, right? And this creates a new kind of developer where they don't really care about how garbage collection works and stuff like that. They don't really want high performance, what they want is to be able to write their code and see it work and being able to deploy it somewhere and everything works, and that's nice, right?

[00:33:30]

[00:34:00] I think that that is really important, because it enables the open source community to grow, which is what I want, after all. On the other side though, I also see languages that are being created with specific domains in mind, so for instance I see Rust or I see, there's another cool language called Last. So Rust was created specifically for when you really care about memory. It is like a safer version of C++, I'd say. And it's really, really interesting. It is not as easy to learn, but it is really powerful for specific things like for instance, I've seen it used in rendering for Mozilla. That's why it was created in the first place. But also, I've seen it used in things like cryptography and stuff like that, where you really, really care about fine-tuning that performance.

[00:34:30] And then other things like Last, where the whole idea is to build a new language that will allow you to write - new [inaudible 00:34:09] systems correctly and I think that is really interesting and where the future is going, where we're going to have languages that allow you to express things at even a higher level. Right now, when you want to do something that is distributed, you really need to think about all of the different protocols of how the different pieces communicate across them. And I think that that will change at some point and instead we will just describe what the system should be doing, from a slightly higher point or level, point of view and just let the language decide how everything works, specifically. But that's what I think.

Bill: I love that. I think that's great and I love that. I love your answer for that. Mark, I don't want to jump in, if you had something to say there.

Mark: No, I think actually Francesc covered that quite well.

[00:35:00]
Bill:
Fantastic. Good, good. Well, what am I not asking that would be super interesting for my audience to hear about Google, about what you guys are developing? Is it true you're allowed to give 10% of your time back to developing new things, as a Google employee?

Francesc: I don't think it's 10. I think it's 20. But yeah.

Bill: 20, okay.

[00:35:30]
Francesc:

[00:36:00]

[00:36:30]
Yeah, we have this rule of the 20% where you're supposed to do whatever we want, but I'd say that as developer advocates, we do whatever we want 100% of the time. So it is true that we personally spend a lot of time learning stuff and writing things that do not necessarily have an impact on what I do. Like we work very closely with the Go team, but lately I've been learning TensorFlow, 'cause I feel like it, right? It is useful and it's very interesting and it also helps conversations that I might have later on. But for engineering, it is true that there is a 20% time, and it is pretty cool, 'cause we actually get, even in our team, we get sometimes people from outside of developer advocacy, helping us as 20%. So people from Software and Reliability Engineering or Software Engineering that want to help us tell better stories, it is super useful. It creates this environment where collaboration is actually required almost, rather than frowned upon.

Mark: I would definitely agree with that. And even in cross-purposes - I mean, I have other teammates who's in DevRel themselves, who either have been doing a little work in the gaming space. But I've had some people who sometimes focus on other communities, maybe language-specific like Python or something like that, but they come and say "I want to do 20% in the gaming space, so if there's a project or something I can work on, please let me know." Or I'll go looking as well. So there's a lot of cross-collaboration there, which is quite nice.

[00:37:00]
Bill:
So you guys are involved in a lot of different efforts and it kind of seems like you spin from gaming platform to talking about things that we were just referring to. So it must be a very, very interesting place to work.

Francesc: Oh yeah.

Mark: It is highly distracting.

Bill: Well, is there anything as we get wrapped up, anything that I missed asking you guys that you just were super wanting me to make sure I asked, or anything that you think would add value to my audience, to wrap things up?

[00:37:30]
Mark:
So I will mention one thing. So we have Episode 100 of the Google Cloud Platform Podcast coming up, we have the distinct pleasure of interviewing Vint Cerf, who is one of the founders of literally the internet, if you're not familiar with his name -

Bill: That's an amazing guest to have on.

Mark:

[00:38:00] Also a very snappy dresser. A very snappy dresser. But that's not that important. We've got a hashtag running on Twitter called askvint, so if you have particular questions that you want us to ask him, please make sure to reach out to @GCPPodcast on Twitter, with the #askvint. We get questions, we're definitely collating stuff from the audience, because I think it's going to be a hugely interesting but also entertaining conversation.

Bill: Wow. That's amazing. I can't wait to listen to that myself. Francesc, anything for you, yourself?

Francesc:

[00:38:30] Not really. Just to add that most of the things that we've mentioned today, there's episodes, full episodes on the Google Cloud Platform Podcast about them. So like if you're interested about Kubernetes, it's now one episode. It's basically half of the podcast, really. We also have Spinnaker and all of the TensorFlow things that we've been mentioning. Just go check it out. And hopefully people will send us as many questions as they have, if there's anything that we forgot to mention, 'cause right now I cannot think of anything.

Bill:

[00:39:00] Would you prefer people - 'cause I'm gonna link up all of the show notes and all of the things we've mentioned, for you guys in particular, if there's people who want to reach out to you, would you prefer Twitter as a way for them to connect?

Francesc: Should we do the -

Mark: Oh my God.

Francesc: How to get in touch with us?

Mark: Bill, if you're happy for us to go through the spiel -

Bill: Yeah, go ahead.

Mark: All right, let's do it Francesc.

Francesc: It's like one minute, but we're highly trained at this. Okay, so we have a website -

Mark: @GCPpodcast.com.

Francesc: We have an email -

Mark: Hello@GCPpodcast.com.

Francesc: We're on Twitter -

Mark: @GCPpodcast.

Francesc: On Reddit -

Mark: /GCP podcast.

Francesc: Google + -

Mark: + GCP Podcast.

Francesc: And finally on Slack.

[00:39:30]
Mark:
And you'll find us in the # podcast channel on bit.ly/GCP-slack.

Bill: That's fantastic.

Francesc: As you see, we've trained at this.

Bill: You are well-trained at that. That's great. Well guys, I've really enjoyed this conversation and this has been enlightening. I've had a lot of fun. So until next time, thanks again.

[00:40:00]
Francesc:
Thank you.

Mark: Thank you so much for having us.

Bill: We'll see you soon. Thank you.