EPISODE 721

[INTRODUCTION]

[0:00:00.3] JM: Not every company wants to move to the public cloud. Some companies have already built data centers and can continue to operate their business with their own servers. Some companies have compliance issues with the public cloud and want to operate their own servers to avoid legal risk. Operating a data center is not easy.

Operating systems need to be updated and security vulnerabilities need to be patched. Servers fail and their workloads need to be automatically scheduled onto other servers to avoid downtime. In contrast to classic on-prem data center management, the cloud provides many benefits; automatic updates, an infinite pool of resources, fully programmable infrastructure as code.

In the cloud, developers can provision infrastructure with an API request. Continuous delivery pipelines can be spun up at the click of a button. This tooling makes it dramatically easier for developers to move quickly and for a business to move faster. Companies that operate their own data center want to be able to have these benefits of the cloud while still controlling their own infrastructure.

Today's guest is Bob Fraser. He works at HPE on OneView, a tool for managing on-premise infrastructure like a cloud. Bob describes the difficulties of managing legacy on-prem infrastructure and the advantages of building a management layer on top of data center infrastructure to make it more programmable. We've done lots of shows recently about Kubernetes in the context of cloud computing. Today's show outlines how modern on-prem infrastructure can be managed like a cloud.

Full disclosure, HPE is a sponsor of Software Engineering Daily.

[SPONSOR MESSAGE]

© 2018 Software Engineering Daily

[0:02:00.6] JM: This podcast is brought to you by wix.com. Build your website quickly with Wix. Wix code unites design features with advance code capabilities, so you can data-driven websites and professional web apps very quickly.

You can store and manage unlimited data. You can create hundreds of dynamic pages, you can add repeating layouts, make custom forms, call external APIs and take full control of your site's functionality using Wix code APIs and your own Java script. You don't need HTML or CSS.

With Wix codes built-in database and IDE, you've got one-click deployments that instantly updates all the content on your site. Everything is SEO-friendly. What about security and hosting and maintenance? Wix has you covered, so you can spend more time focusing on yourself and your clients.

If you're not a developer, it's not a problem. There is plenty that you can do without writing a line of code. Although of course, if you are a developer then you can do much more. You can explore all the resources on the Wix code site to learn more about web development wherever you are in your developer career. You can discover video tutorials, articles, code snippets, API references and a lively forum where you can get advanced tips from Wix code experts.

Check it out for yourself at wix.com/sed. That's wix.com/sed. You can get 10% off your premium plan while developing a website quickly for the web. To get that 10% off the premium plan and support Software Engineering Daily, go to wix.com/sed and see what you could do with Wix code today.

[INTERVIEW]

[0:03:58.5] JM: Bob Fraser, welcome to Software Engineering Daily.

[0:04:00.5] BF: Thank you, thank you.

[0:04:01.6] JM: Many of the shows we've done are about moving to the cloud, but not every business is able to move to the cloud. Why doesn't every engineering organization out there just move all of their infrastructure onto the cloud?

[0:04:18.0] BF: Well, I think you're seeing a clear trend towards hybrid IT, where some of the workloads are in the cloud and some of the workloads are on-prem. With Hewlett Packard Enterprise, one of our missions is to make hybrid IT simple. My part of that at this company is to work with management software that really provides an excellent experience with running a private data center.

From my vantage point, I get to see a lot of why people are moving workloads where. Some of the reasons why they stay on-prem there's a few. First is legacy or a need to integrate with legacy systems. There is backend stuff that's already inside the firewall, so just makes sense for the service to be inside the firewall.

Another common one is security and that could just be the operations of having the control. A lot of times, it's also around the data security. Companies may have some assets that they just don't want to have on the public cloud. Then of course, there's always regulatory aspects to that and maybe compliance issues, data locality issues. Then lastly, one that not every that considers is cost economics. It turns out, we're seeing some workloads move back on premise because the operating expense of it is actually lower.

It's easy to get up and then running in the cloud, but then sometimes you get some sticker shock. Those are some of the reasons why we're seeing a healthy business both in the public and on premise.

[0:05:33.3] JM: There are businesses that have data centers that are already built. There is actually a lot of those businesses. For these companies, the cloud might look really appealing. If you already have a data center, you probably want to keep your data center, because you've already built it, it's like you've already built a house, you don't want to go rent an apartment if you own a house.

You probably want to keep it and take advantage of it. Does it ever make sense to just throw away your data center, or are many people – are they looking at the hybrid cloud idea a lot of times because they have this not exactly sunk cost, but an investment that they've already made in their data center?

[0:06:16.8] BF: That would definitely be a part of it. Some of it is we're still seeing continued investment into on-premise data centers. For some of the things we were talking about before, a lot of people do want to operate part of their business on-premise. We are seeing some really interesting hybrid use cases where people are running their business internally in a private data center and occasionally bursting to the cloud.

You'll see a lot of real hybrid operation going like that. I don't really see the need for on-premise equipment really going away. I think there's enough operational and reasons to keep it going. I think what's happening though, the trend is the want to not just have a bunch of servers with operating systems on it and some software. They really want to have a consumption that's more like the cloud. They want the interface to the app developer to look similar, whether you're on a public cloud, or if you are a private data center.

[0:07:08.8] JM: Right. Much of the advantage of a cloud is not necessarily the fact that you are operating servers that are in a remote location and you have unlimited compute and you've got this wide buffet of options. A lot of the advantages, the procurement process, so if I'm a developer at a bank, for example and I want to spin up an application to do something interesting, I want to write a data science application, or I just want to stand up some internal application, if I'm working on a cloud provider then the procurement process is just I click a button and I get a server and I can do whatever I want with it. If I'm working with on-prem infrastructure, then I'm working with whatever management layer the on-prem provisioning system is providing to me.

[0:08:07.6] JM: True. Couple answers to that that we do here at Hewlett Packard Enterprise, one, we're really pioneering what we call a composable infrastructure. What this is is our hardware can take compute, storage and networking and compose it and decompose it, aggregate it, disaggregate it and present it suitable for running certain workloads. Then every piece of that is all software-defined.

Then we lay our – our management software HPE one deal on top of that. Then we have APIs that we could complete automation. When you get this setup, you could have these pools of resources. If a new need comes along, it's very, very easy for the IT department to quickly jet up

a configuration for somebody to use and then make that available to them in a matter of hours, as opposed to in the old days when people had to run cables, you were talking two months before you could actually get your access to these systems.

Then it's easy to reclaim those resources and repurpose it if you change. That I think is making actual physical infrastructure a lot more fluent and a lot – giving again, a lot more agility and making it similar to consume with public cloud. You can even go as far as to layer with because you got the automation, you can even layer orchestration tools without service catalogs and stuff like that. You can really go quite far with that model.

[0:09:31.4] JM: What is the typical software stack that's running in an on-prem data center and how is it being operated?

[0:09:41.0] BF: I work a lot with the physical infrastructure. You've got the physical layer and sometimes people are running workloads directly on that, like large databases. There's obviously a large, large base of virtualization going on, right? Those applications haven't gone anywhere.

Then where we try to add value there is in the lifecycle operations of those systems. It's one thing to just use some of the tools like VMware vCenter, but then at the day those – the affect host are running on physical hardware, so we've actually provided a lot of integration between the two to make that whole experience really, really simple.

We're seeing in the trend is to move towards cloud native and containers. In that area, what we do is we work with all the major [inaudible 0:10:28.0], usual suspects on the container platforms and then we can either run them in a virtualized environment, or we can actually do containers on bare metal. We make the orchestration of that in based in our own experience and also being able to expand and get back to you towards us.

I think you're seeing all of it. You're seeing some bare metal workloads, you're seeing a lot of virtualization and you're definitely seeing a strong interest in modernization moving in direction of containers.

[0:10:55.1] JM: The “cloud native” experience is desirable, even when you are not on a cloud when you’re on-prem infrastructure, or you’re on some hybrid situation. These people who do have their own data center, whether they’re going to operate their infrastructure entirely from that center, or they’re doing a hybrid cloud thing, they have some challenges that people are – people listen to this that are only family with a cloud, who have all their experiences just working with a cloud provider. There are some difficulties that come from managing a data center. What are those challenges from managing on-prem infrastructure and maybe there’s some fairly legacy infrastructure software running on it. What are the challenges that come from that?

[0:11:46.7] BF: One of the big areas that I think is a challenge that we call its own is the whole area of lifecycle operations. There’s quite a bit of drama involved if you want to do things like operating system upgrades, if you want to do things like firmware upgrades. These have often really consumed an IT department. That becomes a big part of what they have to do and it’s a lot of manual work and it could take weeks to months.

What we’ve done is because we’ve got the software to find approach that we have and we’ve got the OneView software, then what we do is we put automation tools on that scripting or tools like Ansible or Chef. Then what we can do is we can do complete data center automation. That is the thing that really changes the way things are operating.

When you can do this complete data center automation, you can run the scripts and then you can actually do these updates. Then increasingly what we’re doing is we’re making the hardware a cluster-aware of what the next layer on the stack is. We have visibility into a V-center cluster. We have visibility into a Docker container ecosystem or something like that.

What we can do is we can actually do non-disruptive updates and upgrades and well through, you take a node, you take into maintenance mode, you patch it, fix it, move it back it, put it back in the pool. What this is doing is this is really freeing up the IT department. One of the biggest benefits for a company is that they can take this limited resource, their IT department and they can really up their game, they spend less time on these repetitive tasks, or there is more [inaudible 0:13:23.8] automated. Then what they can do is they can work more closely with the line of business people and add more value through the reference path to really going to provide value to the business. We find that that is really resonates with the people that we work with.

[0:13:37.7] JM: Right. If I am working on a cloud, you're putting out something else – if I work on a cloud, the cloud provider is going to take care of things like operating system updates, firmware configurations and also the cloud provider is going to help me with obviously deploying my software, doing something where I roll out a new version of the software and I could do an easy switch over from the previous instance of my app to a new version of my app.
If I'm managing infrastructure on-prem, I have to handle all the updates and the configuration myself. I've got to figure out how to make some workflow, where I can do a rollover one instance of a service to a newer instance of a service. I've got to have some build system. How much overhead is involved in doing all that stuff, like the operating system updates, the firmware configurations, the building software for rollout and stuff? If I do that myself on with my on-prem software, that sounds like I need an entire data center team, or an entire operations team. How much overhead is involved in that?

[0:14:56.8] BF: We could see that take up – it could be a third to 50% of the capacity of the team to operate, because they just get bogged down with these things. If you're doing manual processes and you don't have this automation, it really, really becomes a big brain-on on their effectiveness to move on to other tasks and other duties. Then it does become a good deal. That's why I keep harping this whole idea of complete automation.

The other advantage of this automation is we're increasingly able to doing with the same kinds of goals that the app developers are using. We support a lot of the devops tools with OneView with our ecosystem. You've got choices like Chef, Puppet, Ansible Salt, Terraform. These are the same tools that they're already using for the app layer, and so now what we can do is we can actually apply those directly to a physical instructor.

There's a few benefit to that being able to treat your infrastructure as code like that, because everything now, you've got a sequence stack. The app guys probably already have some of the scripts that they were using to build and test and roll out their app, so now in conjunction with that if you need a dev test system that the infrastructure guys can roll out one that will work nice and seamlessly with that workload and it's all pretty much like the same tool suite. That's I think one of the things that really get you out of this bogged down 30% to 50% of your life is just trying to maintain the ELS, and the firmware, the drivers having to do it all manually, and then

being able to really have a better working experience across the spec from the physical structure through the next layer to the app.

[0:16:39.4] JM: We try to talk about the different businesses that are using this whatever software we happen to be talking about. We try to understand the infrastructure, as well as the business side of things. Because oftentimes, the business concerns are driving the IT decisions or the infrastructure decisions.
What are the companies, the types of companies that we're talking about here that are managing their own infrastructure? Is it like banks, insurance companies, telcos, agriculture companies? What is the average?

[0:17:17.0] BF: I would say the answer to all of that is yes, yes, yes and yes. Any sufficiently large company is definitely going to have large on-prem, or private things that they're doing and they're going to have this need and they're going to increasingly want modernization. They're going to increasingly want agility. They're going to be increasingly want to have do better to better operate and manage their resources. We're seeing this across all the industries.

We align with certain verticals and it's all the usual suspects. It's finance, it could be government, pharmaceutical science, high tech. It really expands – all industries that goes anywhere from SND to small enterprise to very, very large Fortune 500 enterprises.

[0:18:01.5] JM: You mentioned a little bit earlier in the software that you're building OneView, you are concerned with giving these on-prem infrastructure instances, or clusters and with enterprise clusters, you want to give them virtualization and containerization and orchestration. What kinds of technologies do they have in place? I started as a software engineer, I guess after the heyday of VMware. I started in probably 2008, 2009, so I'm not super aware of what the average virtualization stack of one of these on-prem situations is. Can you tell me more about where the average enterprise is at in terms of virtualization and containerization and what are they trying to change today?

[0:18:51.8] BF: Well, I think there's still an incredible presence for virtualization across a few of the different vendors. Some of them source and the usual ones that come up. Then there's also I think an increasing interest in moving to containers. I think one thing is the virtualization is still

really, really popular, really, really common. Then I think the interest is that we provide a better experience there in terms of being able to manage virtualization layer and increase the ability to maintain the lifecycle of it on physical hardware.

Then the businesses themselves are considering doing modernization. Sometimes it's lift and shift and trying to get some things over into containers. Other draw the containers, I think is it's a little bit received of as a little bit more agile and a little bit more lightweight. It makes the apps also a little bit more fluid and easy to move around. We're seeing that as well.
I think some of the things that are preventing that is a lot of it I think ends up being skillset. That's I think one of the areas again where being able to do all those automation and being able to really treat things up and down the stack as code is really key here.

[SPONSOR MESSAGE]

[0:20:20.6] JM: Software engineers can build their applications faster when they use higher level APIs and infrastructure tools. APIs for image recognition and natural language processing let you build AI-powered applications. Serverless functions let you quickly spin up cost-efficient, short-lived infrastructure.

Container platforms let you scale your long-lived infrastructure for low cost. How do you get started with all these amazing tools? What are the design patterns for building these new types of application back-ends?

IBM Developer is a hub for open source code, design patterns, articles and tutorials about how to build modern applications. IBM Developer is a community of developers learning how to build entire applications with AI, containers, blockchains, serverless functions and anything else you might want to learn about.

Go to softwareengineeringdaily.com/ibm and join the IBM Developer community. Get ideas for how to solve a problem at your job, get help with side projects, or think about how new technology can be used to build a business. At softwareengineeringdaily.com/ibm, you will find the resources and community who can help you level up and build whatever you imagine.

Thanks to IBM Developer for being a sponsor of Software Engineering Daily and for putting together a useful set of resources about building new technology. I'm always a fan of people who build new technology, who build new applications and this is a great resource for finding some design patterns and some new ways to build and leverage these technologies, like serverless and containers and artificial intelligence APIs.

Thanks again to IBM Developer.

[INTERVIEW CONTINUED]

[0:22:19.6] JM: The modern practices around moving faster as an organization, it's often things like devops, you hear the word devops, CICD, continuous delivery stuff, infrastructure as code; you want to be able to program your infrastructure, which is much easier with – well, I think it's easier with containers than virtualization and it's easier with virtualization than bare metal servers.

I think the goals with a lot of these enterprises, whether you are a pharmaceutical company, or a bank, or a telco, you want to get these things like devops and CICD. People that are listening to this that work at a startup and all they've ever seen is the cloud and I can infrastructure super easily, that's easy to take for granted. I can say from experience. If you go to a legacy enterprise you're going to realize, "Okay, now I can understand why this company sometimes have trouble moving faster," it's because they're just shackled to the past in some ways.

These things like devops and CICD and infrastructure as code, why is that sometimes hard to achieve with on-prem infrastructure? If you have on-prem infrastructure, how do you get to a place where you can use that infrastructure in a way that lets you move faster?

[0:23:45.2] BF: Okay, the challenges are there is just a lot of moving parts often with proprietary APIs, if you go too far down in the stack. There is bio settings, settings, firmware, drivers, connections to the fabric, whether it's a fiber channel or Ethernet. Then there could be storage provisioning. You take all those things and you can end up with something like a thousand lines of code.

One thing that we've been able to do with the HPE OneView is it uses a template and model. What you do is you get all of your best of breed people across your networking storage and workload people, like with the virtualization or containers or whatever the next layer up. You get them altogether and you define a template. Then that template can then be provisioned against like a server and some compute resources.

Now that thing can also then be repeated, repeated and automated. Now what you do with that though is then you could take that template and you could do it through a GUI, but our GUI actually uses our REST API. Then the team that I work with, I work with a team that takes that REST API and then we put language bindings around it, like Java, Python, Ruby, Golang. Then we also integrate with the common devops tools. Again, it's like Ansible, Chef, Puppet, Saltstack, Terraform, those guys.

Now, what you're actually able to do is you're able to take those same devops, CICD practices, things like declarative programming models, desired state and you're able to actually use those same tools, those same practices that you would use to build the app and it actually represents a physical topology of the machine.

Some of the advantages that you get there with this is you can actually check this in to git. You can actually do version control. That gives you transparency, that gives you auditability, it gives you actually the ability to rollback. You can actually take, run the scripts, it works, everybody is happy, fine, you move on, right? When oh, something isn't quite right, you just roll back to the previous version that was in source control, you apply the script and it puts things back the way it was. Those are the things that you can do.

Then the other end of that is you can either run the script as an active thing that will manage things, but you could also run it as just a reporting tool. Now you get into the whole area of being able to show compliance. What if somebody went and messed with the server while the server template is connected – the server profile is connected to the server template over port if it's out of compliance and you could then force it back, or you can integrate with the original management.

© 2018 Software Engineering Daily

This whole idea of being able to actually check your recycle infrastructure into source control has been really, really powerful concept. It takes a little while – reason why it's novel and you have to change your thinking, change your mindset. Once you get there, it's really, really powerful.

[0:26:33.6] JM: Your goal with OneView, where you're trying to build – if I understand it correctly is it's like a shim over pre-existing infrastructure, such that that infrastructure can be made more programmable.

[0:26:49.4] BF: Exactly.

[0:26:51.0] JM: Okay. I mean, that sounds hard. What does that look like in practice if you're trying to engineer – When I think of on-prem infrastructure – I could be wrong, but when I think on-prem infrastructure, I think a lot of heterogeneity. I think if you're talking about all the different infrastructure configurations and networking setups and so on that you could potentially across a large percentage of the Fortune 500 where you have banks and telcos, etc., etc., these are like different archeological digs of different infrastructure setups. How do you build, how do you engineer a consistent shim over all of those different infrastructure configurations?

[0:27:32.9] BF: Well, I think one of the aspects of it is on the OneView side, it interfaces directly with the hardware. Where we're building it, it was built with the idea that that would be a common API as a central design doc. Each of the teams that are working on something like a network adapter, or a new server, or new settings in a bio boot mode or something, they're all working from a common resources model.

What they do is they do whatever is required to leap in the middle with the resource model. Then because we've got these different resource models, okay it could be two slightly different networks, but they'll still appear at the course level to be by network, and then it complies with that resource model.

Maybe different attributes that you can on one over the other, because it may be more modern, or they have another comp that you can turn. That would just more show up as a different

attribute of the resource model. We can tag those and you know what version it is and you can feel what thing it is when you tell them click on it.

By working to a consistent resource bottle across the different types of resources like servers, networks, storage, we can come to – meet in the middle. Then on top of that, we can put the API, right? One team it's going from the hardware up and they're going to the common resource model. Then you go from there. That's I think how we managed to take a wide variety of different server generations, different networking fabric types, different storage and we can at least put it into a common model, or 80% of the use cases where we really want to be able to get some kind of software-defined management around it.

We don't need to map a 100% of every little thing that you can do. If we have a three-part storage, it has an API on the management side that might let you figure out how you want to do data encryption. On the OneView side, we really just game with the idea that the person who is consuming this really just consume up the storage pool, and then maybe other people who are worried about the policies that go around the storage.

That's okay that you have other tools that aren't completely mapped in. The idea is to have very, very useful resource model that is at one hand generic enough that you could understand conceptually and all of our engineers can work on that common model, but flexible enough that it could be differentiated that if you really double-click on the resource you can see, "Oh, yeah. Okay, here I've got these options."

[0:30:04.9] JM: When you're looking at the space of different ways that that people can modernize and manage their infrastructure – I haven't covered this space too much, but I've done a couple shows about OpenStack, so that's like a open source way of managing some of your data center infrastructure. How does HPE OneView compare to OpenStack, or can you use OpenStack? Can you leverage some of the tools from that in architecting OneView?

[0:30:37.6] BF: Oh, absolutely. We have ironic drivers that actually work with OpenStack. We work pretty closely with the OpenStack community. In fact, Hewlett Packard Enterprise works quite a bit in open source. We've got our own github organization, there's I think over a hundred different projects in there. There are probably about 40 OneView projects out on our public

github that are part of that. As an engineer, the tools, each language bindings, each body language bindings, all of the stuff is out there and we work with it a lot in open source. We can work both with OpenStack, but we can also work with VMware, we can also work with Hyper-V, we can work with KDM.

We really pride ourselves on having a very, very broad ecosystem where we've got integrations with many, many, many of the common tools and vendors whether they're proprietary or open source. I think, that our goal is to give the consumer, the IT department where I was trying to use insist that we want them to be able to have a great experience using the tools that they prefer. We try to integrate with all comers.

[0:31:40.4] JM: If I want to get a cloud, a cloud-like experience on my on-prem infrastructure and I want to use OneView, how do I deploy it? What's the deployment process for getting this cloud experience on a data center?
[0:31:57.7] BF: I like to divide it in a day zero, day end operation. It's going to depend on what you mean by cloud. Pretty much, you have your different choices. You're doing virtualization, containerization, you could be doing bare metal services and you decide which sets of tools you're going to use. Then because the ones these stuff can do all the infrastructure automation and we have this template architecture, your day zero is pretty much setting up the resource pools and setting up the different resources and then maybe setting up the clusters.

We have a whole – if you chose VMware, we have a whole dedicated product called HPE OneView for VMware vCenter. We've actually got plugins that go directly into vCenter and they let you do things like grow a cluster, or add new bare-metal service into the cluster and you could do rolling firmware upgrades, you can see the visibility from the workload, from the [inaudible 0:32:56.8] virtual machine to the ESX node, through things like networking and storage, all the way back to the physical networking and stuff like that.

We make those experiences really, really easy so we handle the day zero getting the last mile. We worry about getting the last mile of what is on the server, what is the physical infrastructure. Then we try and integrate with the next layer up. If you were doing OpenStack, then you would use our ironic driver, our ironic driver would provision the bare metal. Ten we work with all the

other ones like Neutron and everything else. We would just seamlessly fit into an OpenStack environment there.

You can go with popular vendors like Red Hat; we work with them. There, our tool of choice would probably be Ansible, because they're very big fans of Ansible. We've already got reference configurations and Ansible playbooks and publish those on github. You can make an experience where you could just pick the toolchain you want, the stack that you want and we probably already have a body of work that will get you up and running, and then also do common lifecycle operations once you're up and running.

[0:34:04.6] JM: Whether I'm working with Red Hat, or Red Hat Linux, or some other operating system, there's a variety of different ways that the deployment can happen. Once I have – if I deploy OneView and I now have this cloud-like layer over all of my different types of infrastructure, over all of my operating system instances, what's the experience like at that point? What am I actually doing with this? Do I have a dashboard available to me? What kinds of things can I do once I have this cloud-like layer that's sitting across all my infrastructure?
[0:34:46.7] BF: There's a few different views you can get. OneView itself as a dashboard and it allows you to see all the different resources and it does health monitoring and it does alerting. It can manage many hundreds of systems. Then if you have multiple data centers or you have multiple instances, we have a thing called global dashboard, so you can across a large organization, you can actually roll up all of the reporting that comes in from OneView and you can see the health and resource utilization and alerting across all your different data centers and all your different localities and everything.

There's the GUI, there's that. There's that aspect of it there. Then again, we tend to have a lot of plugins, or things that can plug into other tools if necessary. Then we have our third-party integrations that there are other people who provide different kinds of tools that we even go all the way down to the power and draw – electrical draw and temperature and we report that out to data center facilities management software like [inaudible 0:35:47.9].

You can even get a dashboard that will just show how warm your data center is and which machines are heating up and how much electricity you're using. There's a lot of different ways you could go with the OneView.

[0:36:02.1] JM: Let's take some engineering example. That example of understanding the power consumption of the data center, so I should be able to do that from the dashboard. What kinds of engineering does that require on your site? If you want to build a feature to understand the energy consumption within a data center, what does the construction of that feature look like?

[0:36:25.5] BF: That's one where we tend to partner with our ecosystem. What we provide as part of OneView, you can see it in our dashboard is we provide the raw data. We're doing all the sampling. We're looking at power draw, we're looking at temperature and we're also providing things like the CPU utilization, because they tend to go together. If you're really driving a CPU hard, it warms up. Those statistics are all available.

We have message bus, for those are continuously being reported where you can deploying model. Then we work with some of the other – the people that this is their business. They understand data center facilities management, so they can map the location of where the different resources are, which systems are actually getting warm. They can also make recommendations on moving workloads. In some cases, what they can do is they can now show you feedback in through OneView and turn the CPU speed down a little bit, and then that will reduce the power drop.

If they're at risk of getting overheated, the actual management software – so they do their part. They really understand facilities management and what it means from the physics point of view. We have all the raw data that they need to make themself successful, plus things like a feedback loop.

Then you can take that same example and you can take it in other different directions. We have people who do automated remediation. We have folks that do intelligent workload placement. Got a couple of partners Dentify and Tergonomic, what they do is they actually do predictive analytics. They look at the load of on our CPUs that are doing virtualization, or containerization and then track it over time. They suggest you should probably take this workload and migrate it from the set of servers over here to the set of servers over here, or on Sunday I know you're going to run this big batch job, so let's move these things away from there before Sunday.

They can actually increase the overall utilization and effectiveness of the resources. In some cases, they've actually reached back into OneView and moved parts around, so that you need their answer capacity or inform to another layer like vsend or something to might to do the end migration.

When you start getting all of these things talking to each other, really, really cool things can happen. Sometimes they end up with a two-way communication, which is really interesting. Things like automated remediation if a server goes down, but you're supposed to have two servers to get me talking to your load balancer or four. Then it can automatically spin up a new one. Then because everything's automated, the workload just spins up back down there and joins back in with the load balancer.

[0:38:57.8] JM: Speaking of load balancing, if I am on a cloud provider, I can have basically unlimited access to infrastructure. I could spin up as many servers as I want. I could just scale to the moon. If I'm on on-prem infrastructure, there is – I mean, well obviously even in the cloud there's limitations. There are some fixed number of servers that are sitting in the cloud that I actually have access to, but it feels infinite because it's AWS, or its Google Cloud or whatever, and it's a lot of resources.

On-prem, there is some lower bound to how much resource I can consume. What's the process of making sure the developer, so once the developers have access to the internal cloud, how do you make sure that no developer consumes too much, or you don't have a developer, an internal developer who DDoS's your entire internal cloud? What's the process of regulating the instance consumption?

[0:39:59.5] BF: OneView does have role-based access security and things can be tagged and things can be zoned and things can be provisioned. Then also with this template and architecture, you really can just assign access to the physical resources and then carve that up the way you'd like. Maybe at the department level, maybe it's you assign it a different way. There's that course level of doing it.

Then sometimes the person who's actually developing the app is really just on a virtualized, or containerized layer. Anyway, you can still turn those knobs as well. You have a few different ways of doing it. At least, as far as the delegated model and the model within OneView for the physical resources, that's got complete resource pool isolation if you needed and things like delegate of all based access and those kinds of things. You can manage it either done at the server layer, or you can use the other common tools that you would use to reduce how much people are consuming at the next layer on the stack of which.

[0:41:00.8] JM: In the building of OneView, do you have to solve distributed systems problems, like having a master node coordinating the other nodes in the system? Do you have to solve these kinds of problems, or are these solved at a lower layer of the stack?

[0:41:17.0] BF: Well, the architecture of OneView is that it is an agentless system. OneView itself is either a virtual client.

[0:41:25.4] JM: Agentless?

[0:41:26.8] BF: Yup. It's agentless. OneView is agentless itself. It's deployed either as a virtual machine and then it imports all the information from the physical infrastructure that it's managing. Then it just communicates directly with the physical infrastructure. We have all the HPE resources have a thing called integrated lights-out, which is a hardware management layer, which actually can do things like power on the machine, power off the machine, change the firmware, change the BIOS settings.

Then OneView communicates directly with that, so you don't ever have to install an agent, you don't ever have to get a guest OS and put something in the guest OS. Now all of that is you don't ever have to do that. Then basically what you do is you just deploy the OneView VM and then that manages all those physical resources. Then in the case of our synergy system, which is one of our more recent hardware offerings, there's actually a dedicated piece of hardware that's running inside of it called the composer. Then you just talk directly to that. In some cases, we actually have a dedicated embedded appliance and in some cases it's a VM, but we don't ever have to have minions, or little agents that need to be installed anywhere.

[0:42:37.2] JM: If you're building an API that allows people to have a shim over all of their existing infrastructure and be accessed through an API, you have to define what are the different API calls that you need for a cloud basically, for an internal cloud. Can you tell me a little bit about the API service? What did you want to define at that top level of the shim that's fitting over the entire data center, on-prem cloud data center? What's the API surface area look like?

[0:43:16.9] BF: Okay, to begin it's all based on a REST API. It's a representational state transfer. Then everything is modeled as resources. The rough taxonomy of the resources is we can actually model the data center, and then we can even give you almost little schematic drawings of where the different servers would be in the data center if you choose to have that. Then we have, the next one would be things like enclosure groups. Those actually map to the physical enclosure that you're going to be sliding hardware into.

Then inside of that, you have enclosures and you have – in the enclosures, you actually number the base. We tell you about the day and then we tell you about is this a short one, or a tall one and then is it CPU? Is it storage? What is it? You actually are modeling the physical hardware. If you go into the OneView GUI, we'll even draw you a picture and show you – once you click on one of those things, you can actually see what's there. We'll even tell you what connectivity is in there. It doesn't have an Ethernet card in it and stuff like that.

We start modeling all that, but the raw model of the software. They're all modeled as part of this REST API. Then things compose, or contain similar to the way you would build software. I have an enclosure. It has these bays in it. This bay has a server. It's this type of – it's this hardware type. It also has these two network cards, right? We model that. Then when you start poking that software, it reaches in through the actual software-defined goodness and actually manipulates the physical hardware underneath. There's a very pretty cool direct mapping between the resources you see in the API and something you could actually go through the pre-touch.

[SPONSOR MESSAGE]

[0:45:11.1] JM: Fission is an open-source Kubernetes native serverless framework. Fission allows you to easily code serverless functions in any language and have them run wherever you have a Kubernetes cluster, be it in the public cloud, in your own data center, or even on your own laptop.

Fission automatically manages the infrastructure for you, so there's no need for any in-depth knowledge of Kubernetes, no containers to build or registries to be managed. With Fission, you can essentially run your own lambda-like service, benefiting from the speed and the cost savings of serverless on any environment.

Go to fission.io to get started. You can use Fission to create serverless API back-ends, implement web hooks, create event handlers and more, all without having to bother with managing infrastructure, let alone Kubernetes at scale.

To learn more about Fission and to try it out, go to fission.io.

[INTERVIEW CONTINUED]

[0:46:22.7] JM: There's so many different configurations and we talked about a little bit earlier. There's so many different configurations of infrastructure this could potentially run on, but it's a consistent API on top of all those different infrastructure types. What's the testing process for all that heterogeneity? Can you just test that API, that REST full API that you've built, or do you have to do lower-level testing for all, I guess for all of the connectors between that API? Just tell me about the process of testing and making sure you have consistent result from that consistent API.

[0:46:59.6] BF: It is a challenge. We have two approaches to this. One, we do have quite a bit of test rigs and physical testing on physical infrastructure, especially if we're coming out with new hardware that always works very, very closely with hardware. It is a little daunting, because there are a lot of pieces and a lot of moving parts and a lot of things that you have to test.

Fortunately for us, we've also built something we call our data center simulator. We actually can model the behavior of a server, or a network connection. As you're just exercising the code and

exercising the API, you can actually for a broad set of the use cases, you can actually just use this data center simulator. Then we put those into CICD pipelines and we do automated testing and things like that. That's one of the ways that we can survive this, even though there's just an incredible array of different kinds of possible configurations.

With the data center simulator, it's backed with an XML schema. We can actually change, but the simulator perceives of as the physical data center that you're running on. We can make it look like one of our C7000 blade systems, or we can make it look like one of our newer synergy systems. We can actually pretend that it's running on physical hardware. Then that'll get us through quite a number of the tests, but then because it is a simulator, we of course always want to test it at some point in the cycle on real hardware as well.

[0:48:27.9] JM: Okay, so we've talked a significant deal about the layer below OneView, the layer below this consistent API. Once you have this set up, you can do things like install Jenkins, or install additional monitoring software. Let's say I wanted to install a Jenkins cluster and I wanted to have continuous delivery within my organization, what would the process of setting that up once I have this consistent API for my infrastructure, what would that look like?

[0:49:00.2] BF: Typically, Jenkins is just going to run as a workload a layer above us, right? What our own HPE IT department does to service our software engineers with our CICD pipeline is we've chosen to use – in our particular case, we've chosen to use Terraform and Ansible to provision physical infrastructure like its energy system. Then the Ansible playbook we'll also and install. In our case, it's a Docker Enterprise Edition.
Then what we'll do is then those same – another set of scripts deploys beacons on there and then we run Jenkins as a containerized environment. You've got your master Jenkins node. Every time we poke something in our internal corporate github, it runs a web hook. The web hook goes after Jenkins and says, “I need to run these tests.” A container spins up, the Jenkins runs in that container and then it goes away when the tests are done, the container goes away. That's our stack. If we need more capacity, then our IT department which just run some of those Ansible scripts again and it just change a variable in our software-defined infrastructure and our infrastructure as code they'd say well, instead of four, I need six. Instead of a 100, I need a 150.

Then it would just consume some more resources and add them into the pool for that, the physical layer. That's how we built our CICD pipeline. There's all bunch of different ways you can flex it.

[0:50:28.4] JM: If I'm on a cloud provider, one thing that's useful for me is I can look through all these different services. I can get ideas. I can see solutions for queuing and databases and basically, all of my problems can be solved by some API that I can buy from a cloud provider. If I don't have any problems, if I'm just looking for new ideas, maybe I can take some machine learning tool off the shelf of the cloud and I can start messing around with it. I can start building something innovative. What's the process of presenting a category, or a catalog of different things that if I'm the developer, I can look through this catalog and say, “Okay, this is going to solve my problem, or this is going to be an opportunity for my business.”

[0:51:18.7] BF: We have a composable ecosystem program. That is where all of the partners and our open source projects all meet around OneView in the OneView API. Those tend to be in area select automation, areas like cloud automation, or cloud operation, IT operations. We discussed the facilities management there. As I mentioned, there's about 40 open source projects out there as well. You can either directly use one of our SDKs, directly work directly against the API. There's a bunch of open source projects that are already out there and then people can just file a github issue, or they got an idea they want to enhance it, they could fork the code. They can even do a pull request.

Then we've got our composable ecosystem. We've got information there across the different operating system vendors, the different container people, automation tools, virtualization, IT operations. You can go to our composable ecosystem. Then if you want to join, we have a partner program, so they can get them access to resources of the APIs. That's one route. You can probably go buy something “off the rack,” or go get an open source tool and build some cool stuff.

Then HPE also has a very large services organization. Anybody who would really like some assistance on that, we have a complete set of consulting resources starting from the advisory end of it, where we'll come in and work with your line of business to a PLC, give you an initial good success, all the way through fall data set management, where we’ll run your data center

for you and build out all these automation tools to your workflow. There's a lot of ways to get in the game with this technology.

[0:53:08.3] JM: We've been talking about in the context of banks or insurance companies, these large technology companies that have their own data center. There's also collocations, so these places where people are renting servers. Have people used OneView as an API, or a management system for collocations, or these large server farms that are rented to multiple people?

[0:53:34.1] BF: Yeah. We definitely have service providers who they're doing managed services, they're doing managed data centers for people and at whatever scale makes sense. They definitely use OneView in many cases. Yeah. That's absolutely a category. This particularly I think in different geographies outside the US, that's actually a really fairly common model. HPE has a very big channel business as well. Some of the stuff, they end up going through channels, some of the stuff with the managed data centers. We work with a few large ones that are across a bunch of different geographies.

[0:54:10.2] JM: What else is changing in the market for on-prem infrastructure software? What are the changing demands? I talk to mostly people in who are working in cloud providers, or they use a cloud provider. Some things that are changing recently are there's just some higher level managed services people want. Some people like functions as a service. What are the things that are changing in on-prem?

[0:54:35.0] BF: At my end of it where I'm looking at the management stuff, I think what people are looking for is more automation and remediation is one trend. We've got actually pieces of software in OneView that can phone home. Then we can see that there's something going on. I think it can even order parts and have them send.

I think that, you start continuing down that line, we're looking at taking some of our integrations with things like info site that's trying to apply some predictive things and some of the kinds of AI techniques, or collecting some of the statistics for you and making suggestions. This look like you could be running out of resources here, or this could be a problem down the road, or here's

an alert. We found that these other people with configuration similar to yours, you'll benefit from this dispatch or something like that.

I think what you're trying – people are really looking at one aspect of it, they're looking for “more intelligence” in their software. We're tracking all of that. We're trying to build in those kinds of features. We're trying to build in predictive things, we're trying to build in a little bit of an almost an AI aspect to it.

The other thing is I think on-prem software is also itself going to a hybrid model, right? Where there's a big piece of it that's on premise, but people are still interested in things like the monitoring maybe being in the cloud. Maybe some higher level reporting being up in the cloud. Maybe things like logging being up in the cloud. That really I think speaks to just wanting to be able to do centralized management across a big bunch of data centers, or something like that. There's still a very big piece of the software that would stay on-prem, but there would be some like you say, higher level management things that are desired that can be provided “as a service.”

We're looking at the hybrid model of enterprise software in this management space as well. We recently did an integration between OneView and Microsoft on log analytics. We can actually send the log up into the cloud and that's the preferred direction that Microsoft guys want their customers to go to. We're following that. We're working with them on that trend. Those are at least a couple of the things I see changing with at least on-premise management software.

[0:56:53.3] JM: That's a great example of integration. The fact that HPE can work in a partnership role with a cloud provider, I mean, I guess it sounds everyday, it sounds completely normal today, but it's just funny because your built HPE OneView is built in a way that it's a result of a lot of proprietary things that didn't work well together originally. I mean, I come from a later along the timeline of infrastructure software, but my view looking back is that companies used to not play so nice together. They used to not integrate so easily. Today, if you don't play well with the rest of the ecosystem, you don't have a business.

[0:57:44.5] BF: Exactly. I see two forces driving that; one is a lot of the open source has changed the thinking of a lot of these companies, because there's certain aspects where

multiple competing companies are working on common open source projects and benefiting from the results of that. I think that was a benefit. Then I also just think that at the end of the day, there's a customer-driven and market-driven effect too, which is I guess the phrase people use competition.

There are areas where we may compete, where I'm going to have a product, you're going to have a product, we're going to compete on that offering. At the end of the day, any sufficiently large company is going to be wanting to do business with both of us. If we can work together in certain areas and we can provide a better experience for the end-customer, which is I think that's really what you're seeing here. What you're really seeing is all of us want the customers' business and it is in everybody's best interest if we all work together to provide the best possible experience for that customer.

It's always a changing – it's like shuffling a deck of cards; you work with another two, three partners in a different customer, the mix could end up being different, but I think that's just modern business today. Sometimes you compete, but you always have to know how to integrate and what pieces make sense together.

[0:59:05.9] JM: Bob Fraser, thanks for coming on Software Engineering Daily. It's been really fun talking to you.

[0:59:08.9] BF: Yeah. Really, really enjoyed this Jeff. Thank you very, very much.

[END OF INTERVIEW]

[0:59:15.1] JM: GoCD is a continuous delivery tool created by ThoughtWorks. It's open source and free to use and GoCD has all the features you need for continuous delivery. Model your deployment pipelines without installing any plugins. Use the value stream map to visualize your end-to-end workflow. If you use Kubernetes, GoCD is a natural fit to add continuous delivery to your project.

With GoCD running on Kubernetes, you define your build workflow and let GoCD provision and scale your infrastructure on the fly. GoCD agents use Kubernetes to scale as needed. Check

out gocd.org/sedaily and learn about how you can get started. GoCD was built with the learnings of the ThoughtWorks engineering team, who have talked about building the product in previous episodes of Software Engineering Daily, and it's great to see the continued progress on GoCD with the new Kubernetes integrations.

You can check it out for yourself at gocd.org/sedaily. Thank you so much to ThoughtWorks for being a long-time sponsor of Software Engineering Daily. We're proud to have ThoughtWorks and GoCD as sponsors of the show.

[END]