The Open Source Way

Episode 06 – Project “Kyma”– An Easy Way to Extend Enterprise Applications

SAP Open Source

Karsten Hohage: Welcome to The Open Source Way. This is SAP's podcast series in which we'll talk about the difference that open source can make. In each episode we'll talk to a different expert and we'll talk to them about why they do it the open source way. I'm your host, Karsten Hohage, and in this episode, I will talk to Krasimir Semerdzhiev and Valentin Vieriu about the “Kyma” project. We will learn what “Kyma” is, why we run it open source and what there is to learn from that project. Krasimir Semerdzhiev, or rather in short Krasi, because I'll have a hard time to call him by his full name all the time, is a true open source enthusiast and we're really happy to have him on board. He works as a chief architect in the SAP Technology Office where he leads a North Architecture workstream, thus defining the end-to-end experience for implementation partners, for so-called ISVs. He has spent the last 20 years in technology development in various roles and he is most of all a prominent evangelist for the adoption of cloud native architecture principles, internally as well as outside of SAP. Valentin Vieriu works as a community manager at the “Kyma” project. He gathered all-round understanding of how the web works from product management, user interaction, design, frontend, all the way to backend and DevOps. He's a passionate frontend developer and he's getting more and more excited about the possibilities that, most of all, machine learning algorithms offer in that space.

Karsten Hohage: Let's maybe get to the “Kyma” project here. Valentin, what is “Kyma”? Why is it called “Kyma”? Maybe in the short version for a start.

Valentin Vieriu: Yeah, sure Karsten. And, so thank you for the introduction. I think that the question regarding “Kyma” is the question that we all the time get. Why the name? It's a similar story with Kubernetes. In the sense that when they tried to name Kubernetes at the beginning, they had a little bit of a difficulty. What should the naming be? I think Kubernetes started from the Borg project which is not a very friendly name if you think about the history of the Borg and how much damage they did in Star Trek. So, you know, they had to do something about it. They wanted something unique. And it went to Greek words and Kubernetes is Greek. It's about navigation. It's a pilot. And it's about the sea. And obviously containers were already there, dockers were already there, so that the whole nautical theme started with this naming convention. So basically, when we started with “Kyma”, we also had to choose a name. And the nautical names are something that kind of started to become trendy and “Kyma” means "wave" in Greek. So we are pretty much the wave that you ride your application on, metaphorically speaking. That's how it started. And it started quite some time ago, I think. I mean, it was before my time. Krasi actually was there at the very beginning, right Krasi?

Krasimir Semerdzhiev: Totally, I mean, it was an exciting aspect back then and we hit exactly the dilemma that you mentioned. You know, how to find a name that's in Greek from one side. And then here's the nautical theme. We hit a couple of ideas back at the time. But at the end we all got around the “Kyma” idea. And we like it so far because actually when you go to Greece and then you walk around, you find at least like 10, 15 different taverns which are called the same way. So it gives you just kind of a whole new feeling when you go there in the Mediterranean.

Karsten Hohage: Taverns that are all on beaches, I assume?

Krasimir Semerdzhiev: Yeah, of course. So people need to have this positive association with the whole thing.

Karsten Hohage: That's a good positive association because of what you said before, Valentin, I now have "resistance is futile" in my head all the time from the Borg. But now, we've got the name down. What is “Kyma” and what did it originally set out to do?

Valentin Vieriu: Well, “Kyma” is an open source project and it's built on top of Kubernetes, which is also an open source project. And Kubernetes is quite, let's say, low level. It just takes care of our containers, scales up and down your application, makes sure it is resilient. I don't know, maybe scales up regarding traffic but you need more than that for an application to run, you know, at least for a cloud native application. And back at the time, we had our own challenges. So the “Kyma” project started very close to the team that developed the Commerce Cloud. We were having our own challenges regarding extensibility, what would be a good way to extend our products and Kubernetes were showing up. So then we built this product in mind, “Kyma”, with very clear intentions of you need to run your applications somewhere. At the beginning was the idea of extensions for our commerce. And obviously you need to have all the tools necessary to deliver a cloud native application. So you have all of our logging, tracing, all the observability topics. And then obviously you need to have some sort of a service mesh. Then you need to have, maybe, a serverless workload. We identified the main topics.

Karsten Hohage: Let me maybe interrupt you there. If I wanted to put that short: “Kyma” is an extension platform and it provides you with all the services you need right underneath your application or your extension to the application, right?

Valentin Vieriu: Well, it started with that in mind but it's more than that. It's also what you can call a kind of a runtime where you run your application and it gives you all the things around. So that application runs successfully, you measure it and then you can make your decision. And actually, the application inside should not care about the infrastructure around. Kubernetes is not enough for that. And when we start with that, we realize that there are so many options to choose from. I don't know if you ever heard about Cloud Native Foundation. It's a foundation that actually tries to sort these things out and bring a little bit of order in chaos. But at this moment, in this Cloud Native Foundation, there are around 1500 plus projects. If you’ve ever seen the cloud native landscape, it's something that, at least in our industry, we are aware of. It's thousands of applications. So all of them solve a very particular problem. And the initial part was: ‘OK, which one of those? Which is proper? Which is corporate grade? Which offers security mature enough?’.

Valentin Vieriu: So for a developer, it's very hard to start picking. You have to spend a lot of time around that but you want to build your app. So what we did is, we pulled together; cherry-picked, what's the best in monitoring, what's the best in logging, what was the best in service mesh and so on. So we cherry-picked them one-by-one and then we created the package, and that is what's inside of “Kyma”. And also, we had to run, you know, write our own code to kind of make sure that all those work together. We created some obstruction layers on top of those, because sometimes you never know if the technology will succeed, and then you want to have an abstraction there to make sure that if you want to replace it, or someone in the future, you could do that. So we introduced some abstraction layers, which are relevant for application developers. We put all those things together in a package that works together.

Karsten Hohage: So “Kyma” is, in the end, a subset of the things relevant for building cloud applications, cloud extensions, a subset of services from the Cloud Native

Computing Foundation? I don't know if I should ask that, is “Kyma” itself part of the CNCF?

Valentin Vieriu: Actually, we have a small spot in there, if you go and you look for “Kyma”, it's in the serverless section. So I think there's an entire story towards that. We wanted to be part of the Cloud Native Foundation but we were so general and so big that we were not solving a very particular niche. We are part of the landscape in the serverless part. So we are one of those 1500 projects.

Karsten Hohage: The next question I always have on the list here is really, why do we do it open source? I don't even know if we still have to answer that. I mean, we are re-using a ton of open source services and being part of it all? It's pretty obvious, right? But maybe there are some more details to be said about that.

Krasimir Semerdzhiev: You nailed it, Karsten. So basically, since we're all based on open source, we obviously are true believers in that way of doing things. So that's why “Kyma” is not only using and is based on the open source project, it's also doing open development in the classical sense of it. So all the development we do is happening in GitHub. All the issues are there, all the check-ins are there. So we don't have like a hidden repo where, you know, we do stuff and then we flush it out. No, I mean, we actually work out there in the open, so that has been the defining approach for the project from day one. The main reason why we do it that way is that “Kyma” is not for us. I mean, “Kyma” is actually for all the people out there that are trying to work with our products and that are trying to customize and trying to kind of tailor them and to help our customers differentiate and so on. So ideally, all those, you know, countless partners and companies and developers out there, they need to have the feeling that they have a handle on the stack that they put next to our products, that they can influence it, that they can evolve it, that they can put enhancements in there and that they are really in control. So if, for whatever reason, some of our functionality is not sufficient, people can take “Kyma”, can take the source code, can change it, can add on top of it in order to meet their business goals and timelines and needs. So they don't need to basically file us tickets and feature requests and so on and then wait for us to solve them. That's not how software is developed these days especially that low in the technology stack.

Karsten Hohage: OK, now I understand that it is and why it is and how it makes sense that it's an open source project. How exactly is it SAP's extension platform or whatever? How does it tap into the SAP world? Like if you want to put that on a couple of explicit spots.

Krasimir Semerdzhiev: So “Kyma” is an extension, extensibility platforming in a box. So you can take it, you can put it next to anything and you know, obviously we developed it with idea that you put it next to SAP but we've seen people putting it next to many other things, like payment gateways, like robot automation, facilities in warehouses, where certainly there is some SAP somewhere in the landscape. But, you know, we are not necessarily envisioning exactly those scenarios when we develop “Kyma” but that doesn't stop people from doing so. So the way how we handle that is that we build connectors in many of the different SAP products that are connecting out to “Kyma”, registering their events, registering their API services, therefore becoming part of the picture. In there, in “Kyma”, then you can write applications which are triggered by events coming from one or more applications and then consuming API services, resulting in one or more applications. So this gives you the chance to build this event-driven integration and then react smartly to different things that happen in different places of your IT landscape.

Karsten Hohage: Ok, so Krasi, thanks, I understand it's totally not limited to an SAP world and not limited to extending or registering SAP applications or services but is just truly an open source platform for that.

Krasimir Semerdzhiev: So at some point in time, we even had an openSAP course which showed how you can use “Kyma” to extend WordPress. Because basically what WordPress was, was our kind of image for something that's truly open and truly widespread. And we wanted, when you educate people on how to use “Kyma”, we couldn't think of anything better to extend there that is more widespread than that. So that's why we stuck basically “Kyma” next to WordPress and showed people how to build image enhancements, text classification, sentiment analysis and so on, using “Kyma” and the events that were able to extract all the WordPress. You can find out on open.sap.com as one of the courses there.

Valentin Vieriu: Yeah, but I mean, Krasi, “Kyma” is a product targeted to enterprises, so, you know.

Krasimir Semerdzhiev: Yes, of course it is.

Valentin Vieriu: Itself, it consumes a lot of resources, just all the things running around, so that was some of the criticism that we got from time to time. But people need to understand this is not something that you just run your WordPress website on.

Karsten Hohage: So I don't run this right next to my self-hosted WordPress, right? Not that I'm self-hosting WordPress, but…

Valentin Vieriu: Oh, you can run it on your machine that's not the thing but it just consumes a considerable amount of resources to run all those services on the side compared to your workload. So it becomes valid when you start having more workloads that need to scale up, corporate-sized kind of projects, but it will run on your machine also.

Karsten Hohage: What else apart from, you don't necessarily want to run this at home, what else did we learn from the “Kyma” project?

Krasimir Semerdzhiev: Yeah, well, one thing that we learned is that we see a lot of people picking it up and then running it on premise, so next to systems that they run there. And we see that in two flavors: On the one side, people that do have a cloud strategy and want to make a jump towards cloud services. They take “Kyma”, they run it there and then they use the time at their own pace and their own schedule to essentially peel all the different customizations, put them into “Kyma” so that once they are comfortable with that, they can make the leap towards cloud. You know, putting all the customizations somewhere and then move towards a cloud service and stop running their own prime systems and maintaining their own prime data center. So that's one thing that we've seen people doing. Another thing that we've seen people do is also running “Kyma” in cloud environments next to existing cloud services which do not offer flexible enough customization abilities and therefore opening space for “Kyma” to serve as the event-base integration in between. Another thing that we learned from running more than a thousand “Kyma” clusters ourselves is that running something at scale is not the same as running an individual instance. Or basically to use your example, running “Kyma” on your machine. You know, that looks slightly different if you have a thousand machines to handle in the same way. So we learned a lot of things from cost and footprint perspective and that small things that, you know, seem negligible in your machine actually, when multiplied by thousands, actually starts to make a difference and then bubble up in priority when you're looking to where money is going and you know where our costs are growing.

Karsten Hohage: Now, maybe explain that to me. “Kyma” doesn't really in itself know yet what's going to be happening based on it, right? How do you scale something like that anyway? Does that happen totally dynamic? Or do applications or services that build on “Kyma”, do they have to like pre-register how many resources they might need or something, or how does that work?

Valentin Vieriu: Well, I think when Krasi said about thousands of servers, maybe we need to mention that we, I mean at SAP, we do run “Kyma” in the cloud platform part. So we have a managed “Kyma” runtime there, where we also use it ourselves internally. And also, we have a free trial where people are coming from outside and just run it. So we also have clients that run it there besides the open source. When we say it's very hard for the open source part to realize how many clusters there are. Actually, we cannot because you download the software and put it there. It can be closed but the number of clusters that we monitor now is the one in the SAP environment and we noticed that. Regarding the workload and some of the learnings, I think was quite interesting where we start running at the scale. We start seeing, exactly, as you said, some of the advantages of having those extra services around. We wanted to make it leaner and so on. So those clusters run individually. Each account has its own “Kyma” instance running with everything inside. And this is the case for every client. It's kind of it's own isolated world. So when it comes to scaling up and down, it depends on every individual workload. So the whole mechanism is based on Kubernetes but also obviously some of the internal mechanics that you have, you know. You can define how scaling happens for your cluster, depending on CPU memory, depending on workload, on requests. So it’s very individual. But “Kyma” offers the possibility for you to tweak and tap into all those mechanics, which Kubernetes offers and all the other services around it, to allow you to have that scalability.

Valentin Vieriu: But I think, Krasi, maybe you can pick it up more because you're more aware of the technical part than you are aware of the internals.

Krasimir Semerdzhiev: Yeah, the other interesting challenge is when you start managing multiple and thousands of clusters at scale, then you very quickly figure out that the problem that we ended-up building “Kyma” to solve. The custom code running in our commerce, Hybris commerce stacks, on premise and in the cloud and so on. So that was kind of the original problem which pushed us towards releasing “Kyma”. We fell into the exact same trap. You know, now suddenly we end up having custom code in large quantities running in the same cluster where our stuff is running in. So therefore, suffering from all the same issues that we ended up solving for our commerce colleagues, like noisy neighbours and resource starvation and all those kind of typical challenges there. So now we actually have to pull ourselves out of the water, further segregating their custom codes, isolating it properly, also with dedicated resources to make sure that “Kyma” stability itself is not impacted by behaving or misbehaving custom code that gets deployed in there. And the resources for the substantial infrastructure services are still available and can always be consumed as needed.

Karsten Hohage: We were just now talking about what you called, in short, keeping the core clean, right?

Krasimir Semerdzhiev: That's correct. That's a major, really major theme.

Karsten Hohage: And you just said originally, “Kyma” set out for applications to be able to keep their core clean, and now you talked about keeping the “Kyma” core itself clean, right?

Krasimir Semerdzhiev: Correct, so that we didn't go as far as allowing anybody to stick any piece of code in between any two lines of code in “Kyma” like some of the enterprise software that we hear people are using. So from our side, this gives you a massive amount of flexibility. So you can really do anything you want. But this is really a nightmare for upgrades, for updates. People end up delaying their upgrades and updates because of the cost, the sheer cost of doing the upgrade and retesting all the scenarios that potentially could be impacted on top. So that's why we are currently exploring what are better ways to extend the enterprise process and enterprise software side-by-side with “Kyma” running there based on events, based on APIs, based on replaying event logs. So essentially finding other ways to allow the same level of flexibility, while making sure that all the systems are naturally evolving and are kept secure and are kept up to date and can benefit from all the innovation where all the vendors in the industry actually are actively working on.

Karsten Hohage: OK, so that's one important aspect of things to learn from the “Kyma” project, “Kyma” actually collected some of the good fit services from the Cloud Native Computing Services portfolio. How does “Kyma” manage the dependencies that are generated through that?

Valentin Vieriu: Yeah, well, that's a very good point, and I think we learn a lot from the from our experience now in production and as we are very close to our main stakeholder project, in a sense, I mean, our main client right now, the biggest one definitely is SAP when it comes to “Kyma” and some of the feature, the requests that are coming from that are very interesting. So we are probably looking into the future of changing and evolving anyway. Kubernetes is constantly evolving from one version to another. New features are there, so change is definitely a constant. But we put some gates and we are very proud of it. So we introduced some things which are called custom resources. So we have a little bit of an API on top of Kubernetes or at least a higher level, which allows us to hopefully easily replace and change some of those underlying services. So we are watching the market all the time and we are seeing, you know, what are the trends, who's performing better, who is faster? Because the decisions that we made two years ago, they definitely are not the best ones right now.

Valentin Vieriu: So we are starting to change. But as we have this layer on top of it, the people that are using “Kyma”, they don't have to worry about that as long as they are using our interface. We also offer support when it comes to upgradability, so you can upgrade from one version to the other. And for you it should be decently transparent if something underneath changes. So it's hard to say right now. But definitely I mean, speed and making more streamline, allowing your class to be more utilized by your work. Some of those are definitely scenarios that we have in mind. And we also, having in mind for SAP running this in an efficient way. So what I can say for sure is the mechanics that we have in place are allowing us this, and we are monitoring and making sure that we choose the best strategy and the best tools that are out there.

Karsten Hohage: OK, but when you're saying you're choosing the best tools and now there is one name you mentioned in all of this that was Kubernetes, I don't assume that tomorrow you will say, oh, here is something that works better with containers, we'll just exchange Kubernetes.

Valentin Vieriu: I don't know, I mean Krasi, we could do that technically, right?

Krasimir Semerdzhiev: Yeah, potentially we could do that but it's a bit unlikely, at the current point of time. So when we started with “Kyma”, Kubernetes was kind of the safe bet because we needed something to turn everything in a box and Kubernetes satisfied all those needs pretty nicely. We were pretty sure that we wanted to benefit from all the other features of a service mesh, you know, end-to-end tierless encryption with different authentication plug-ins with fault injections. So basically, all the features which back then Istio was providing. So Istio itself is not a CNCF project and it doesn't matter. I mean, we still decided to bet on it for the time being. Is it really mandatory? Well, I mean, potentially you could also run without it but then you're losing all the benefits out of that. So basically, if your project is relying on all those features and doesn't want to kind of boil the water from scratch, then using Istio in there is a good idea. We use it in many places there in “Kyma”. Another thing to mention is that many of the components that we bet back then on like Envoi and so on, the only grew and only gained more traction and only gained more validation and so on. So, we're really happy with the technology choices that we made two years ago so far.

Karsten Hohage: And this all happening in an open source environment. Does that kind of mean when you need something from Istio, from whatever, you also contribute to these projects? Are you like an active exchange, basically?

Valentin Vieriu: Oh, yeah, we do have a contribution more in the past now, lately, we focus a little bit more on our product but we do contribute and they are important contributions that we had to other projects. So that continues, for sure. And just what I want to make sure for this open source solution, once you get started on your local machine, you can put whatever you want on the side, it's still Kubernetes underneath. We suggest just some set up. But if you would like to use another technology, definitely you could do that and it's decently easily replaceable some of them. So yeah, we contribute, we hope that we can contribute even more in the future. That's how open source works. You maybe need a feature you try to influence and you submit the pull-request, it gets accepted, then for you it's also good if you don't have to maintain a fork or something like that. So until now we do have our own customizations that work with “Kyma” but they are in an extensible way. So we are trying as hard as possible to not create any type of forks or custom code on top of the existing projects.

Karsten Hohage: And maybe just like from that question going to this, very overarching "we're open source here topic" for just a second there. From your experiences, is it easier that way with the contributions both ways or is it easier in an in a hierarchically organized structure inside of a company to get what you want?

Valentin Vieriu: Well, I think it's both, I would say definitely. I mean, we do have a structure in place which allows a clear decision process. You need to propose a topic. So sometimes I actually can take even longer than in a hierarchy. So we have the rules,okay? You make the proposal, it needs to stay there for three weeks and until everybody sees it. Now in our case, most of the developers are in the team. So we communicate quite directly. We have some external contributions but we would love for it to be more. So I think in projects like Kubernetes, where it's quite bigger, the problem becomes more obvious. So in our case, I would not say that it slows you down but it can slow you down if you're in very, very big projects.

Krasimir Semerdzhiev: Yep, there is a simple saying there, that if you want to reach far, you run with others. If you want to run first, you run alone. So basically, since we are here, we are here working on that for the long run. I mean, we fully understand that we cannot do that alone and we need to do that in the wider consensus. Because I mean, as well-established as SAP is, it still needs all the input from all the people down on the ground who are on a day-to-day basis innovating on business processes, implementing their customers scenarios, you know, differentiating in industries and so on. So we need all that kind of input and expecting that this is going to formally come in the sense of, you know, requirement documents and so on. Open source makes that so much easier.

Karsten Hohage: Before we totally get into the call for contributions here, which we had a tendency to do here already, let's maybe turn back for just another minute. And you mentioned the WordPress example, which is more like a lab example that you did.

What kind of real life uses, as I said before we get back to contributions, what kind of uses of “Kyma” you see so far?

Krasimir Semerdzhiev: So starting with SAP itself, I mean, we do have our own App center, so sapappcenter.com. It's based on our Commerce Cloud product, naturally, and then it has “Kyma” on the side. So where a number of integration flows are implemented there as extensions, event-based and then following the paradigm that we defined in “Kyma”. We do see a number of other scenarios in the commerce space around, like marketing, newsletters, generation for users, personalized experience, product recommendations there for particular users tailored, based on their history in the store, doing fraud detection there, analyzing activities there of a certain user in the store. I mean, all of those things are events that get emitted there out of our commerce stack, which we can react on, aggregate and then compute their potential index there for issues, which then triggers other activities that analyze the situation and suggest remediation. We've seen people doing image optimizations, minimizing them, filtering out unfocused images, doing stuff like automatic image sorting there and image, recognition of what's coming in. So a lot of scenarios which if we have to kind of sit down and implement ourselves, is probably an unthinkable amount of effort for our small team. But, you know, now with the tooling that we put forward, people out there can take that and go and innovate, which is what we after.

Valentin Vieriu: I just want to, I mean obviously this is not a commitment or anything as I'm passionate about the machine learning, I contacted some internal teams from outside, then we're talking to them: "Hey, would you be interested in trying to run some GPU machine learning inside of “Kyma” and try that?". So that could also be something in the future where you could use this for. So those are more like labs and PoC-level but interesting options are out there as using as a workload.

Krasimir Semerdzhiev: That's how every innovation starts. I mean, you need a good idea.

Karsten Hohage: How can people get involved in “Kyma”?

Valentin Vieriu: Oh, that's great, because I would say that would be my main, let's say, call for action for this. And what we need for a healthy project is, we need a healthy number of contributors internally but also externally. So I'm looking, as a community manager, all the time at those numbers. So we do need external contributors. We need fresh ideas. We need the people to propose features also to pick up and and solve them. So the best place to start it is our slack channel. It's an open source project, so support is not guaranteed but our developers all are there and answering the questions and everything. So it's a very clear structure per topic. So you should just, if you have something with serverless, you go into the serverless channel and you ask a question. When it comes to GitHub, that will be the best way to look at for issues and also for feature requests. We do have this interesting structure because we were talking about structure and how we do it. We have like two organizations. We have an incubator and then we have the main organization. So ideas, if you're just starting and you want to try something out, you can propose a project in incubator. You know, people get excited about it. They start contributing, get some mature state where it can be picked up in production, then just moves into the into the main room.

Karsten Hohage: So the incubator is the one for the creative chaos that I just mentioned before, right?

Valentin Vieriu: Exactly, and there, you might end up with different, a little bit lighter version, no different structure but as you get into the main repo then CI pipelines are there, automation, everything which you have to provide, otherwise this doesn't scale. So GitHub definitely would be the second part. Any feature there, creator issues, every day, every week we actually have a filtering and we are prioritizing a lot. Like top priority is contribution from outside because we want to encourage this as much as possible. We do have our YouTube channel where we post most of our meet ups. By the way, we have a monthly meet up right now. We tried a biweekly one but we switched to a monthly one, where basically we come and present the new features and we discuss important topics. As I said, the feature requests and everything that goes first in GitHub, enough time for people to react and ask questions. We also bring them into the meet up, discuss with them. So all of those are open forums for people to feel that they can contribute. They can feel that their voice can be heard. So those are our main topics.

Valentin Vieriu: We have, in the openSAP space, an initial “Kyma” course there that we launched. And I think there are a couple of other resources where more focus to the managed offerings which will happen as education: one on the SAP blog and community blog and developer community. There is also our colleagues from the developer community. They are also creating content. So this happens. I was kind of trying to push more for the open source part but all of those are accessible. But some of those are very specific to SAP which also there's a lot of learnings there for sure on those channels.

Karsten Hohage: Seems like a lot of places you can start in, at least. Now as a final question, maybe. Now that we've listed all the means where people can get in touch with “Kyma”, what are your main messages that you want people to take away from this podcast today?

Krasimir Semerdzhiev: So from my perspective, 2-3 things that people should take away from us, is stay curious and stay hungry. Also stay up to date of what's going on in the industry. So there is a lot of innovation happening all over the place, multiple projects. It's really hard to follow. But basically, there is a lot of progress made on multiple fronts. And as long as you're clear of what problems you want to solve, for typically most problems there is more than one solution available that you can choose from. So just pick what works best for you. It worked quite well for us so far and we're really happy with how it developed. Another thing is, yeah, don't build monoliths and keep the core clean. So as much as possible, try to separate custom code out of the product that you're responsible for that you have to provide SLAs for and that you really want to evolve and update and patch and upgrade as soon as possible. As soon as you have something running, I think you should be able to flush it to all of your installations and don't leave anything behind. So don't fall into the same trap that that we did. Try to learn from our experience and to get to kyma-project.io. So that's where all of that is nicely summarized.

Valentin Vieriu: I will also add something from the open source part, and I would say, you know, it is possible to do open source, like real open source, even in starting from a corporation the size of SAP. And I think that as probably this will be one of the target audiences - SAP interested people and also maybe some of our colleagues - I would say you could try to see how easy it is to contribute and how easy it is to create a pull-request and a feature, how easy it is to, you know, if you would use this in your internal team, how easy it would be to propose some changes and make sure that those are going to be a part of the main code as we move forward. And I would say it is possible to develop open source projects and use the size of your corporation, as we do with “Kyma”. And I think that would be one of the main takeaways that I would like to push for.

Karsten Hohage: OK, thank you, Valentin. Krasi, did you want to add anything else? Oh okay, you fully agree? That's a nice addition and final word. You fully agree. Great. So thank you, Krasi. Thank you, Valentin, for being our guests today. It was nice to have you here. I hope everyone out there is, like, totally interested in working with “Kyma” now. Thanks, everyone out there for listening to „The Open Source Way“. If you enjoyed this episode, please share it. So the next one is probably not due before January 20th, 2021. And in the meantime, whatever you celebrate, have great holidays and hear you again in 2021. Thanks for listening in. Bye bye.