CORPORATE PARTICIPANT

Mehran Hadipour
Robin Systems – VP Alliances

PRESENTATION

Moderator

Welcome, everyone, to the Intel Network Builders webinar program. Thank you for taking the time today to join us for our presentation titled: “The Role of AI-based Hyper Automation for Cost Optimization of Network Services”

Before we get started, I want to point out some of the features of the BrightTALK tool that may improve your experience. There is a Questions tab below your viewer. I encourage our live audience to please ask questions at any time. Our presenter will hold answering them until the end of the presentation.

Below your viewing screen, you will also find an Attachments tab with additional documentation.

Finally, at the end of the presentation, please take the time to provide feedback using the Ratings tab. We value your thoughts and we'll use the information to improve our future webinars.

The Intel Network Builders webinar series takes place live twice a month, so check the channel to see what is upcoming, and access our growing library of recorded content. In addition to the resources you see here from our partners, we also offer a comprehensive NFV and SDN training program through the Intel Network Builders University. You can find a link to this program in the Attachments tab, as well as a link to the Intel Network Builders newsletter.

Intel Network Builders business partners have been working to accelerate network innovation by optimizing their solutions on Intel technologies.

Today we are pleased to welcome Mehran Hadipour, Vice President of Alliances for Robin Systems at Rakuten Symphony Company. Mehran leads Alliances at Robin.io and is responsible for Robin’s Industry Partnerships and Alliance strategy. A respected industry veteran who is passionate about driving business success through alliances and win-win partnerships, Mehran brings to the table years of experience in executing and driving change at major infrastructure and enterprise software players, as well as several Silicon Valley-based startups and high-technology firms. He holds a Master of Science in Computer Engineering degree from Syracuse University, and a Master of Business Administration.

Welcome, Mehran, thank you again for joining us today. I will hand it over to you to start off.

Mehran Hadipour

Thank you so much for the nice introduction, and I appreciate you guys taking all the time this morning to hear a little bit about this interesting topic that I think you will find very informative. And to level-set the conversation, there's been a number of very innovative technologies that are based on AI and automation that is taking hold in various industries, and it's becoming more and more apparent that AI-based automation tools could be adding a lot of value to telecom operators that are providing network services and optimization of their network. Today's talk will cover some of those opportunities, present a certain approach to delivering that kind of automation, highlight some of the benefits that could be achieved using these types of tools, and I hope that you find it an eye-opener in terms of what can be done in technology, and the business perspective, and some of these optimizations we're talking about could be really delivering significant benefits to operators going forward.

Intel, as a partner of Robin, has been very instrumental in terms of building enhancements and technology features based on-- that highlights the value of their hardware platform in some of these use cases, and these specific use cases also take advantage of Intel technologies to achieve these capabilities.

To get started, I want to introduce Robin a little bit as a company to highlight what we do on some of our products to level-set the playing field a bit. So Robin is a technology company in Silicon Valley. We've been in the business of cloud-native infrastructure and automation services for some time. We started our journey in the enterprise sector and then later on migrated to telco. Today we are running the entire 5G network of Rakuten Mobile on Robin end-to-end and orchestrating the network services associated with the 5G workload. Recently, just a couple of months ago, Robin was acquired by Rakuten Symphony. Rakuten Symphony is a software provider as part of that family that provides a number of offerings in the telecom sector, including Robin, which is our Symcloud offering from Rakuten Symphony. Our goal for telecom operations has been to deliver a platform that has automation and orchestration across the entire telco stack, including edge applications and telecom services.

Robin has three distinct platforms. The first is our cloud-native platform, which is basically a Kubernetes distribution that has been enhanced to run telco services. It includes a number of enhancements in addition to upstream Kubernetes. It uses certain Intel technologies and Intel OpenNESS technologies, including markers and SR-IOV, and is designed to support large-scale deployment of telco network functions. Robin MDCAP, which is our Metal to Service Orchestrator, is the primary platform that is being used to do this kind of automation for-- AI-based automation that we'll talk about later. It's a platform that runs on the primary data center or in the cloud and can manage a large number of clusters deployed at the Edge and core, and do lifecycle operations, or scaling and healing automation, a number of capabilities to make network functions deliver faster and operate better.

So, MDCAP is an independent platform and runs on Kubernetes as an application, and it can support deployment of workloads of any Kubernetes distribution, on-prem or on any cloud.

So, MDCAP, as I mentioned, is a primary vehicle to orchestrate and automate certain capabilities, including AI-based automation that we're talking about here. If you think about the problem of managing network service delivery that spans across a number of data centers with core functions and OSS/BSS layers running on them, as well as all the way to the far edge, which could be running O-RAN DU/CU workload on a small footprint distributed across thousands of nodes that has to be managed, the number of capabilities that are essential to be able to operate and manage and deploy network functions in an independent manner, discovery is the primary function of understanding what is deployed and adding to the inventory, and prime it for observability so we can monitor the health of the devices, and hardware, as well as network functions, the lifecycle management of those network functions, and doing actions around healing and scale. All those things has to be done across the entire infrastructure, at scale, with a large number of sites deployed and managed in a single pane of glass. This type of hyper-automation is essential to be able to bring the operational aspect of the 5G workload in a much more affordable manner, and also be able to manage the entire network function delivery centrally and easily.

MDCAP architecture is highlighted in this picture. There are several components that are important to mention. The infrastructure orchestration refers to the deployment of bare metal hardware or virtual machines, and all the back-end physical network devices, and managing them in terms of their lifecycle. The cluster orchestration is a functionality of deploying Kubernetes and managing the Kubernetes clusters, and provide upgrades, and automation on any type of scale and workload that needs to be managed on top of the cluster. The service orchestration refers to the fact that the network services that are running on these Kubernetes distributions on a distributed manner, including a RAN running in the far edge, or core running at the central data center. It has to be managed and orchestrated end-to-end.

To achieve all these three levels of services, MDCAP has an element-level manager that does the discovery, health monitoring, policy-based automation. It gives you a signal view to the entire infrastructure as it's deployed, a distributed engine that manages the inventory, an active policy management tool that understands the policies that are defined for network services and enforces them, including topology, config management, and so on.

All the observability is an important factor of doing AI workload management, so we have end-to-end observability built into MDCAP that captures the information coming from each individual cluster and aggregates them, and correlates them to an actionable workload that could be then deployed to make up the updates, the automation, the scaling, and healing, orchestration and so on.

The lifecycle operations of the entire deployed network functions and bare metal components are also managed by the system, including things like restart and termination, upgrade, deployment, healing and scaling of individual components.

So, MDCAP as an architecture is basically an executable engine that could be used for some of these AI-based automation policies. However, to really make the system react to data collected from the clusters, and network functions, and take action based on certain AI understanding, there needs to be additional components to be able to create a data model and create an inference from the data model and then use it to take corrective action.

I'll cover some of those capabilities in subsequent slides, and talk a little bit about how we actually achieve that with MDCAP.

A little bit about MDCAP itself. So, MDCAP, as I mentioned, is a single dashboard for entire infrastructure. It looks at all the deployed environments on-demand. It has a concept of workflows that could take action at scale across all the installed components. For example, you can change the frequency or upgrade the BIOS, or restart a pod at scale across the entire infrastructure. And you can create these workloads that apply to specific components in the inventory, allowing a very manageable model to do upgrades, healing, scaling, and automation around network services.

This is an example of a workflow. A workflow in MDCAP could be deployed in what we call a Copilot. The Copilot allows you to configure graphically all the steps to be taken for a specific action, and ties those actions together. You could have parallel tasks or several tasks, and each workflow could be assigned to now execute across certain types of elements.

So, this Copilot allows you to execute the workloads. And then in the Playground, you can actually test the workflows and then you can execute them at scale using the batch engine to deliver the appropriate change across the entire network function deployment for hundreds and thousands of sites.

The Factory itself allows you to do the creation and testing of these workflows. It gives you a graphical user interface to deploy a specific dash, indicating a certain action to be taken. This, for example, happens to be an HA installation of a Kubernetes cluster. So, all those tasks could be automated. For example, if the action to be taken is to start a new cluster for the purpose of taking over some of the load, and do horizontal scaling of a network function, you can then execute this workload that achieves exactly that.

The Factory itself has an entire log management system that allows you to restate the workflow at any point and allows you a seamless way of managing a complicated set of tasks, and understands what the issue was, and then restarting the workflow and debugging what is being delivered.

Now, that we actually have an understanding of the underlying software infrastructure to manage some of these AI-driven actions across the network functions and network services, let's talk a little bit about what kinds of actions would make sense to do based on AI. And there are three of them that I'll talk in a little more detail about today.

The first one is quite interesting. It's called Active Power Management. The concept here is that the requirements of network services vary quite a bit based on time of day, specific events, even day of the week, and there are a lot of opportunities to optimize the power consumed by the network service by managing the power dynamically and allowing the consumption of resources to be optimized for the traffic that is being passed through the network.

This is essential, especially looking at the 5G deployment, and scale of the 5G deployment, because you're talking about thousands of nodes that are running network services, including gNodeBs that are distributed and driving antennas, and when the traffic on those specific nodes are low for whatever reason, let's say it's the middle of the night or different timeframe, running them at full power, it consumes a lot of wasted resources and power seems to be a very significant expense in the operation of a 5G network, and building a model that dynamically updates the powers of the system on-demand, without impact in service, and optimizes the usage of the

resources in such a way that the network traffic is maintained, while power is reduced, is a significant saving opportunity for the operators and the level of automation that is required to do that is being discussed later in the subsequent slides.

We actually did another demo of this in MWC in Barcelona a few weeks back that showcased how those updates could be made, and we have an active model for managing the power, that there was a lot of interest in the show around that.

The same concept could be extended to things like healing and scaling. On healing, you can actually predetermine-- based on some of the AI input received from the hardware, you can predict certain physical failures, for example, from a disk or a network card and take corrective action, and healing action in advance of the failure, such as maximizing the availability of the network.

Similar actions could be taken on network function scaling. Additional workers could be assigned to run more core network functions, and so on, to try to scale out horizontally or vertically the network services at the appropriate level to achieve the additional performance required, and then dial it back when that demand decreases or goes away. This dynamic fashion on doing scaling and power management, and healing not only brings an unmatched level of improved availability to the network but also reduces the operations cost and the infrastructure requirements. You don't have to design the network services and hardware infrastructure for the peak level of consumption. You could design it based on average and then accommodate the overall peaks by adding additional resources on-demand. This approach can also extend across regions. For example, if some regions are experiencing more traffic, you can add more resources to the larger region, or extend the capabilities around things like network slicing. This allows you to dynamically change resources being used by one size versus another.

Let’s start by talking about dynamic power management trends. I think this could be a pictorial representation of the potential, in a 24-hour period, a network’s traffic can change in certain timeframes. You will have a much lower demand on the network, while in some other times there could be a significant peak. The variability in network traffic requirements could vary greatly, and there was a significant difference between one of the peak over time versus the non-peak over time on our network traffic being consumed, and that represents an opportunity to adjust and fine-tune how we can deliver these network services by adjusting the appropriate network infrastructure components.

So how do we go about doing that? So if you look at a cloud-native platform that is deploying network functions such as UPF, and you're going to deploy this model across multiple worker pods in a Kubernetes cluster that is running Telco core, you would see a traffic flow over 24 hours as depicted on the orange graph on the right. And now if you look at the underlying infrastructure that is available to you, in terms of the DAaaS that Intel provides in terms of power management, SR-IOV configuration, for example, IPMI, and all the things that you could potentially use to adjust infrastructure to accommodate the change in demand.

And the Telemetry Platform that is driven by the hardware, including CPU usage, power consumption, network traffic consumption, and so on, all of those could be now used to collect matrices from the underlying infrastructure, from the platform itself, from the cluster, and from the applications. And we have a Robin Cloud Native Platform, Prometheus, endpoint configured, that receives all the appropriate cluster information, application information, and platform information, into Prometheus and creates an approach that you can now start training a model to understand exactly what are the components over time, or how much resources that computes, to create an AI model that could then be used to create an inference from the model, and take the inference from the model and then drive actions back into the infrastructure.

So we basically take the Prometheus data and train a model based on that data, store it in the repo, and create inference models to adjust certain parameters to achieve the optimum result. The correlation of traffic to CPU usage, to power consumption, to network traffic on each individual core on a core cluster, all that could be done in the model itself.

And then after the model is trained, and that inference could be made, you can basically provide input to MDCAP to take specific actions across the worker pools that are running the traffic, apply it to all the elements in the inventory that are hosting that worker pool, and make changes to the workflow engine to then take corrective action on power. So we build the power management workflows

in MDCAP that take actions, for example, that can bring down the frequency to a lower level, shut down certain cores, or even bring down certain worker nodes if they're not needed over a certain period.

So all that corrective action could be done in terms of an MDCAP workflow and applied in a distributed way across all the elements of the cluster, and getting to a point that the AI inference actions are driving how power is consumed across the entire network stack functions end-to-end. So this gives you a capability to do dynamic adjustments of power consumption across all the elements of network service, including gNodeB, all the way to Core, or anything in between, and achieve a level of optimization that would be very hard to do otherwise.

It's important to consider that doing all of these things manually is a non-starter. The amount of-- everything needs to be very well-coordinated, and it should be non-disruptive. So it's important to make sure that a tool like MDCAP could be used to orchestrate things automatically, and it also has to be dynamic. When the network traffic comes back, you have to now restart the pods that you shut down, or change the power settings on the servers that you modified, and so on. So everything has to be bi-directional and to have a feedback loop. As these changes are made in the matrix components that we collected, they're going to be used to adjust the models, and in a closed-loop fashion, we can now infer what are the best settings in the underlying infrastructure across all tunable components between any available paths that we have to achieve a dynamic workflow that optimizes the power across, as the consumption of network changes based on the network traffic. So that gives you an overall picture of how this capability could be dealt, and that's exactly what we demoed in MWC in Barcelona last time.

So I'm going to switch gears a little bit and talk about some of the other avenues of optimization that we have done with this technology and talk through some of those capabilities as well.

Some of the same concepts that I talked about here could also be used to do other functionalities, which I take through. The benefit that we're able to see, by the way, in terms of these power management workflows executing is that we think that we can get to an optimized energy consumption model that could be applied to the network services in such a way that we get guaranteed energy savings over a period of time just simply by normalizing the power delivered-- the power used to deliver those network services.

So if we think about this challenge with scaling, for example, similar actions could be taken, with the process we just talked about, by enabling a scaling model for delivering network services automatically. So think about the environment that we have the UPF running across multiple workers and on a cloud-native platform. You're dealing with changing demand of network service, and for certain peak moments, you now try to figure out how scaling could be managed.

A similar process we could follow, we can collect platform metrics, app metrics and cluster metrics from each individual node, aggregate them into AI models, train those models to understand where additional resources are needed, draw inference from those models. The online training could be done, and the offline training could be done on the models dynamically. And once we have the additional interference models, the actions could be given MDCAP to create additional actions to scale the network functions. So not only the power consumption of a network service could be updated, you can actually, for example, derive a workflow that creates a brand-new instance of the cluster by adding another UPF to do what we call horizontal scaling to the same cluster, or actually create a model that you can do horizontal scaling by...

At certain points in time, for example, at this point in time, you can have-- you can say traffic getting to a point that I need additional resources, you can add additional UPF to drive the higher level for the second scale, and then in the next step, you can actually add a brand-new cluster and horizontally scale the UPF function across the infrastructure, and once this workload diminishes, you can take the reverse action by moving the UPF back or shutting down that cluster completely, removing it from the infrastructure, and shrinking back the resources that are being used to deliver network services back to a normal level.

And because Kubernetes is so manageable in the sense that a workload could be instantiated very quickly, parts could be started with the relevant network functions, across available customers based on standard org charts seamlessly. A lot of these automations are very practical to achieve based on AI-based input. I think the primary factor I'm making about scale success is to have these tools for the

development of the workflows for taking those actions. These workflows need to be tied to a specific deployment of your network services. So they're dependent on the infrastructure back-end. So it's something that the operator needs to do, based on how the Core is deployed, and what the rest of the infrastructure is doing.

Other than that, the underlying science of managing power and managing scaling using AI models is very well understood, and this is something that could be simply achieved with the process I've just talked about. The benefit of having something like MDCAP that does all this orchestration automation for you is because we have a repeatable, easy-to-manage workflow engine that can take actions across the cluster components in a unified manner, and in a repeatable manner. This allows you to do hyper-automation, using available data sitting in the actual models, that makes the whole job much easier in terms of being able to do adaptive power management or adaptive scaling.

As you scale, and additional workers could be added to the cluster, now you have two independent clusters running network functions and you can deliver significantly more throughput, and flatten down the workload that is handled by each individual cluster. The cluster that is running the peak workload could be temporary and you could remove it once you get to the end of the game, and bring it back on when you need it.

Now, I want to chat a little bit about the last use case we had in the slide, and this whole concept of predictive network function scaling and healing. There is a lot of capability now in the current hardware that’s being deployed in the data center on the Edge around predictive maintenance. The reporting could be made in terms of disk failures that could be predicted, network card failures that could be predicted, and so on. The question in the table is that can we use some of those data coming out of the OBF layer on the hardware itself to take corrective action on the platform, and improve the availability of network service.

So the process is somewhat similar. The event, for example-- We use MDCAP to deploy Intel-based servers on the Edge or on the Core to try it for observability and make sure that all the hardware matrixes that are generated from the hardware level is also collected in the Prometheus endpoint, and there's the relevant data that could be captured around how the hardware is forming and achieving, and the same model could be now stored and created, and you could get to a point that you can now recognize and alert that, for example, a network interface card is going to fail, and they're having intermittent issues or challenges, or you have a situation that you have a disk drive that could be failing and so on.

And now, the question is, how do we use the AI models available to decide at what point corrective action needs to be taken, and those actions could be now passed on to MDCAP, and MDCAP can basically use a certain policy engine to say, okay, for certain conditions of predictive failure that we have detected in the model, I would like to basically shut down that specific node or vacate it, or take additional corrective actions. It’s the first thing that you could do. For example, we can call on the node, meaning that no additional resources or cards will be allocated to it. So Kubernetes will basically segregate the node and we can then execute the network function healing workflow to try to move the network function for the impacted node, the one in purple, to an available worker someplace else. So that UPF function could be now moved from worker number one, which is having a problem, to worker number three dynamically. This obviously gives you a much better path in maintaining availability and doing predictive action before an actual failure condition could arise. And many of these hardware-based reporting functions are already available from the Intel platform that could be used to create these models.

Workflow relocation is seamless in the sense of Kubernetes distribution can take a specific workload and its configuration, and vacate one node, and move resources into another in a very seamless manner, and as such, you will minimize downtime prior to a node having problems. There are obviously scaling and healing capabilities within the platform itself. So you can take actions within a cluster, but many other times you can move the node to another cluster.

Another thing that we can take on is this whole idea of storage monitoring; disk systems that are capable of reporting failure conditions and bad sectors very often. That information could be used to now build a model that suggests, okay, there is a reasonable chance that certain drives could fail on a cluster. That information could be used again to call on the node and then take corrective action around that specific event.

It's also possible to take corrective action to preserve the data. The CNP platform, for example, is capable of taking frequent snapshots. The snapshot's basically a logical picture of all the persistent volumes and Kubernetes configuration. Those snapshots could be now pushed into another node on the cluster using a backup feature, and restoring-- the data application, including this persistent data, could be restored to another node. That gives you the ability to move one workload from running on an individual cluster in the underlying infrastructure into a brand-new cluster while maintaining its persistent data volumes, as well as configurations. So this allows this whole action to be automated, from the point in time that you detect that there could be a likely scenario of a failure from a drive to a point in time that the workload is restored in an alternative cluster, and operations are continuing. So this kind of automation, obviously, improves the availability of the entire network function and automates a lot of the tasks associated with failure, recovery, and so on.

So I hope, by these two examples, I gave you a very clear picture of some of the things that are possible with AI-based automation. I think I want to leave you with a couple of thoughts to consider before we go to the questions.

One is that it is important that you have an orchestration layer, like MDCAP, that can accommodate these automation workflows in a seamless fashion, with the understanding of the inventory of the installed components, and network functions, and infrastructure layer. It's also important to have an appropriate level of metrics collection across all levels of infrastructure, including hardware, to be able to properly monitor all the metrics associated with the service delivery, and take corrective action around it, and a data model that could accommodate those and ingest that data, and have effective inference done. Although these inferences are not complicated, you make corrective actions based on the correlation of certain elements, having all the facts in terms of the actual throughput required to how network functions are behaving across multiple nodes, and multiple elements, and multiple pieces of hardware is an important part of the success of this kind of deployment.

So we have a few minutes to try to now talk to some of the questions that are from the audience. So let me start with the first question. I guess the question here is, what are some of the use cases that give the most benefits of this type of automation?

I think it's an interesting question in a lot of different ways. One is that if you're trying to solve the problem of resource consumption, and you're in a situation where you're running a 5G workload and power is a key consideration, obviously, power management gives you unique benefits to-- the element of power management is really not impacted as much the infrastructure component itself, so you very easily can do this type of work with a variety of actions to be taken. The level of action and adjustments you make to the underlying infrastructure is also pretty adaptive. So I would say power management is the most interesting use case in terms of having the most significant benefit.

If I have to pick a second-best, I would say automation associated with the scaling and healing could also be interesting, especially on a complex network that is multi-tenant. If you're delivering, for example, network services on a model that you have defined multiple tenants or infrastructure users across this network service delivery, and you have a comprehensive setup, like a neutral host model of a network, being able to dynamically allocate resources to different tenants and change those would be also quite interesting, because it's a complex problem. It's a problem that requires adjustment across the network stack. And automating that using AI could be interesting.

Next question. A question about operational savings. Yes, actually this is quite an interesting question. We talked a little bit about power savings. We talked a little bit about how that could-- how you can take corrective action using AI-based tools to save power. But here there's also-- the question is, can this also be used to do operational savings? Obviously, a lot of things that operators have to deal with that go across things like change management, for example, or scaling and healing automation, all of those have an operational burden if they're not automated. So if you're using AI-based tools to, for example, take corrective action across... automatically recover from perceived potential failures that could be imminent, that not only improves your availability, it also improves your operational costs because you don't have to wait for a failure to occur, and then take manual action, restart a new pod, or do new things to start the environment all over again. So all those things could also result in operational savings beyond and besides the savings that you achieved

on infrastructure. I think that is a valid question in the sense that we're not just only talking about the power savings. There also could be operational savings and so on.

There's another question. It is actually an interesting concept. So what is required to be able to do some-- deliver some of these benefits on the infrastructure itself? And I think this is a very observable question. If you look at a lot of things that I talked about, they all depend on data collected by the infrastructure, including things like failure prediction data from the hardware, or reporting that is coming from the infrastructure around network traffic being delivered on-demand. The availability of that data in a timely manner to be placed in the model, and then training the model to be able to take appropriate corrective action is obviously essential. A lot of the technology on the back-end systems provides a lot of that information, but that information needs to be extracted, centralized in a repository, train the models accordingly, and so on. So there are dependencies around the availability of the data to make this capability real. But fortunately for us, there's many of the hardware being-- the modern hardware being deployed have that capability, and the systems are there to be able to collect that information, infer appropriate actions from it, and make use cases such as this here.

I think that ends our questions. If there was anything, any other questions, please put it on a question mark. If not, I wanted to take this time to thank you for listening in. Hopefully, you enjoyed the talk and got some ideas about how you can deploy these technologies to benefit your use case going forward. If you have questions, please reach out to us in the background, and we can definitely work with you to achieve the goals that you will have and discussions you might have. Thank you so much and have a wonderful day.

Moderator

Great. Thank you for joining us today and sharing such great information. Thank you to our audience for joining us today. And please don't forget to give our team a rating for this live recording so we can continuously improve the quality of the webinars. Thank you again, this concludes our webcast.

Mehran Hadipour

Thank you.