The Open Source Way

Episode 03 - InnerSource rocks

SAP Open Source

SAP

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 we'll talk to Michael Picht from the SAP Open Source Program Office about InnerSource. We will discuss the advantages of an InnerSource approach and why InnerSource should be the default development model for global enterprises and quite a few things around that. My guest, Michael Picht works as a chief architect at SAP. Prior to that, he worked as a developer, software architect, project, program and product manager with SAP Application Development with a focus on supply chain management, business processes and innovation topics. In the SAP Open Source Program Office, his focus areas are processes, tooling and InnerSource, of course. Hi Michael. Welcome to The Open Source Way.

Michael Picht: Hi Karsten, thanks for the invitation.

Karsten Hohage: You're more than welcome. Before we tackle InnerSource, is it correct that one of the activities outside of SAP was also, or actually, had to do with open sourcing information? Is it correct that you are a food blogger?

Michael Picht: Yes, I'm a food blogger. At least I used to. So I think I stopped several years back, but 10 years ago or so, we were quite successful with our food blog. And it was a lot of fun, of course.

Karsten Hohage: And food-blogging in that case doesn't just mean you posted pictures of food but you actually posted your recipes, right?

Michael Picht: Yes, exactly. So I like to be creative, also when I cook. I cooked, took pictures before we had lunch or dinner and then posted the recipe together with some pictures.

Karsten Hohage: But for all I know, of course, the recipes didn't go under an open source license or anything, you just simply posted them.

Michael Picht: No, at that point in time, I was not that much into open source so I did not know anything about open source licences, I just published it.

Karsten Hohage: I addressed that first because I happen to know that the actual open source projects you're involved with in your outside SAP time are rather exotic and may not be as readily understandable as sharing food recipes, right?

Michael Picht: Yes, yes, they are quite technical and maybe for some people a little bit exotic, you're right.

Karsten Hohage: Would you care to share them anyway?

Michael Picht: So I'm a fan of Arch Linux. I'm working on some tooling for the package management of Arch Linux and then, apart from that, I work on some other tools that I need, for instance a music conversion program and some other stuff. I'm just starting tools that I need and they are not on the market so far. So that's how I come to open source projects.

Karsten Hohage: So you're competing in many fields, it seems. You're competing with SuSE and RedHat Linux there, having your own Arch Linux distribution. You're competing with the master cooks of the world by sharing your recipes and you're competing with the sound engineer of this podcast here by involving a media server software, right?

Michael Picht: Exactly.

Karsten Hohage: Ok, well, let's maybe turn to our topic: InnerSource. What exactly do we mean when we say InnerSource?

Michael Picht: InnerSource is about applying learnings and best practices from the open source world to the in-house development. So in practice this means that, if there is a project team that develops a certain software inside a company, this project opens up and is open for contributions from other teams. Traditionally, you have assigned dedicated project teams to certain software developments and the software is developed only by this team and nobody else. And in an InnerSource approach, this is more open and other teams or individuals can contribute bug fixes or features. This is what InnerSource is about, basically. And via this you can increase the collaboration. So

many, many companies use InnerSource to increase the collaboration inside the company. You can also avoid double development and you can increase the reuse inside your company. So many advantages.

Karsten Hohage: OK, I see the advantages for the overall organization. Why people from these other organizations, why would they contribute to a project of others?

Michael Picht: Now, often they are consumers of these teams so they use the software of these providing teams and many of these central teams, they have resource restrictions because they are not only working for one consumer but for many. And they get a lot of bug reports and feature requests. And due to the resource restrictions, they cannot fulfill everything in time. And then these consuming teams come in because they are potential contributors. And by contributing, they can, of course, speed up the development of new features and speed up the bug-fixing. And so it's a win-win situation, in fact.

Karsten Hohage: It's basically these other contributors, they speed up the delivery that is targeted at them?

Michael Picht: Yes, exactly.

Karsten Hohage: OK, cool. I understand that. Do they use all the same tools as open source projects if it's run in InnerSource – GitHub and everything, or is it totally different?

Michael Picht: No, no. They use the same tools. GitHub, for instance, or GitLab, so you need some collaboration tool. You also need a tooling where you can submit change proposals or pull requests how we call it on GitHub or GitLab. Because otherwise it would be quite chaotic. Otherwise the providing team, if they open up their software, everybody could do the changes. Nobody is reviewing these changes. This would be quite chaotic. So you need a structured process. And to support this, you need some tools and contributing teams submit typically a pull request which is nothing else than a change proposal. And then someone from the providing team, he or she is then reviewing this pull request. And if this is OK, then this is supplied or incorporated or merged into the original project.

Karsten Hohage: Ok, so even though that happens within the same organization, also the processes are all the same or is there still a little bit of a difference to an actual open source project?

Michael Picht: I think the processes are very similar, if maybe the same, but at least similar because you have this review process in place in order to avoid this chaos that I was talking about and in order to make sure that the software remains with high quality. This is quite important.

Karsten Hohage: I guess apart from the fact that as long as you stay within, in our example SAP, you don't have to worry about licenses, right?

Michael Picht: Yes, that's correct but there might be companies who have different business units and who need to think about this because if one business unit contributes to a project of another business unit. I don't know. It depends on the contracts and on the legal setup of the company.

Karsten Hohage: OK, I was just going to say, that's more the legal setup of the corporation or the organization we're talking about. Some are divided up in different limited corporations and so on rather than being on one. I understand. OK, cool. And InnerSource is that a term that we coined or is it generally called like that or who invented it or...

Michael Picht: It was not an invention of SAP, unfortunately. As far as I know, the term was invented by Tim O'Reilly in the year 2000 and everybody is using it now. This is really the official term for what I just described so how this is working.

Karsten Hohage: OK, and so this has been around since the year 2000. Is it very widely used or is it very limited to the core IT industry?

Michael Picht: It's widely used not only by IT companies. So many, many companies are using it. Companies, I don't know, banks and broadcasting companies or from the media area, but of course, also IT companies. I think IT companies were among the first who used this methodology. But as you know, also companies coming more from the

traditional industries like electrical engineering or mechanical engineering, most of their products contain our software so they had to become software companies. And this is the reason why they also, in the meantime, use a lot of InnerSource, just to increase the collaboration in their company. So if you talk about the companies, those increased collaborations are one of the main reasons, in fact, why they use InnerSource.

Karsten Hohage: But we usually apply that and talk about InnerSource only when it's about software. It's not like we're seeing companies that InnerSource the construction plan for the next highway bridge or something, right?

Michael Picht: Yeah, typically, InnerSource, the term InnerSource, is used in conjunction with software but the concept of InnerSource, in my opinion, could also be applied to non-software projects. For instance, we are running inside our company a group called InnerSource Ambassadors which has the task to drive this topic forward inside SAP. And we are running this also as InnerSource and we are not producing software. So this team does not program anything but we are also driving this in an InnerSource way because it's a virtual set up. Everybody can contribute. So I would also call it InnerSource.

Karsten Hohage: OK. And can you, just because you said InnerSource a couple of times again now, can you just put your finger on what's the difference to a project organization rather than a line organization? What further does InnerSource really take it then being a project organization?

Michael Picht: I think you referred to this matrix organizations where you have the line organization and you have the project organization and then different people from different lines come together. So the management, of course, commits to these people that they can come together and jointly work on the project. This is what we typically call matrix organization. This also can happen in InnerSource, especially if you contribute bigger features to a project, because then you need a certain management commitment also for the maintenance of this feature. But InnerSource also happens on a much more detailed level because every individual can contribute bug fixes or can contribute small features. And maybe the managers of these developers do not even know about that. So you do not need a big management commitment for this. You can also do it on a very small level. This is what I would say is the difference.

Karsten Hohage: OK, so in one term, the "no commitment" of resources is the difference.

Michael Picht: Not necessarily. As I said, for bigger features, if you contribute to an InnerSource project over a longer period in time, you take over some more responsibility which might be required for some bigger features. Then you need a management commitment for that because otherwise the providing teams, not contributing teams, they cannot really rely on the consuming team that they can take over the maintenance. So at that point in time, you need some commitment from the management. But InnerSource also happened at a much smaller level. Individuals contribute small bug fixes or features and then typically you do not need a big management commitment.

Karsten Hohage: OK, but I take it from your last words that sometimes or in certain phases, the "no commitment" can be the problem that is included in InnerSource.

Michael Picht: Yeah, this can be the problem.

Karsten Hohage: OK, is there any special phase when this mostly occurs?

Michael Picht: I think it's between the team who is providing the software and the team who is contributing to this project. I think they need to align before the contribution happens whether a bigger commitment is necessary or not.

Karsten Hohage: OK. And how about when things go into maintenance, like when development has finished, and we go to operations?

Michael Picht: Typically, if we talk a smaller commitment, a smaller contribution, a bug fix or feature, after the review has been passed by the contribution then the team who receives the contribution takes over the maintenance. This is typically the case. What sometimes happens, especially if this is a bigger contribution, then for a limited time the contributing team takes over maintenance. But I think finally the team who is actually responsible for the project takes all the maintenance. This is what happens typically. But

it really depends on the size of the contribution and on the setup. There is not a one-fits-all answer to this.

Karsten Hohage: Yeah, I was just going to say it might be different depending on different setups of the projects. OK, great. Let's maybe turn to you. You deal with this from out of the Open Source Program Office. What is your job? What are your tasks concerning the InnerSource activities?

Michael Picht: As I said InnerSource takes over some ideas from open source. So after we had founded the OSPO two and a half years ago, we took over the whole thing. First of all, we built up a network of InnerSource projects and contact persons. Because we had some other urgent tasks to do at that point in time, we did not drive it forward very actively. But after some time, teams came to us and said: “Hey, we heard about InnerSource and we would like to apply it but we do not know how this can work and how we can start with it. We need some consulting. Can you help us?”. And then, because before that we had built up our network, we knew a lot of people from InnerSource projects already. We had a meeting with them and discussed: “Hey, now we get these requests. What can we do about it?”. And then we came up with the idea to start this group, what I already mentioned, these InnerSource Ambassadors and run it also as an InnerSource project. And via that, driving InnerSource within SAP forward – this is what we did. So my job is to facilitate, to work together with this group on driving it forward. In practice, this means that we work on how-tos, on documentation, on some guides that we can give development teams at hand as to how they can start with an InnerSource project, what they need to take care of, some checklists and so on and so forth. We are doing presentations inside the company to talk about InnerSource and to maybe inform teams that there is something like InnerSource that may be beneficial for them. But we also are in contact with external organizations like InnerSource Commons. We are doing presentations there. And because this was the original request, we are also consulting teams on how to become an InnerSource project. This is basically what I'm working on.

Karsten Hohage: OK. And in the end, these InnerSource Ambassadors you talked about, you are one of the ambassadors yourself?

Michael Picht: Yes.

Karsten Hohage: Are you kind of a Primus inter Pares, a first among equals or how can we understand that?

Michael Picht: First of all, I'm one of these so-called InnerSource Ambassadors. I think currently, we are a group of 12 people or so and I would not consider myself as a Primus inter Pares. We have a weekly sync meeting and I'm hosting this. This is the only special thing that I am doing. But apart from that, I'm a normal InnerSource Ambassador, I would say.

Karsten Hohage: OK. And now as ambassadors of course, you said one of your main roles is consulting projects that want to do InnerSource. Are you actively pushing people to "Hey, why don't you do it InnerSource?" or are there automatically enough projects that, from within themselves, want to do that?

Michael Picht: I think we are doing both. So sometimes we are pushing teams or what means pushing, we cannot force them, of course. But for instance, if you have some central services that are offered, some very central software which is offered inside SAP, then this is always a good candidate to become an InnerSource project. And sometimes we contact these teams and ask them: “Hey, did you ever think about becoming an InnerSource project? Would this be interesting for you? We can consult you”. And so on and so forth. This is what we are doing. But many teams come to us actively and ask for support. And then, of course, we help them.

Karsten Hohage: And about the projects that are run InnerSource...like as a potential contributor, how do I find out about them?

Michael Picht: In fact, this was one of the challenges that was reported to us by developers. So last year we did an open source survey inside SAP and we also had some questions about InnerSource in the survey. And one thing that came out was that one of the challenges is for developers to find InnerSource projects. And this is what we worked on in the past to improve the situation. So we did basically two things to improve the situation: One thing is in our internal job portal, we have a specific position type for InnerSource projects. So as a project lead of such a project, you could offer a position. You could, for instance, say: “Hey, we have this feature that needs to be developed and

we assume that it takes a developer 10 percent of his time over a time period of 4 months”. And this is the description of the position. Then developers can look into this portal, can filter by position types. They can filter for this InnerSource position and then could apply to this if they are interested.

Karsten Hohage: So that would lead, as you described it, to the committed contributions as we called them before, right? Or would that still be considered uncommitted, free InnerSource, whatever?

Michael Picht: So I assume. And this is also what we recommend to developers who apply for such a position, to align with their managers. So this is what you would call a committed contribution. Yes.

Karsten Hohage: OK. And you named the central services. What comes to my mind is the kernel services. For instance, they are probably a good candidate for something like this. Is that right?

Michael Picht: Exactly. And this is, in fact, also one of the teams that we contacted. So we thought in the InnerSource Ambassadors group the kernel services could be a good candidate to become InnerSource. And then we contacted them and two of these services decided now to try it out. And they became InnerSource projects just recently.

Karsten Hohage: Mm hmm.

Michael Picht: And now they would like to make some experiences. And if this is positive, if they are convinced, then I assume that also the rest of the kernel services will become InnerSource projects.

Karsten Hohage: So when you said just recently, we don't have an idea yet if this works or if we are getting sensible contributions here or whatever?

Michael Picht: No. They just did it a few weeks ago, so I would give them, let's say, three or six months to make some experience and then we can have a discussion with them.

Karsten Hohage: And is that the only type of thing where we can feel or is that the main characteristic of projects that can be run InnerSource – being a very central service that everyone is in the end using, like the kernel services? Or are there totally other projects as well?

Michael Picht: There are also others or there could be other projects but these central things, they are really a good candidate because they have a lot of consuming teams and these consuming teams, they are all potential contributors. And in the end, if you would like to start with InnerSource or if you would like to become an InnerSource project, you do not only need the project but it's also crucial to have contributors. So therefore, these central projects are always good candidates.

Karsten Hohage: OK, but are there also other examples?

Michael Picht: Um, let's think about it. Um, no.

Karsten Hohage: No?

Michael Picht: At the moment, I do not have good examples for this.

Karsten Hohage: I admit at this place that we have, of course, talked a little bit before and you mentioned things like a chat bot. Would that also be considered such a central service, or...?

Michael Picht: Yes OK, the chat bot. We have this central chat bot tool within SAP. It's called Tobi and this is also very central because many teams are using it. We have another tool that we use inside SAP. It's called Bridge. It's a framework that offers different services for the developers. For instance, you can see the lunch menu in the different canteens or you can do a time recording via Bridge. And it has a highly modular architecture. It's based on plug-ins. And they get contributions. So they get plug-ins as contributions. But this is also something I would consider as central.

Karsten Hohage: OK, maybe I was thinking the wrong way. Central, yes, but the kernel services if I drew it in a simplified architecture, probably are very much deep down. And

the chat bot is more on the user interaction side. That was why I was asking about it that way.

Michael Picht: OK, this is the difference between both. But if we look from a reuse perspective, both are very central because many teams are reusing it.

Karsten Hohage: OK, I understand. Where do we stand now that we found out we're doing kernel services, we're doing the chat bot, we're doing a couple of other things. Where do we stand as SAP on our way to actually becoming an InnerSource organization or an organization that truly embraces InnerSource? Let's put it that way.

Michael Picht: An organization that truly embraces InnerSource, so when the default development model at SAP would be InnerSource, then we have reached our goal, I would say. We are not yet there. We made a lot of progress in the recent years. So currently I think we have several hundred repositories which run with an InnerSource approach. As I said, we made good progress but we are not yet there. But what I've seen over the last years: the InnerSource trend is accelerating. Many teams come to us and want to have consulting. So entire development units switched to an InnerSource approach. Just recently, one of our development units, I don't know, 200-300 people, switched completely to InnerSource. I really see the application of InnerSource increasing within our company.

Karsten Hohage: And which one was that team that made the complete switch?

Michael Picht: It was CX. I do not know what it stands for.

Karsten Hohage: You don't know what this stands for. Maybe we should include this to the links under the podcast, then once we publish it or something, you can find out.

Michael Picht: I think it stands for customer experience.

Karsten Hohage: Customer experience. OK, but that doesn't have to do with UI or…?

Michael Picht: I think they also have UIs.

Karsten Hohage: OK, but as far as I know, the UI department, or the UI5 people, they are also doing a lot with InnerSource, right?

Michael Picht: Yes, they are one of the pioneers, I would say. I think they do InnerSource for almost 10 years and they are quite experienced. And also, some people from them are members of the InnerSource Ambassadors by the way. They, one year ago or so, inner-sourced their complete documentation. So UI5 is a very basic UI technology which is used by many SAP applications. And so these applications are, as I said already, potential contributors and the InnerSource team wanted especially to improve the documentation by taking into account the contributions from the users, by the consumers. And this worked out quite well.

Karsten Hohage: We have now also established that InnerSource is not always only about sources as in source code but also sometimes about documentations and collateral products, basically.

Michael Picht: Yeah. We have to distinguish the documentation. The example that I just made, UI5 documentation, this was so to speak the software that they were working on where they accepted contributions. But documentation is also of high importance from another perspective because if you would like to get contributors, you have to make it as easy as possible for them to come into your project and to learn how things are working and what your project is about and how they could contribute. And therefore, you need a very good development documentation for your project because otherwise it will be very hard for contributors to really contribute to your project.

Karsten Hohage: Which is basically a similarity to open source again where you have the same thing, right? Structure your code well, document well and so on. So people can even have a chance.

Michael Picht: Exactly.

Karsten Hohage: OK.

Michael Picht: The other important thing is communication. So sometimes some traditional projects – I exaggerate now – they meet in the coffee corner, five people, and

make decisions. Nothing is written down, nothing is documented. This does not work in an InnerSource approach. In an InnerSource approach, similar to open source, you typically use asynchronous communication like mails, messenger, slack and so on and so forth. You document decisions so that people who come into the project later on see why certain decisions have been made. And this is also quite important if you start an InnerSource project, to really change the way you communicate.

Karsten Hohage: Mm hmm. OK. And from an organizational standpoint, also, is there like a, I don't know, when you kind of think of InnerSource as open source. What open source requires is a large pool of people who may be the ones contributing to exactly your project. Do you think InnerSource needs a critical mass, a critical minimum size of organization before it can work?

Michael Picht: Yeah, I think so, indeed. Just thinking about a small start up company, I think they are anyway using an InnerSource approach because if you have 10, 20 people working in this company, they know what they do anyway and they have a good collaboration. But typically, if companies are growing and become bigger and bigger and bigger, maybe you also acquire other companies and then suddenly you have these different development units. We can call that silos. Then you need to do something in order to foster collaboration. InnerSource could really be a good approach.

Karsten Hohage: So you'd basically say InnerSource is one of the means that can help you deal with complexity?

Michael Picht: Yes, exactly.

Karsten Hohage: OK, cool. What's the criticism? What are the issues that come up when you do something InnerSource?

Michael Picht: The classic is, of course: InnerSource will never work within our company. So this is the most ethical one. If we look at projects...when we approach projects and ask them: "Would InnerSource be an approach for you?", sometimes they say: "Ah, then I get a lot of low quality contributions and our quality goes down. We cannot risk this". But as I said at the very beginning, you need to have a structured process how you accept contributions or reject them. And via this you can manage the

quality. Some teams are afraid of the upfront investment that is necessary. You need to work on the documentation. We did discuss this already. You need to, maybe, refactor your code so that it's easier to understand. You need to come up with some automated testing capabilities. But this is something which is for the overall benefit of your project, independent of becoming InnerSource or not. This is really what you should do anyway. And so it’s not really a negative thing about InnerSource.

Karsten Hohage: And we're, of course, not here to mostly talk about the negative things and scare people off. I just remember in our last episode Sebastian Wolf said, or someone in this podcast said, for every hour we invest, we get two hours back.

Michael Picht: I think this is a learning from the open source world. This might not be true for any project, but it shows...at the end a project can also benefit, or should be able to benefit, from an InnerSource approach.

Karsten Hohage: As you just said, not true for any project. Can you summarize a list of parameters that a project should have or characteristics to be a good fit for InnerSource?

Michael Picht: First of all, we said this already, it should have a good documentation, it should have automated testing. We need some collaboration tools so that we can manage the contributions. Then it's necessary that you have contributors at all. So if this project is done in a very exotic technology and maybe in your company, you have only five people who understand this technology, it's very unlikely that you get contributors. What's always a good candidate is projects that are consumed by others because these are potential contributors. What is also an advantage is if the project has, or the software has, a modular architecture because then you can distribute responsibilities much easier. And I made the Bridge example. Other teams could contribute entire modules or entire plug-ins. This is an advantage. But this does not mean that for more monolithic projects, InnerSource wouldn’t be a good fit. Just for modular projects, it's much easier.

Karsten Hohage: OK.

Michael Picht: So these are some of the characteristics.

Karsten Hohage: This is basically the call-to-action to everyone – check your project. If it fulfills many of these criteria, then you may want to think about going InnerSource. What's our goal for InnerSource at SAP?

Michael Picht: Our goal is to make InnerSource the default development methodology at SAP. It is clear to us that not every project can take over this methodology because it simply does not fit for them due to the criteria that I just mentioned. Sometimes there are also some legal or contractual boundaries. Because of them you might not be able to share your source code inside the company. You must keep it confidential. This also happens sometimes. But for those projects for which it's really possible, it makes sense. Our goal is really to convince them to become an InnerSource project.

Karsten Hohage: So basically, for everyone where there are no clear reasons against it the suggestion is the following: At least do part of it InnerSource, right?

Michael Picht: Yeah, exactly.

Karsten Hohage: OK. Or would you personally say you would want to make it the default development model?

Michael Picht: I think I don’t have the power to do this but I'm of the opinion that this should be the case. InnerSource should become the default developing model at SAP.

Karsten Hohage: Then here is your chance again if you want this to be the default development model; what are the one, two or three key takeaways to now wrap this up? We are reaching our targeted time here. What are your one, two or three things you want everyone to take away about InnerSource?

Michael Picht: First of all, if you are a project team, you're responsible for certain tool, a certain software, think about InnerSource. Maybe this is the right methodology for you. You can get contributions. You can get in closer contact with your consumers. You can understand their requirements better. And maybe contributions help you to have a faster time-to-market. This means to deliver your stuff in a quicker time. So really, think about becoming an InnerSource project. Vice versa, if you are a consumer of some software

that some other team does and maybe you are tired of waiting for your features to come. You sent them a lot of requirements, but it takes a lot of time for them to realize it and to implement it due to resource restrictions. Maybe you can convince them to become an InnerSource project and you can then contribute your stuff and get your stuff quicker. This would be number 2. But we also have to speak to our management. If you are a manager and responsible for a bigger development unit, maybe it's an opportunity or a possibility for the unit to switch to InnerSource because via InnerSource you can have a much better collaboration inside your entire development unit. You have a better utilization of your development resources. And I think you can also increase the motivation because you can provide developers with the opportunity to broaden their skillset if they can contribute to other projects. I really think that this could be a win-win situation. This was my point number 3.

Karsten Hohage: OK, great. Your almost last word was the win-win situation or maybe even win, win-win-win-win situation for quite a few roles that you mentioned there who would benefit. Thank you very much Michael for being our guest today. It was great to have you here and thanks all for listening to The Open Source Way. If you enjoyed this episode, please be sure to share it and don't miss our next episode in around two weeks. We share this podcast not only on openSAP. You can find it in all your regular podcast channels like Spotify, Apple Podcasts, Google Podcasts, etc, etc. Thank you all for listening in and thank you again, Michael. Hope to hear you next time. Bye.

Michael Picht: Thank you. Bye bye.