Open Source: The Nerd Version of Formula One

Transcript

Jorge Castro:

The software here is so important. There's someone out there who’s going to tell you that no matter what, the Linux kernel will be taken care of, because humans - there will be zombies tomorrow if the Linux kernel didn't work right. We could also do it in a way that's really fun and good for people.

Katherine Druckman:

Hi, welcome to the Open at Intel podcast. I'm Katherine Druckman, an open source evangelist here at Intel. My colleague Chris Norman and I went cloud native with Jorge Castro, a developer advocate at the Cloud Native Computing Foundation. We completely nerded out on taking the desktop ‘cloud native’, open source, community, sustainability and more. Enjoy, and please join us again for more important open source conversations. You can find more from the team at Open.Intel at open.intel.com, and at @openatintel on Twitter.

Hey, so I'm talking today to my favorite coworker. His name is Chris Norman

Chris Norman:

Hello!

Katherine Druckman:

...and he's very cloudy, and we are joined by our new friend, Jorge Castro. And if you are working anywhere in the Cloud Native landscape, if you're working with Kubernetes, if you've even heard of containers, you probably know who Jorge is. I'm going to say that. So, I'm very excited about this conversation for a lot of reasons, but one of them is because we have mutual friends who happen to be some of my favorite people. So, I have very high hopes and so I think everyone listening is going to enjoy this.

Jorge Castro:

I'll try not to let you down.

Katherine Druckman:

I know and we are going to nerd out. This is gonna be a really good episode.

Jorge Castro:

I cannot wait.

Katherine Druckman:

I've built it up now...

Jorge Castro:

We're going to talk about Linux for like 6 hours.

Katherine Druckman:

I think we should.

Jorge Castro:

Is that what we're doing today?

Katherine Druckman:

Actually, yeah, we are. Clear your calendar, everyone. OK, first of all, just introduce yourself in just a minute, so that people can get a picture, in case... For that one person who's listening who doesn't know who you are.

Jorge Castro:

So, I'm Jorge Castro. I am a community manager currently finishing up a sabbatical. And I've been fortunate enough to work in open source for a long time. I've worked on Ubuntu. I've worked on Kubernetes, Kubeflow and Cloud Custodian and a bunch of affiliated projects around the CNCF Cloud Native landscape, I guess.

Katherine Druckman:

That enormous diagram that we've all seen.

Jorge Castro:

And yeah,

Katherine Druckman:

Never ending landscape.

Jorge Castro:

...and just kind of doing what everyone else is doing, and seeing the explosion of open source, and doing what you can to help move that forward - ensure that the next generation is all set to go and just enjoy the technology that we all get to play with every day.

Katherine Druckman:

Can you believe this is our job?

Jorge Castro:

I know

Katherine Druckman:

I love what you said about the next generation and and we'll definitely get to that. But the one thing that I wanted to make sure we get in there is a conversation about an immutable Linux distribution that you work on. Commonly known as immutable (we can talk about that word later)...

Jorge Castro:

Naming things is so hard.

Katherine Druckman:

I know, right? It really is. Naming things is maybe the hardest thing in engineering, but it is called Universal Blue. You can find it at Ublue.it, which is, I think, charming. “You blew it!”

Jorge Castro:

Yeah, you're supposed to point at your laptop and be like “it's time to you blew it!”

Katherine Druckman:

There you go. So, tell us a little bit about it and how does all this work? What's cool about this?

Jorge Castro:

So, to back it up a little bit, I have always been into desktop Linux and all that kind of stuff. I've always used GNOME, and Ubuntu back in the day. I was very fortunate to meet a lot of people that worked in client and things like that and going through... before 2010, when people were investing in the Linux desktop earlier, almost over 20 years ago. And then I took a detour to cloudland. And always still kind of used a Linux desktop and things like that, but in cloud, I've always found it interesting where cloud dominates the computing.... Linux just dominates the computing industry, right? You can't have a modern world with mobile and all that stuff unless there's apps running to do all that stuff. I've always thought it was kind of interesting that client has never really quite seen that success outside of mobile and a few use cases by a vendor. And I've always thought it was very strange, because it's the same powerful technology, and it just never quite got there, and this has always bothered me. And I was at KubeCon in Detroit and I was meeting up with Colin Walters, who works on Fedora's CoreOS team. And he had been a Debian developer for a long time. And for a long time, he's working on a technology called OSTree, which is a git-like structure where you shove the OS in there. That's going to be the extent of my technical knowledge. I'm a community manager. And he talked to me and was like, “Hey.” And it was always weird, right? It was like that weird thing that you didn't understand, but you're positive it was useful for someone, somewhere. And when we had breakfast, he was like, “I shoved the whole operating system in there.”

“Like, what do you mean?”

“I mean, I shoved the whole OSTree and everything into an OCI container, like a Docker container.”

I was like. “Can you... could you repeat what you mean?”

And in the server and cloud use case, it immediately became apparent, and I think that that community is going to run with it. It's the concept of being able to derive your operating system using common OCI tools that you use today. So, imagine if you were to make your perfect server operating system. If you could say “from CoreOS”, and then do all the things in the Docker file for things that you want on that operating system, including using all the existing tools that you have. And then at the end, you type podman build, and then an OCI container comes out. Then you push that to a registry. And then you boot the metal off of it. And then you have the same model that Cloud developers are using to build applications that they deploy on Kubernetes and all of the stuff that's happening, but now we can do that at the operating system level, in a way that's consumable, with tools that have been around - Kubernetes celebrated its 10th birthday yesterday. So, I immediately I was like, well, the server nerds are gonna love this and having ties with the Kubernetes folks, SIG cluster life cycle, Cluster API folks, as they started to see this, everyone was kind of like...” Yeah. I could totally see”. I was just talking to someone with a vendor who was like, “Well, sometimes we get Intel NICs that are not supported in the kernel yet, and we might have to do these little tweaks and things like that.” I was like, “Well, what if you could just from your existing distro, do the business that you need to do to get it to work, and as long as the image builds, you know, the machine is going to work. Could that actually work?” Before we left, Colin leaned over. “I don't have time to work on any of this, but I'm running my entire laptop out of GitHub.” Because GitHub offers Git hosting, actions and a registry. So, I sat there, and I thought about it for a little bit and I said, “Could this actually work? What if we could take a bunch of SRE nerds who aren't distro folks? What if I can make my perfect desktop? Right, but build it in the same way that I'm building and deploying my apps on Kubernetes and stuff. It would solve a lot of problems, I think. But at the time I didn't think about that, because I'm a nerd and I went immediately to my home lab. I've got a stack of NUCs just like everybody else, and I got to work and I started to shop it around with friends. It's important - I want to mention that many of them do not run Linux desktops. They're just cloud nerds, and because the tooling is common, I didn't need to convince them to go install Linux on their laptop, they could just help me script out some things. So, what we do is we grab Fedora which publishes their images now as OCI in test. This isn't in production yet for them - they're going to do this, probably in Fedora 39, which is next cycle. And I said, “Couldn't we just ingest all of these images, put whatever we want on them, and then just give people what they want?” A little kit for you to make your own thing, similar to the first time you learn cloud, when someone says, “Hey, you need to deploy this stack.” - You went looking for that set of YAML files, so you could ‘Docker compose up’ or whatever it is that you want to do. We could do that for client. And it ends up that not only can you do it, but it works really great. So, we ingest everything from Fedora for versions 37 and 38. And then people started to show up and said, “hey, this is neat, but I like KDE and not GNOME.” And in cloud you make your little build matrix. We just picked a different set of packages. And I'm kind of relearning how to write multi stage Docker files and things like that. While we're sitting there building it, we're not learning distribution specific tools. But we're just reusing our common cloud language because making distributions is hard, right, like at Ubuntu? You know, getting a change into distro, there's a lot of engineering and things that happen. And I didn't really want to make a distro.

Chris Norman:

There's a lot of challenges that come with integration and a lot of testing that you have to do, to make sure that all of the components work together, right?

Jorge Castro:

Right. And you don't want to fall into that trap. We're like, “Well, if we know Fedora is going to do this, but the feature isn't ready, but we can prototype now, wouldn't it be neat?” Could this actually work? So, we did. We made KDE, GNOME. We made a few tiling window manager ones. We started to find all sorts of things in the Fedora Archive where people were saying “I want this cool thing. I want this cool thing.” And usually in Linux, when you're setting things up - Hardware acceleration on my Intel 2-in-1 - you go and you see instructions for how to enable that in your Linux and it's always a manual step. And on the other hand, you see how people just go to Best Buy and buy a Chromebook. They don't have to set up any of that stuff. So, I said, “Couldn't we just grab all these web pages and just shove them in container files? The instructions to enable this? And see what comes out the other end.” And what came out the other end was a really nice operating system that just works and just boots because it's image based. And then we get to remove a lot of the complexity on the client for upgrades, dist upgrades, adding PPA's, having to do ‘dpkg-reconfigure –a'. The RPM fusion thing was built for a module for the kernel that's a day old, so I have to wait for the build system in the distro to catch it. And all these kinds of things which are still problems, but now they happen in our CI, right? And then we can catch them, but the end user always gets an image that works and all of a sudden, they have the granularity to go back in time, boot off the image. You're not doing a snapshot/backup/restore. It's literally booting off an image, so it's clean.

Chris Norman:

This is the premise of Clear Linux - that we do all the integration at compile time so that you don't have to worry about the distribution hell of dependencies.

Jorge Castro:

Right. That's something I did learn from Tim Pepper, who had worked on Clear Linux. I started to shop around the idea because I wanted to do it right. I've learned enough, and I've learned to take advantage of all the senior people that you know and ask them for advice. You know something he had always told me was Clear Linux was designed to do a bunch of work client side and then you want the package manager to just splat the disk and not do the other stuff. But however, in kind of the traditional Linux land it's all about packages right in your package manager, and that's what you're picking, and you're supposed to know this stuff. But as I started to realize, as I was doing this as an advanced user who knew all this stuff, I don't want to do it either...

Chris Norman:

It's like starting a car. You just want to turn the key and have the car go. You don't want to be putting a different set of wheels on every time you want to go to the store, right?

Jorge Castro:

I've done all of that already. I don't have to prove it to anything. You know, I was like, “Why? Why would I compile my own kernel, when you know Colin King?” However, like in traditional Linux, there's this kind of, "Uh oh, but we're supposed to be about packages and and doing all that stuff”, which I found was very interesting - that the Linux desktop culture is very much entrenched in the package, right? Meanwhile, I'm hanging out in cloud land, and all the best Linux people that I look up to, that are just absolute experts in this stuff, and they're all running Macs, right? Because they decided that getting that work done - they didn't have time for the Linux desktop, right?

Chris Norman:

Yeah, that needs to be ‘Someone Else's Problem’.

Jorge Castro:

So, I was like, wait a minute, “Why don't we tackle it this way then?” That's when I really started to think about the economics. Because, if you look at just e-waste - I live in Michigan. I live in Ann Arbor. The University of Michigan is here, and they have a property disposition. You can get used computers and stuff, and you go there and it's just pallets and pallets of computers. And you start to have that nerd thinking, “Well, if I put my image on that thing, I know it could do something. I know it could be something great.” And we know that people can use Linux because they bought Chromebooks. And the great thing about Chromebooks is they purposely don't mention Linux at all. It's invisible.

Katherine Druckman:

Yeah, it's irrelevant to the Chromebook user, yeah.

Jorge Castro:

Right. And I think a lot of people get upset by that. Because no, I want everyone to know that this is Linux.

Chris Norman:

Steam Deck’s the same story, right? It's made gaming on Linux a thing, even though people don't realize they’re doing it.

Jorge Castro:

Right.

Chris Norman:

I have my Steam account. I boot it up on the Steam Deck and it just works.

Katherine Druckman:

No, no one cares really what the solution is as long as the problem is solved. At least, for that user...

Jorge Castro:

So, if I can get you the reliability, on a Chromebook, of a Chromebook, that can run on way more hardware, because it's a generic Linux distribution. And Fedora’s been around forever. You could get just about anything on it. It comes with a container runtime, so anything that's not natively packaged, you're gonna pull from your registry anyway. Could this actually work? And then I've also thought “Well, the cloud model, like a lot of these - the infrastructure for this stuff is already being invested in.” The registries are on edge already. And if we can make the Linux desktop payload just be another cloud payload, could we bring the distribution costs down for people so that Linux desktop can have a shot at being useful for someone?

So, that's where I started to think. If we can make this cheaper for people, if OEM's that ship Linux can use this model... Can you imagine if you had a bug with your wireless driver and you go to the bug tracker to report it, and we just have the bug trackers just generate - the engineer has a patch. He thinks it might work. What if it was just a bootable image? Right now, I file an issue and it's “Jorge, can you bisect your....?" But if you got me an image, I can tell you whether it works or not, because that's how app developers and cloud work. What if we could bring that to the operating system? And then you start to think of all of these things.

I'll give you an example - the out of tree on new Intel cameras. Someone in Fedora enabled those and then they put it in RPM fusion and things like that, but I can't get those in Fedora without following the instructions and things like that. Well, what if you could just have that on the image? And then the manufacturer and the OEM, they could figure that out. But I am on the image. We could slipstream that in. What if we can bring that kind of tight feedback loop between OEMs, the customer and the distribution?

So, then we started to think about, well, what we're doing here is taking the distribution model and making it more "continuous delivery", which the distributions are already doing. But what if you go all the way to the end, to the user? So that's when I started to think about, if distributions were more around this model, could that save a bunch of people a bunch of money? I don't know, but I know that the current model isn't working.

Chris Norman:

I saw a news article the other day about a lot of the Chromebooks that schools have been buying at the beginning of COVID are now reaching their 'end of life' time frame now. So, you know this is a great example - something that would never been an issue if we'd gone down this road.

Jorge Castro:

Right, right. And the great thing about Chromebooks is they have a set use case. It is a reliable thing that you don't have to care about, that has a browser and that's enough for just about everyone. And then on the admin side, that school IT admin needs a little GUI. But we have tools like cockpit which I include in some of our images which is the web UI that you can use to maintain stuff. A student could write you a plugin to help you manage that stuff. So now we start to think about, "What devices and hardware do schools have, do NGO's have, and can we give them that model, but using general distributions that allow people to do what they want, without the lockdown of a specific vendor.

Chris Norman:

So, you've got more than a proof of concept, right? It's a working implementation. Where are you taking it?

Jorge Castro:

I have no idea! What we're doing now is... so Fedora is... we're duct taping stuff. It's good duct tape, but at the end of the day, we're kind of just taking where Fedora wants to go and seeing, maybe where it needs to go. What we have now is a good working concept of what a next generation Linux desktop would look like. And you mentioned the word 'immutable'. I used the term 'image based'. I think the market has determined… - I like to tell people, when people say "Well, I don't know immutable desktops sound kind of controversial and are you going to change people's minds?"

Katherine Druckman:

I wonder the same.

Jorge Castro:

The market decided. There's exactly zero non-immutable Linuxes that are successful in the market. Because you can't ship one, and why would you want to support that?

Katherine Druckman:

Right, right. Yeah.

Jorge Castro:

And people are doing that. But I'm a developer, so I will buy a developer laptop from an OEM that has traditional Fedora or system 76 or whatever. That does exist. But what if we can bring that reliability of the Chromebook. And you had mentioned Clear Linux, a lot of the ideas that I started to come up with was Clear Linux just says we're gonna do automatic updates. And if you go to any traditional Linux distribution and say, should we do automatic updates, they're gonna argue about it for two years, and then at some point, someone says, "You know what? I've never seen my dad talk about updates for his Chromebook, or anything."

Chris Norman:

That happens behind the scenes.

Jorge Castro:

He just uses his computer, right?

Chris Norman:

You just use it. Just switch it on, yeah.

Jorge Castro:

So, you cover that use case, but then can you do the advanced use case? Because in my brain I'm always like, well, if I have a container file that I could do [with] whatever I want, I'm gonna make the world's greatest Unix workstation for myself. So, I started to add my cool stuff. And you all announced a new font, Intel [One] Mono. I copied [it] into my git repo, and the next day it was on every single one of my machines, and all of my friends' machines, because we made a developer image with Visual Studio code and it's got home brew on it and it's got the latest version of Python and Go and we're just nerding out.

Katherine Druckman:

Well, that's pretty cool.

Jorge Castro:

And it lets people play and experiment, but they're doing it in an image-based workflow where you can experiment without accidentally putting yourself in a position where your computer doesn't boot, which is a problem with traditional Linuxes. Nobody wants to pay that tax anymore, I don't think.

Katherine Druckman:

I love that you mentioned that actually, because I think there's a gut rate response when you talk about immutable distros, which to be fair, I've only really started paying attention to this sort of ecosystem, if you want to call it, in the last less than a year, but I think there is an initial [reaction] like “But wait, don't nerds like to tinker?” But of course, you can, right? It doesn't actually mean you can't change things. It means something else. Right?

Jorge Castro:

Yeah. I'm tinkering in CI and I'm learning. What's really great is - I've had people say “I don't know. I like to tinker live and stuff.” But then when they start contributing to the project...

Chris Norman:

But that's where you do the prototyping right. This is what I want it to look like. OK, now I put it in the CI so that it's automatically pushed down, right?

Jorge Castro:

Right, right or what I do is... I just get too lazy and I just do it directly in GitHub and then if the tests pass then I know I'll get it the next day on my system.

Chris Norman:

So, for something like installing a font, that's a very low risk change, right? I just wrote up how to do that for Clear Linux on the forums yesterday, but if you made a more major change that would need testing on actual devices, do you have a Canary channel that you use for people to do more risky changes?

Jorge Castro:

Yeah, so right now our tag structure, which is just standard container tags. We do like Fedora:38, :37. So you could do that. We do have a :latest because even though it's not really best... I don't want to get into that best practice, but right now it's not ready and I'm more scared about leaving people behind on an old version. But you know, once Fedora actually blesses this as stable, what I do want to offer is a more diverse set of tags that lets people ‘control the throttle’, I call it. And with our NVIDIA images, especially, because we have to compile those certain drivers for that kernel, and there's a series of drivers. That matrix is actually much more complicated but does solve a lot of problems that people have. You're not compiling the GPU driver on your laptop, and waiting for that process to happen, which if you've used Linux for a while, you don't want to be there. Ideally, what you want is to use a GPU that has drivers in the kernel, and then you just always get the goodness.

Chris Norman:

It just works.

Jorge Castro:

But, unfortunately, people do have that stuff. So, in my view, having those drivers on the image itself, similar to those camera drivers - it gets people working hardware. It's not ideal, but it gets people working hardware. In our brains, we're always thinking, “Wouldn’t it be neat if we had a test suite and labs and conform...” and all this stuff that you talk about when we are talking about CNCF projects. No, not currently. We're just enthusiasts doing the thing. So that that's kind of where we were going to "let's get the operating system out of the way. Let's make that smooth. Let's reuse as much existing technology that is funded, because the reason OSTree and and all this CoreOS stuff is happening is because people buy OpenShift." You know, let's not kid ourselves here. But I want the stuff for the desktop as well so if I can piggyback on that... That's the first part.

The second part ties into what I consider the biggest problem in open source today, and that's the economics of maintaining our open source projects and our people. So regardless of what people say about flat hub, flat packs. You mention that in a group of engineers and you're going to have people talk for three days this way and that way. And to that, I'm just responding with, "Flat Hub puts a button to pay that open source developer on the UI." And whatever problems you might have with flat packs, we can probably fix it. Or at least get us to a spot where the priority should be - if you're using a system like this and you see a cool app and it's an open source app and you click a button and you can send that person some money to help us bootstrap and maintain a healthy ecosystem of open source and you have to have proprietary software, support is going to have to come along too because you have to do both. Would that be what we needed? If someone can have an idea for an app, they can put that on the Apple Store or the Google Store and start making revenue immediately and open source maintainers don't have that. At best you'll find a Patreon link or GitHub sponsors for the people that bothered to go to that page. So, for me, it's like lowering that cost, but then giving open source developers the opportunity that all the other developers have that pick a platform - we don't have a Linux desktop platform. But if you take a free desktop stack, you shove it into Flatpack and see what apps people can make. You could do that and then now not only do we have "A Chromebook model", but now we have an avenue for open source developers to bring those native apps that they love to there, and since the desktop now has a container runtime, you get all the server stuff "for free". So, any of the technologies that you're reading about, or whatever, are now available to you. And then you have a system there - you can get all the browsers. Can this actually work as a way to bootstrap the next generation of people who are going to be the ones maintaining the kernel, Kubernetes, all of these things that kind of run the modern world? And we have to figure that out as a community - how to make that sustainable as a whole. And that doesn't matter what company you work for. We're in the business of ecosystems here, and we all have to take care of the natural resources that are available to us and for that, that's the people, right?

Katherine Druckman:

Yeah, open source is people.

Jorge Castro:

So, could we do that right? So, in my brain after playing with this stuff, I actually believe that. Could we put things that look like Chromebooks, that can do what Chromebooks can do, with more, except all of those students could run a Kubernetes cluster and fire it up. And it would run great because there's no VM there, it's Linux. And it would run great. And if they wanted to learn front end stuff, they have all the npm stuff is there. If they want to learn Python and Machine Learning, they have it. And could that be the way to bring that to end users? And that's a lot of the stuff I'm talking about. Remember the first time someone explained to you what Ubuntu was? It's not just a Linux distribution, but the idea of bringing that technology down to people? But making distros is hard. Now we can leverage people that use the technology every day and for them universal blue is a simple concept. It pulls something from Git. It does podman build on a Cron and then it shoves it into a registry. It's not rocket science. The

NVIDIA Repo is pretty cool because that's doing kind of more sophisticated stuff and the engineers that worked on that... - it's really cool. And there are certainly cool things that we're doing, but at the end of the day, for a Kubernetes admin this is very simple for them. If we can use that expertise to give people better clients and then we could use those clients to get people better access to technology. Will that work? And that's been the question that I've been asking myself when I started taking a break, and it ended up being a sabbatical, because it's been gnawing at me. And not just me. This just happens to be the implementation that I stumbled across. I know lots of people who are working at... Everyone in open source offices all across the industry are thinking about this exact same problem. Linux Foundation, CNCF. Everybody's thinking about how do we get that next generation in? How do we ensure that they have the right tooling, the right training, the right kind of safe spaces? We finally learned that yelling at each other doesn't... you know the first part of open source was a lot of yelling?

Katherine Druckman:

That’s a whole other conversation.

Jorge Castro:

So, could that work? As I started to do this... I have a son. He's six, and I watch him use technology and things like that. And you almost kind of start to believe it.

Katherine Druckman:

I love all of the different angles you're bringing in, on a conversation about sustainability. You're talking about human sustainability, but you're also talking about the green definition. We talked about reusing low end hardware. We talked about all of those things, but you're also talking about the psychological sustainability of being a maintainer in the current environment. I wondered if you could talk a little bit more about the problem as you see it and where it came from. We've all been around open source for a very long time here, and these conversations were different 10 or 15 years ago, or longer. These conversations were different. The world has changed. Obviously, technology changes very quickly, and and that's a given. But it's not just that. It's the way that we interact with it. The way that open source software is made, the culture around it, I feel has changed. It was a little bit scrappier and more DIY and more purist, I think, when I entered the conversation than it is today. And again, the way... Chris and I have talked about the monolithic to microservices [trend], right? Again, everything's very specialized and you're in this tiny little pocket and maybe you're not even aware of the larger ecosystem you're a part of in a way. And I just wonder if you could kind of talk about the way you see especially the cloud native landscape evolving and how it may have contributed to sustainability problems that now you're talking about addressing?

Jorge Castro:

I... wow, that's... Really

Katherine Druckman:

That was a lot, I know.

Chris Norman:

Profound, profound statement.

Katherine Druckman:

But to be fair, you gave us a lot to go on. So, I'm trying to unpack a lot of different threads at once.

Jorge Castro:

I think you're touching on part of the reason I wanted to be on the show, and I've been mailing every YouTuber I know, is.... If there's one thing I think, having gone through the early phases - Ubuntu when it was brown, the early phases and stuff, and the early days of Linux...

Katherine Druckman:

I can’t even remember, by the way, what you said. “Remember the first time somebody...?” I don't remember the first time somebody told me what Ubuntu was for. It was so long ago.

Jorge Castro:

No, I've aged myself.

Katherine Druckman:

It was brown, though I think, yeah.

Jorge Castro:

It was brown. Yeah, … is at first, you're just kind of like, “This thing is so cool. We're gonna take over the world. And we're gonna spend most of our time arguing with people, telling them why this is the best thing ever.” Back in those days, you had a Linux User Group and you would go and you would have, yeah, yeah.

Katherine Druckman:

Yeah, because that was still a conversation that had to be had, before we won.

Jorge Castro:

And then Cloud and Mobile happened. And then I think open source. You almost kind of believed.

Chris Norman:

It's shifted from being its own thing to being just a part of way software is developed now.

Katherine Druckman:

Nobody thinks about it in the same way at all, and even the people working in it, the people making the open source software, they are, in many cases, so far removed from that original conversation because again, this is just the way software is made. It just is.

Chris Norman:

So, we're now seeing a proliferation of internal open source. Projects within corporations that never even see the light of day.

Katherine Druckman:

But I wonder how that contributes to all of this sustainability issue, because again, people are working in their own little worlds. It's a little bit... It's just such a different conversation.

Jorge Castro:

Yeah, and I think initially when people look at technology, they, you know “War!” Everything is a war and you're thinking in that military, “we're going to take over,” and “market share.”

Katherine Druckman:

Like a flame war.

Chris Norman:

Android versus iPhone!

Katherine Druckman:

Vi versus Emacs

Jorge Castro:

Yeah, and everybody wants to know, what's your favorite distro? “Say something bad about Snap!” or whatever it is you want to get out of people. And then it ends up being, you won the cultural war, and you don't know how to... All of a sudden, the questions are totally different. And it becomes more of... What I've learned in my cloud native journey, as you know that's what people call it, is it takes a village. That is the one thing I learned about Kubernetes is - it takes a village. No one person or team can know anything, and things work a lot better when you get all the smart people together and then they figure out what should be commoditized. Because at the same time, we need companies to support open source, and in order to do that, they need to be competitive in the marketplace. So, for them the trick is how much of this stuff can be commoditized (that we want to maintain along with everybody else) but allow us to make the secret sauce that goes on top so we can compete with each other. So, it's co-op-petition in a way, right? But we need to ensure that ecosystem (in this case, I'm gonna use Kubernetes as an example,) Kubernetes needs to be healthy and we want companies to be able to compete in that marketplace. But what we don't want to happen is the people and organizations that are contributing the most doing a bunch of the leg work, and then people that aren't contributing the most being rewarded in the marketplace - because they're not spending the money to invest in open source and we know that we do not like that pattern.

Chris Norman:

We see a lot of friction internally of people not wanting to go down the open source route because they're very concerned that they’re gonna lift all boats. And it's like, well, no, then you make the total available market better, bigger, it's good for everyone, right? But no, we can't help the competition.

Jorge Castro:

A lot of this stuff is organizational and people wise. If I was working in the perfect organization, that doesn't exist, open source internally would be a no brainer. We're going to lift all boats and we're going to rock it, because we're agile and we know how to ship product, and we know how to put stuff together faster than anybody else. But what can happen, what people are afraid of, is we're gonna invest in doing it the right way, which sometimes takes longer because investing and doing open source correctly is the harder road, but it's better long term. However, you do see it when companies are doing it right and then a competitor comes along, doing half as much work, and then gets the bulk of the pie.

Chris Norman:

Rides on your coat tails.

Jorge Castro:

So, in many ways, that's why we need to ensure that the rules around the ecosystem are fair. And that's where I think the challenge is, because that is going to be the crux point of bringing in those new contributors. But they also need to understand when you're wearing your ecosystem hat, and when you're wearing your work hat. And I also think that it took time. And we didn't have time in the early on because we are now getting to the point where, you know, people are cycling out, they're retiring. And before it was the early Linux ([where]you kind of knew everybody) and everybody was in the same place. Now we've gone through that generational set of learning, where we can sit there and say, at some point, all of us are going to end up working for all of the affiliations anyway. So, your relationship as an engineer, as a professional is to be that... Here's how I see it. I've worked for companies that have been friends and competed against each other. And I'm not the only one. But when it comes to the maintenance and health of this project, I'm expected to put the needs of that project first,

Chris Norman:

It's commitment. It’s a long-term commitment, and it's a consistent commitment.

Jorge Castro:

Exactly, and I think what you will find is, in the projects that are run well and are successful, you see that kind of stewardship of those projects, even whatever happens with the affiliations in the background become. That's important to some people, but not to us when our job is to ship Kubernetes that day, or whatever project it is.

Chris Norman:

It's important to the hosting company that they have the right number of people in the right number of projects.

Jorge Castro:

Exactly, and one of the things I'm excited about and sharing more with people is how do we internally (having seen OSPOs work), how do we internally sell as engineering....? I think the engineering aspect is solved. I think you can say open source as far as an engineering principle or whatever, if you're in infrastructure and cloud...

Katherine Druckman:

Sure. Yeah, we won for sure.

Chris Norman:

We know we've won. We have to convince everyone else we won. Yeah, yeah.

Jorge Castro:

Yes, yes, we can. I think we convinced the engineers.

Chris Norman:

Oh, the engineers. Yes, it's upper management and the bean counters.

Jorge Castro:

The bean counters, is there... For example, we could be investing in the open source. But that is a long-term investment and sometimes, due to the way companies are structured, they're kind of just caring about that quarter. And it becomes an investment of open source and a long-term thing that is hard for organizations to ingest and justify, unless you have that organizational knowledge. Or you've been burned in the past, where you've learned, maybe, it is important for us to invest. I mean, I don't have to convince Intel that working on the Linux kernel is a good idea....

Katherine Druckman:

We are very big believers.

Jorge Castro:

...from an engineering perspective, you know. But not everybody has gone through that institutional knowledge where it takes you to know, “hey, we're shipping hardware. Software is an important part of that now, you know?

Chris Norman:

Without software it's just sand.

Katherine Druckman:

Oh, I like it, yeah.

Jorge Castro:

And that's what actually keeps me up at night, you know?

You convince the engineers, but can you convince the bean counters that when they go to KubeCon and they see all this stuff, are they going back and saying, “It's a good thing that we didn't have to do this for ourselves, you should see!” You know, in a way, open source is like an extra IT team that works for your business.

Katherine Druckman:

Oh, absolutely, yeah.

Jorge Castro:

Do they see us that way?

Katherine Druckman:

How much burden is relieved?

Jorge Castro:

Right.

Katherine Druckman:

Right. This upstream contribution, that's such an important piece that people need to understand.

Jorge Castro:

Right.

Katherine Druckman:

So, a couple things, so many things.

Chris Norman:

Get us back on track!

Katherine Druckman:

There are so many things here. OK, first of all, I want to, just.... So, one of my smartest friends (thanks, Kyle) tells me, “Oh, the work that Jorge is doing right now, this is the future of Linux.” So exciting, right? So, is it? Or do you see this as part of the landscape that's evolving? Or, it's just a section? Or is this the way we're going to do stuff?

Jorge Castro:

I think, on client, this is the present. And I want to be clear that I didn't invent any of this. A lot of organizations are working on this. I'm using Fedora Silver Blue, as an example, because that happens to be the tech that I got comfortable with. And by them choosing that Cloud Native approach made it very easy. OpenSuse's doing a similar Cloud Native model, but they're using BTRFS snapshots. Clear Linux as we've talked about before... Everything that I am talking about here can apply to that, other than the OCI thing, but you're smart... It's more “Can we talk about the model?” I think, if I were to go to SCALE and give a talk about this, it's just the future of the Linux desktop. I don't think that's the right way to look at it. I look at it as,“This is the present.” Chromebooks have been out for a very long time. It is obvious to me that the market has chosen to not choose traditional Linux. But the model is here. The technology is here. We know Linux is capable of doing these things. Hardware’s so much better now. Not only can I buy a laptop that works with Linux, I could choose one from multiple vendors, with multiple components. And a lot of those companies have great Linux stories and are doing the work. We're nerds, so we're always kind of well, “Except that one model number!” So, we know that the hardware is there. That was the hard part. We had seen - remember netbooks, early, pre-iPads?

Katherine Druckman:

Those are so great.

Jorge Castro:

And the hardware is just so much better now. But the software element was missing, I feel, to get us to that level of reliability. The users are demanding this. I think a lot of this is just cultural Linuxisms that we have to get over. We just have to get over it and we have to figure out how to get people there. And part of the reason I am doing the project is - If you had to learn how to set all this up by yourself, you'd give up! But now that I have the ability to push this into Git in a container file, this opens this up for a lot more people. I'm trying it. We decided we were going to try it because at first it was "This is going to be a passion project"- have set expectations of what it's going to be. But you see someone struggle getting an NVIDIA driver working on their on their laptop, and this is the only thing they have. And if we can provide value for them, and if we could provide value for Fedora to show them that their investment in this technology (even though they started it off for server and cloud), if we make it all the same and it's all edge, you can sort of solve another problem also and it's neat.

Katherine Druckman:

I mean, as a developer, a developer in a former life, that part of it is so appealing. The consistency element. Yeah. So, before we've been going almost 50 minutes. Which is awesome.

Jorge Castro:

Yeah, I told you, you were gonna get the Jorge Castro experience.

Katherine Druckman:

I'm not editing anything. It's all going in. This is gold right here.

Jorge Castro:

Did Kyle tell you how much I talk?

Katherine Druckman:

Nerdy, delicious gold.

Katherine Druckman:

But I did want to...

Chris Norman:

We have enough for 3 or 4 podcasts here!

Katherine Druckman:

Right. Yeah, but...

Jorge Castro:

Slice it, dice it.

Katherine Druckman:

I want to get to the Cloud Native, the CNCF thing.

Jorge Castro:

Oh, yeah, yeah, yeah. Let's talk about that, yeah.

Katherine Druckman:

So, we're participants, we're big fans of the CNCF around here. Tell us about that organization and tell us where you fit in.

Jorge Castro:

So, I'm hoping to start here next Monday, actually, my sabbatical is coming to an end. Universal Blue is a great project and it's sustainable now because it was designed by SREs. So...

Katherine Druckman:

Always a good sign.

Jorge Castro:

I'm hoping to change 2 numbers every six months and call it a day. But as I've been doing this, it has exercised those muscles to think about this kind of stuff, a lot, and it also reminded me that we need to copy and paste the model across more organizations. So, one of the great things about the CNCF is - there's a lot of stuff there. And one of the bad things about the CNCF landscape is - there's a lot of stuff there. And you don't write... Kubernetes is kind of like the operating system-ish. I guess the center, and you have all the other projects like...

Chris Norman:

The engine that drives it, yeah.

Jorge Castro:

...and a lot of the stuff runs on top of Kubernetes, right? So, how do we take those processes and use that in as many projects as possible? Because Kubernetes was large, it was important. By the time I got there, it was already successful. There was no day where it was, "Wow! We were gonna fail until Jorge showed up like that.” That never happened - a problem, right? But I have worked in smaller CNCF projects like Cloud Custodian, where it's mostly a single maintainer, Kapil, just trying to do a day job, be a CTO at a start up, and run an open source project on the side. And then we've been stuck in incubation forever, or sandbox forever, and we're trying to move to incubation and you're filling out all the paperwork, but unlike Kubernetes, you don't have 6 full time community managers helping you out. And then you start to realize, wow, for all this talk about code reuse and how smart we are in cloud, here I am, doing it the hard way. So that's a problem I want to fix. There's just templates. Even something as simple as - do you have all the information in your readme for your project? One of the things I've learned is at the bottom of your readme, put the link to your mailing list, your channel. And you talk to a lot of people and they get this. They know this. But then you also end up seeing something really cool and you have no idea where to go from there. And you kind of wish...Oh, and then you run into the other problem with open source maintenance. Oh, I'd love to help them out and just do all this stuff. But I've got my own problems. The other issue I think we have generationally is, at first it was so simple. We have Linux, Apache and MySQL and PHP and Python done! Now, not only do we have tools, we have multiple copies of tools and then every time there's a new language, those tools double again. Now we have everything in rust. The amount, the explosion, you can't put that genie back in the bottle. So now it becomes a people problem again. And how do you run an open source project? How do you handle your first PR where you can tell that someone didn't read the documentation and your project isn't really going in that direction, but now you have that awkward conversation? If I would have just written that down in my contributor guide, set expectations, I wouldn't be in this mess. Right. I know those lessons. And I keep making the same mistakes over and over again because I'm a human being. So those are the conversations that I think maintainers should be having amongst themselves and it needs to be in a place that can be the honest place, where we're talking about open source and isn't where the bean counters are around like we have. We have to do the nerd Community home lab. at that point. No embarrassment. I had to do something really bad to get this thing to work. Look at how horrible this is. Isn't it amazing? You have to have those safe conversations in an environment where you can tell another open source maintainer, "Yeah, I tried to do this with that toolkit. Here's what I've learned." And having those kind of shared best practices around, not just the engineering but... At first when I was doing the project, I was all about getting eyeballs on it, right? As I was struggling, you're doing the open source thing and you're struggling. I need to get more eyeballs on this. And my friend Marco was like, if you had 30,000 eyeballs right now, this would be a disaster because we haven't written down the docs yet. How many times have you heard write docs? All the time, right? But you just get so caught up in the one user shows up "My laptop didn't work and finally got it to work. Thank you." And you get so caught up...

Chris Norman:

Everyone swarms in.

Jorge Castro:

...that you, you kind of forget to do the 'eating your vegetables'. You've got to eat your vegetables. So, now that I've kind of gone through that on my side project, and one of the nice things about doing it in a side project is you know you can make mistakes and not be in trouble at work because you made a mistake. And that's where I kind of started to think, could we bring these kinds of lessons that people are having? And how can I help connect those maintainers to each other in a way that helps them understand, "Hey, we're having we're having an issue with this bit of thing that we have to set up for our project. What software do we use to enable our communities. Are you picking Discourse? Are you going with mailing list?" You could have podcasts for days on this kind of tooling.

Katherine Druckman:

Yeah, we could. We should actually.

Jorge Castro:

And you can go on the Internet and get all sorts of stuff, but it could also be more efficient if you're talking to people who are maintainers, just like you are, that have gone through that process. And it kind of filter out... well, what, do my other compatriots that are in the field doing? I can watch an F1 race and figure out how they're driving. But the conversations that the drivers have between each other, and how they take turn 6, or whatever, is totally different from what us fans see. So, right, so we want open source maintainers to be able to have those high bandwidth, low noise conversations that are about tooling or a bit of open source governance.

Katherine Druckman:

I love that analogy.

Jorge Castro:

We didn't even touch on governance. That is an entire sphere of things. So how do I, knowing that it takes a village, how do we get all of this expertise for these different areas, around a common place where they can talk about it and it's no stress about... Well, there's always going to be stress. But there's something to be said where it's like “I'm kind of struggling with this part in my project. Here's a room of other people struggling. Let's hug it out.” And as I think as we've seen so far, the village is doing a pretty good job. As self-critical as we are about the issues that we have, about supply chain security and all of that, we should also be proud of the kind of anthropological successes that open source has had as a group of people that are trying to fix - modern society is complicated. You know that the level of technology that we're using is incredible, compared to just a few short years ago, so...

That's where I'm at. You know, I want more and I want to make it faster.

Katherine Druckman:

Everything that you've said is basically a reminder that software is made by people and there's this human element and we can talk all day long about the technical thing, but it's not actually, yet, made by machines.   I mean it will be, but you still are going to have people somewhere in the equation. And there's all that human work, and all of that stuff, that goes into it. And everything, again, everything we've talked about for the past hour goes back to the fact that it's the human work. It's the human conversation. It's the human interaction. It's the human documentation. It's all of these things that go into these projects and I think what I hear you saying is that, you see the role of, for example, the CNCF is really trying to help fix the “problem with open source”, that isn't really a problem with open source, but...

Chris Norman:

“to grease the wheels”

Katherine Druckman:

Yes, there is a a human element that needs to be addressed, that maybe an organization like that is empowered or at least positioned well to...

Jorge Castro:

Yeah, and one last point. The software here is so important. There's someone out there who is going to tell you no matter what, the Linux kernel will be taken care of, because you know... human society... -there be zombies tomorrow if the Linux kernel didn't work.

Katherine Druckman:

Yup.

Jorge Castro:

But we could also do it in a way that's really fun and good for people. If it was,“I work on a thing that is important, but it's not a lot of fun.” You wouldn't want to do that. I think you want people to be excited about the stuff that they work on and not have... Having gone through that open source burnout that I know we didn't talk about this, but we know people struggle with.

Katherine Druckman:

Sure

Jorge Castro:

And fixing those problems and understanding them outside of an engineering aspect is, I think, a point that we should talk about in a future episode because that is an entirely different thing.

Katherine Druckman:

The title “The Human Problem”.

Jorge Castro:

Yeah, yes, it's not good enough that the stuff is maintained. Those people need to feel like they're doing something awesome.

Speaker

Know they’re needed.

Jorge Castro:

It's a nerd version of F1. Show me the spectacle of open source development, that's what I want to see.

Chris Norman:

There's the tagline for the podcast. Katherine. Yeah, the nerd version of F1.

Katherine Druckman:

Oh, I love it. Oh, this has been so good. We're going to have to do a follow up, for sure. We have barely scratched the surface, I want to say, on all of the wonderful things that we just brought up. And I have so much to think about, and so much to talk about, I would love to have this conversation with many people and probably you again. And I thank you very much, Jorge, for being here and thank you so much, Chris, for joining me and having fun

Chris Norman:

It's been a lot of fun.

Katherine Druckman:

and any excuse to hang out with Chris virtually is good in my book. So, thank you and thank you everyone for listening. If you have any feedback, where can people find you?

Jorge Castro:

I'm @castrojo, on Twitter and @Jorge@Hachyderm.io on Mastodon. I'll give you all the links and stuff later, and I always will talk about this stuff. If you want to do that or you're a KubeCon, I'll be hitting the road here going to DevOps days and things like that. If you want to talk about this stuff and you're passionate about it, I need your help. We all need each other's help so.... Thank you! And try Universal Blue! It's pretty cool.

Chris Norman:

There it is. Yep.

Katherine Druckman:

Keep the conversation going.