EPISODE 1529

[INTRODUCTION]

[00:00:00] ANNOUNCER: This episode is hosted by Jordi Mon Companys. Check out the show notes to follow him on Twitter.

[INTERVIEW]

[00:00:07] JMC: Hello, Heather. Hello, Pablo. Hello, Scott. Welcome to Software Engineering Daily.

[00:00:11] HM: Hello.

[00:00:12] SJ: Hello.

[00:00:13] PRM: Hey, hello.

[00:00:15] JMC: So, I'll introduce each one of them in a minute. But let me actually dive into the topic that brings us together today at Software Engineering Daily. The topic has actually design collaboration. It's somehow tangential to the core topic of software engineering. But I think an interesting one nonetheless.

In mind, we review design collaboration, also known maybe as design management or design operations, is the act of collaboratively building software with design as a central element of it along with source code and other assets. I first spotted this trend fairly late while working at GitLab a few years back. We spotted the rays of the product manager. Let's call this persona as a way of synthesizing different roles. And this person was able to collaborate on code via pull requests, merge requests with code reviews, code approvals and even merge conflicts. Why not? But this same person was actually in charge of product related activities and workflows, right? That required versioning, updating, discussing, iterating upon other types of digital assets, right? Like images or any car kind of data structure that is not text-based like source code.

Source code collaboration with version control systems and source code management systems like Git and the services built on top is mostly resolved. It's not an easy thing. But the workflow is proposed by the main Git providers are fairly standardized and have solved the problem at large. But the same cannot be said about other digital assets that are not text-based. Again, images, animations, video and so and so on.

And this is only talking about design with a small d. But when you think about design with a capital D, the problem extends even further, right? Design with a capital D includes research, making hypothesis, testing those and even working with other types of tools and workflows, right? And so, whatever design management may be, the recent acquisition of Figma by Adobe for the price of 20,000 billion dollars is for me at least evidence that the space is not only economically very valuable and potentially profitable, but a space where open source software is expanding to. We'll explore the case of Penpot later. But the opportunity is just great.

With that excuse, with the acquisition of Figma to sort of like pinpoint this broader topic, I've brought with me the three guests that are on the show, on the Software Engineering Daily Show podcast today.

With me are Heather Meeker, open source licensing specialist and general partner of OSS Capital. OSS stands for Open Source Software. **[inaudible 00:03:26]** claim actually is, and I quote, "Open source is eating software itself much faster than software is eating the world." And we will have time to discuss that even if it's tangential to the core of the topic today.

She's also the partner in crime, if I may say, of Joseph Jacks who has been interviewed several times in this very podcast. And even a former colleague of a colleague of mine at GitLab, Scott Williamson, if I'm not wrong. A recent incorporation.

Scott Jenson is a veteran designer that has been involved in numerous extremely popular products at relevant companies. Think of QuickTime and Apple Newton at Apple, to mention a few. Physical Web, Gmail Mobile at Google, Symbian, many others. And he was one of the thought leaders, if I might say, in this space that drew my attention as I mentioned in the beginning and also drove most of my thinking in this specific area.

And finally, Pablo Ruiz-Múzquiz, the CEO and Founder of Kaleidos. The company which is, and I quote, from their claim to, "A great hybrid between a technological partner for startups and an incubator for employees' idea." Examples are **[inaudible 00:04:44]** Taiga, a project management project. You will have time to describe it briefly, too, Pablo. And Penpot, which is the project that is actually of most interest today.

Himself, by training, is a physics major. Hopefully not a theoretical physicist. Am I wrong?

[00:05:06] PRM: You're right.

[00:05:07] JMC: I'm joking. Okay. I'm joking, by the way. I just bought a book about the the Paul Dirac who was a really popular theoretical physicist and was not crazy like most of them. But he became a software engineer by trade, and an entrepreneur and many more things that we'll learn later when he introduces himself.

I'll start with Scott. Because as I said, he I think is one of the most prominent evangelizers of design in general. But the design collaboration conundrum that brings us all together here. And it hasn't been always easy. Scott, would you actually define in better terms what design collaboration is? If you're comfortable with that term? Or any other that you might find more adequate? And your whole process sort of like approaching it and approaching even open source communities about it and so forth.

[00:06:08] SJ: Well, that could be a very large topic. So, let me just try to keep it really short and we can ask follow-up questions. But the design process has multiple levels to it, right? At one level you're like, "Oh, these icons don't work, right? Let's fix the graphics."

But the more deeper ones is how do we start the program? What's the first thing we show the user? What's the information architecture and so forth? It's a wide range of things. There's a reason why UX designers often significantly overlap with product managers because we're making product decisions. That's part of the collaboration process. You touch all aspects of the product.

What I particularly started talking about recently, and I've spoken at **[inaudible 00:06:48]** and other conferences about this, is the role of this design collaboration in open source. Because when you just talked about GitLab and how these people that are doing all this design work are also doing checking code, that's not a good thing, right? They do that because they have no choice. The only way for designers to work in open source is to become programmers, right? We shouldn't be programmers. I mean, we should be cognizant of it and obviously aware of it. But the only way –

For example, just recently, I suggested a change to Mastodon. The new hot open source thing to talk about these days. And I did an icon change just as a test, right? I mean, just changing an icon is trivial. And the answer was very polite. It was like, "Oh, well, just go build it yourself and then do a pull request." And I'm like, "Are you kidding me?" That's what design collaboration is in many cases, is that you must be the programmer.

And so, it's a little bit of prisoner's dilemma, right? You have to become like your captors, right? And so, what I would argue is that as open source matures – because in for-profit software, which has lots of issues. I'm not trying to say – designers don't do that, right? Designers are their own specialty. And they work with people who actually coordinate.

My broad point, and I'll wrap up, is to say that design collaboration is best when the team understands it as a centralized person that organizes it and everybody does what they're best at. And because we're so source-code-focused and GitHub-focused, or Git-focused, that's often a challenge. That's my quick summary.

[00:08:29] JMC: And to add to that, would you say that consensus-driven organizations/projects are also not the best suited for this? Because you said centralized?

[00:08:38] SJ: The issue with open source is not that it's consensus. To me, the biggest constraint with open source is that the programmers are so scarce. What it means is that you're so desperate to not lose your programmers that you will do nothing to piss them off.

And so, there's this kind of, "Oh, my God. If you want to work on it, great. But I don't want to offend you, because you may go." And that creates a very weird dynamic where you can't upset the programmer. And I'm not suggesting you upset programmers. I'm just trying to say there are hard decisions to be made of products.

And because open source, everyone wants to scratch their own itch, you tend to get everybody piling everything into the product. And you can't do that and have a good product. And that upsets a lot of people in open source. And managing that decision process is very tricky.

[00:09:31] JMC: I think that's what actually programmers usually call being opinionated. But I also agree with you that it tends to be a good thing in my view. I'm also a product-minded person rather than an open source developer. Last question to you before I move on to Heather. What does the acquisition of – Actually, the raise, the popularity of Figma, one million active users last time I checked. And the acquisition of it, not necessarily in financial terms, but from a design perspective me. Because, I mean, 20 billion dollars is obviously, for any of us, a huge amount. But what does it mean for someone that works in design collaboration, or at least partially, that a product like Figma has become so prominent and important?

[00:10:23] SJ: Well, I want to be really careful here. I really like Sigma. And I really like the team. I think they've done an excellent job. And I think a lot of people are not upset with Figma. They just have concerns about Adobe and what it might mean to the overall ecosystem. And these are all fears. They're not really obvious just yet.

I mean, what we've seen recently with a Pantone color problem with Adobe is just kind of an indication of the kind of concerns that are showing up. What it highlights to me is the fact that people are saying, "Well, wait a second. We can go to an open source solution that just gets rid of all the drama, right?" It's just no longer an issue anymore.

To me, I'm trying not to say that the Figma acquisition is evil. But what I am saying is it's causing people to significantly look at alternatives like Penpot because they just de-risk a lot of these situations.

[00:11:14] JMC: So, Heather, Figma is a portion of the pie. And there was a post, a recent post, from your colleague **[inaudible 00:11:24]** Joseph that I mentioned before that says that it's only a piece of the broader avalanche, those are my own words, of open source products that are providing with tools to creative people, designers and so forth. I'm thinking of, well, Penpot in the case of Figma or design collaboration. But also, stable diffusion in the case of – I'm not sure how the category is called. AI generative? Okay. Godot Engine in the case of video game engines and so forth. Again, large pieces of software creation tools that involve not only source code, but also creative assets. What's your view from the venture capitalist and investment point of view?

[00:12:19] HM: Well, it was a very interesting transaction, of course, particularly because we're in, well, maybe I would say bad times. But at least I would say weird times economically. And there hasn't been nearly the pace of transactions that there was even a year ago. The acquisition was remarkable because. It was a big acquisition, it was I think Adobe's largest acquisition ever actually. And during a time where companies are not being acquired for huge multiples. And I think I read that the multiple was like a 50x multiple. That was huge, right?

So, one thing that's interesting about that is that, okay, we have a somewhat dreary landscape in terms of deals. And all of a sudden, this huge deal happens. We need to pay attention to why it happened in this area. And I think, as Scott was alluding to, the visual aspect of products is one that, well, in my personal view just needs improvement, right?

I'm very interested in user friendliness and user interface. And I think if you look at the products that have really good user interaction, those are often best of breed products. And programmers – and look, I was a programmer too. So, I get this. They tend to be a little more focused on functionality than user interactions. Anything you can do to improve the ability of developers to create like beautiful software as opposed to functional software is really, really interesting.

And of course, that is – Adobe has been a leader in that area for a long time. It was interesting that it was focused on that area. And for every investor, we need to be looking at what the market is telling us is valuable. That was interesting.

From the point of view of open source – being an open source investor or a commercial open source development investor, what was really interesting to me about it is the aspect of collaboration. You can build a tool that does something. But if you build a tool that allows people to collaborate on doing things, it adds tremendous value. And that sinks perfectly with our investment thesis.

Many of our portfolio companies are what are called open core. Meaning that there is a core of open source functionality. And then the business is built around deploying that functionality in a way that allows people to collaborate within an enterprise. That's I think consistent with the market signal here that the collaboration aspect is very important. It is one of the uh more effective ways to build a business around open source software is to have basic functionality that is open source and free. And then on a commercial level, provide the infrastructure to allow people to collaborate.

[00:16:01] JMC: I completely agree with especially the – I mean, both points. But the first one, too. And I would actually extend that to even when you – to the design with a capital D that I was referring to at the beginning. Because even with products that do not have an interface and even for products that are aimed at developers, and I'm thinking of, for example, command lines, right? Like, text-based input interfaces that do not provide with a lot of history unless you, well, request it through a command and so forth. They don't paint any interface. And so, even those, if they focus on developer experience, they require a lot of user research. In this case, developer experience, right? Because the user is a developer.

I would argue, there's a lot of research behind them that make them elegant and beautiful, as you said. There's not even a need for a UI. I mean, although, as you said, those that are driven by UI, and point-and-click interfaces and GUI's, those definitely require the design element to be embedded fully in it.

[00:17:04] PRM: Jordi, if I can just jump in. I just had a project where I did the UX of an API. And I worked with a team to have an API that was actually easy to use. This goes all the way down the stack.

[00:17:16] HM: Yeah. And developers are consumers and users, too. In fact, the ascendancy of software in our lives means that someday we're all going to be developers, right? If we aren't already. Yeah, of course, it's important for developer tools as well because developers are users just like any other users.

[00:17:45] JMC: The case of Kaleidos is extremely relevant to today's conversation because it's the company behind Penpot. And we will describe the product and, again, the problem it tries to solve from an open source perspective in a minute. But it's also a peculiar company in many ways. I'm not sure if a cooperative is actually the best way to describe it. But as I said before, employees are highly involved in the products that the company launches. And it's highly involved with open source. And it's also highly involved in, shall we say, developer tools? Project management tools? Because before we move on to Penpot, Kaleidos is actually the company that Pablo runs along with his colleagues. Launched before a project called Taiga, which is – again, could you describe it, Pablo? Try to solve for a sort of like a neighboring problem, right? Correct or not?

[00:18:44] PRM: Actually, I think it's the same problem. It's just a different angle. So, yeah. Kaleidos is an employee-owned company. It's not a cooperative. But it actually behaves like one in many ways. And we are not doing any consultancy products anymore like two years ago. We stopped that. We went all in with our open source products. Taiga being one of them, the first one. And then, of course, Penpot.

We saw this issue, big issue, with developers and designers working together. We saw that there was this trend and this ethos about being on the same page. Really working hard to sort out the same problems onboard, etc. Actually, developers respecting designers, designers respecting developers and all that.

We saw that because we were experiencing that eight years ago at home, right? But there was an issue with the best practices in how we would develop software. And that was the limb principles, and the Agile manifesto, and Scrum and Kanban and all the methodologies around that.

And since we have this obvious elephant in the room, which is that developers overrule designers. Like, developers have more power as a quite uneven distribution of power in sofwtare development, it happened that we imposed Agile to design. We said, "This is the way we're going to make products. We're going to use lean/agile, your **[inaudible 00:20:30]**. We don't care. And to make it easier for you, designers – Like, this was all about being sympathetic and all that. I mean, this was not like – It didn't feel like we were imposing this. This, what I'm trying to say. We created Taiga, which is an agile project management platform. It's open source. But it's beautiful to use, easy to use for designer. It was to onboard designers into Agile to make them welcome, feel comfortable and in charge to a certain extent, right? It was Kaleidos way, or Kaleidos developers' way to say you're welcome here to the lean/agile process.

And we were very successful at that. We actually collaboratively conceived Taiga as such a tool. And so, immediately, we were able to have just one conversation about the project, right? Everyone was on board. Users stories were codifies. Nobody had to like use this tool and then report elsewhere or stuff like that.

Everything was great for a time. Everyone was enjoying that. Like, yeah, we cracked that. Until we had this design is not able to scale up issue. Like, we were able to tackle bigger problems. Thanks to onboarding design into Agile. We were automating more stuff. Just leaving the human factor where it belonged where it made a difference. And we were actually making the ratio between designers and developers go up. In the past, we would have one designer per 15 developers. But now, it went to one to ten in the world. You can average data to one for eight. So, one designer eight developers.

At Kaleidos, we are one per two. So, it's one designer for every two developers. That's an anomaly that we're very proud of. But what that means is that it's tough to keep that an even distribution of power.

Thankfully, designers said, "Right." In this company, we're only allowed. Because it is our choice to use open source tools. We want to do that. But we're not being able with the current status quo of open source design and prototyping tools to really shine. To really be productive and enjoy our work. You, developers, have the luxury of choosing everything you want. It's going to be the leader or the challenger in that category. But we don't have that luxury. We're second-class citizens in open source. We're going to ask for that exception to the rule and use Figma, right?

And what we what we decided to do at that time is, "Okay, let's do that um temporary exception to the rule. But then we have to create the Figma killer." The open source. Not just any other proprietary Figma clone, but actually an open source Figma killer.

And what we did was to complete that journey to make developers onboard the design process, okay? So, we're not just –

[00:23:47] JMC: That was bold. May I say that that vision is bold.

[00:23:55] PRM: It's bold and it's quite challenging. Because, Heather, Scott, you already know. Creating a designer prototype tool is not just run-of-the-mill product. It gets complicated super fast. It's meta UI. Like, you have to let people be creative and bring Innovation. Not judge what they're going to do with the tool. But also, in terms of technology, it's quite challenging to have a performant tool like that.

The specific angle that we have is not only it's open source. Not only it goes with open standards. And we'll talk about that in a minute, because it is a text-based representation after all in a persistence layer with SVG, of course. But also, it brings further collaboration, true collaboration, between designers and developers.

My hidden agenda, not so hidden these days since we had this breaking moment with the Figma acquisition, is that, as a developer, I voluntarily give away power to the designer. I want them to have a bigger stance, more power, more leverage. And at the same time, I hope that any best practices around source, control, versioning and stuff around software development can be brought into the design process. It's a win-win, right?

It's sharing a vocabulary. It's sharing some process. It's basically the same way we had designers onboard in Taiga and saying, "Oh, actually, this is an Agile process I can enjoy. We want developers to onboard design process and say same."

[00:25:39] JMC: Now that you mentioned version control, it's actually – And source code management. It's actually the core of my experience. I've worked in several companies. For example, GitLab. But isn't that one of them main problems? Because it works really well, again, for text-based – Well, I mean, it might work for actually uh Penpot. But Git doesn't work really well with large files, right? It has this extension that I believe Microsoft donated that it's called LFS, Git LFS. Or large file system, if I'm not wrong. But the user experience – In general, the user experience of Git can get extremely complicated. If you add another module to it another extension, it can – It's got a clunky user experience to say the least. And it's an extension. It's not native.

And there are alternatives. I think the open source virtual control system manages well large files and – Yeah. But there are closed source alternatives like Plastic, acquired by Unity and – Plastic SCM. And **[inaudible 00:26:40]** to name a few. Is that a problem or not? What are your opinions on that, Scott?

[00:26:47] SJ: Sorry. I have a strong opinion on that one. Because I think what happens is that we're saying is, "Oh, source code is done through files." Therefore, we have a file based controls Git system. And therefore, the design must fit in that process.

And so, it's like really putting a square peg into a round hole, the fact that design must be file-based. I'm not trying to say that we shouldn't have tools. I think the steps that Penpot has made are excellent. But one of the first things that I'll do on a project is I'll put up a big white board with sticky notes and stuff so that we can balance and see things. And again, we could have digital equivalence to that. But my point is they don't easily fit into a file-based model. I'm just calling out that it's always going to be a little bit of a challenge.

[00:27:36] JMC: Pablo, you want to add anything?

[00:27:37] PRM: Yeah. I think it's not a fundamental issue that design cannot be always represented as a text file. I think what is an issue is that developers have to care about design, right? The moment we have developers caring about design – and we are trying to do that. This win-win scenario will only happen if the developers actually care about design, because they certainly understand the power collaboration and design per se. Then, I don't think there's going to be an issue. Because all the innovation, all the creativity, all the tech that we need, we will have it, right?

Now, having said that, Penpot does use a shortcut until that happens, which is very convenient, which is an open standard. Not just any industry –

[00:28:30] JMC: Can you, yeah, elaborate on that please?

[00:28:33] PRM: Well, this is about the web/internet standard or vectorial graphics, which is SVG of course. This is by the W3C. And you can represent vectorial information. It's math, basically, in a file, right? And quite potent.

And so, we are in this sweet spot where we don't have to make any compromises. We can have both right now, right? And we know that we can do diff SVG. Or we can do a lot of stuff. But I think – That is a nice shortcut. And of course, that means that a designer could actually just commit push on a Git repo right away from Penpot without using Git. I think it's a fallacy that Git repo, it's for the whole team. It's just a known failure. But let's not try to make it something that is not, right? Let's make sure that we have the right integrations with the productivity tools that the rest of the stakeholders in the team use. And in that case, that would be perhaps Penpot. Why not allow a designer to actually start a continuous deployment process just on a click of a button, right?

And also, I'd say this is a full duplex, why not have back, from the engineering perspective, data from a server to A-B testing output flow backwards or flow back into the design itself and just programmatically redefine the design itself? I think we are just in a – This is very initial really.

But I go back to my point, is that developers rule software development the software development too much. I mean, too much for my taste for anyone. And developers also rule over open source. We have those two problematic intersecting things there. Developers are ruling software and are ruling open source. The way to escape this is, somehow, in some aspect, they need to stop ruling some of these two camps.

[00:30:52] JMC: Who give away power? That would be my question. But, Scott, you probably have something to add. And I'll move on to Heather after that.

[00:30:59] SJ: Well, yeah, definitely. I just want to strongly just support what Pablo just said. Because that is the biggest issue, is that I think a lot of programmers, A, don't understand design and somehow feel that collaborating with designers is about power, right?

To me, power isn't something that each person has. It's what the group has, right? And if each buddy works as a team, we get more power, collectively. And so, moving from an individual concept of power to a group concept of power is exactly the kinds of things you need to have high-performance teams.

I just find myself getting beaten over the head by the problems of file systems. But to Pablo's point, it's because the designers don't understand that it's a shared system. So, we're talking at two different levels. But strongly agree with what you just said.

[00:31:57] JMC: Pablo, this is actually a pretext to ask something to Heather. I don't want to put you in an awkward spot. But Penpot was already popular before the Figma acquisition. But after the Figma acquisition, as I think Heather mentioned at the beginning, there was a huge transfer, or if not a new grow in user base. Can you share a bit of the data of that monumental spike in Penpot onboarding and sign ups? Maybe don't share anything that's confidential. But can you give us an –

[00:32:33] PRM: Nothing. Nothing is confidential, I think. I mean, we operate under open source ethos, and transparency is a key value. I think the only issue here is how much? What is the amount of things you can actually deliver, right? But it's not a matter of – No. I actually have a ton of data. But, yeah, it was already – Penpot was already successful and trendy. But we were having – I think this is – there's so many levels. The acquisition was a breaking moment. We were being very careful not to be overruled by developers initially, right? Because if you have the first open source design and prototyping system that there is, not that there are not all the open source tools that allow you to design or vectorial, of course. And we all use that. But the first one that had these collaborative aspects, etc., etc. And it was going to be easy, too easy perhaps, to have the developer audience onboard.

I'm a developer myself, a Python developer. I know that it would be very easy for me to lobby hard for Penpot, right? We wanted to make sure that it was a designer's first experience. Like, actually, this was conceived and built by designers for designers first. Even if we had the collaboration between designers and developers, like a key ambition, right?

That we had a balanced community. Sort of 50-50, right? Remember that we have one per eight, one designer per eight developers. That would be a toxic ratio to have initially for a designer's tool. I hope you take my meaning. Like, we wanted to make sure that we had a compensation early on.

We were having that. But the acquisition made that actually go much better. Because guess who actually onboarded Penpot after that? It was developers. They go their way slowly onboarding Penpot. It was designers, right? So, we are now actually having many more designers than developers. But it's not actually this 5.6K percentage uptick. Or, I don't know, hundreds of new Penpot deployments on-premise. Any incarnation of private cloud that you can imagine. But actually, the breaking moment is design is realization. This eye-opening moment where they understand that they cannot just go like this forever. They cannot just put all their IP, their effort, their work on a tool that is proprietary, that's not open source, that doesn't use open standards.

We thought it will take us two years to make them realize that. We know developers have already said that. But this was a massive acceleration of that mindset that we needed. We got 20 billion invested directly into making a lot of designers say enough is enough, right? And I can now see that this is a huge problem.

Developers would say, "I told you so." But now designers would understand the subtleties of all this. I'm not so excited about the massive uptick. And it actually is two orders of magnitude. Formerly speaking, yeah, I'm excited about the uptick. And it's not fading away, by the way. But actually, the breaking moment, not for us, but for the design community. If we had been just another cool Figma clone, we would have some stuff, some success. But the fact that we were completely different, coming from a completely different approach, I think served as well.

[00:36:27] JMC: We already know your reasons. Not Taiga. Kaleidos reasons for being bullish on open source. It's part of your philosophy and your spirit. But I I'd like to know, actually, Heather – So, Heather's actually also putting her money and other people's money as a general partner into general open source.

Although the conversation is about to sound collaboration in the problem space, I think I find it impossible, like Pablo just describe, to separate it from the fact that open source is eating that pie again.

You've touched already on this, Heather. But what is your general impression of, yeah, maybe open source bringing in the design community, which was kind of, I would like to say this, death to the cause of these sirens long ago.

[00:37:25] HM: Well, first, I'd just like to underscore something that Pablo said, which is he mentioned that their platform is based on open standards. And that's at least as important as the open source in a way. Because we've all seen the convolutions that creative tools have gone through when they were not based on open standards. And also, all the tools have been gravitating towards open standards. Because even the proprietary vendors realize they really can't sustain their own proprietary standards.

[00:38:03] JMC: Can you clarify why open standards and open source are different things? And why are those two things relevant separately?

[00:38:11] HM: Yeah. Open source software is basically software that's under licenses that allow completely unrestricted use of the software. I mean, that's a very simple way of putting it. But that's about freedom to use the actual software that you might use to create things.

There's a difference from an intellectual property point of view between the tool, the software, and what you create. Those two things are different works. We call them like works of authorship and copyright. They could have entirely different licensing paradigms. You can use, for instance, an open source tool to create something in a proprietary standard. Or at least there are ways to do that. And vice versa, you can use proprietary tools to create stuff that is in open standards. The open standards have to do with like the format, the way that the actual resulting work is represented. And importantly, whether it's portable across different tools. And that portability is it's one of the things that's great about open source because it tends to produce output that is portable and adhere to open standards.

For instance, it's like LibreOffice. If you use that, it's a pretty good product by now. It produces an open document format. As well as some of the more traditional proprietary-ish formats.

[00:39:53] SJ: Okay. I just want to underscore what Heather just said, especially because there is a great amount of activity and interest in bringing machine learning to all of these UX problems. And there's really two very big categories to this. One is using machine learning within the product itself. But another one is to bring it to the files.

And so, by having open standards, you basically liberate your data in a way that encourages a whole new category of machine learning, which is to run across your files and do interesting things to them. I think there's going to be a tremendous amount of creativity applied to this. And open standards for storage of your data is one key to unlocking that.

[00:40:36] HM: But I'd like to go back to the original question. I got a little sidetracked with the open standards, because I like that idea so much. But, yeah. I mean, our thesis, what we believe in and what we've literally put our own money in and others money into, is that businesses that are built around open source software are the future of business. And that's a pretty audacious statement to make. But we think that the software industry, the proprietary models are going to become more and more irrelevant because the expectations in the marketplace are to get access to source code and to have all the benefits of open source based tools. And the models, the traditional models. And I don't want to name names, but like Microsoft, and Adobe and so forth. I mean, these are the companies that built huge and successful products based on the proprietary model.

This is not the direction the software industry is going in. The direction that's going in is that an open core model is more interesting to users. It's easier for them to justify using the tool because they can't get locked out of maintenance and support. And and also, just the collaborative nature of it makes for better products. And that's what we truly believe. But more importantly, we believe that that's the future of the technology industry at at a minimum, right? If not everything else, too. Because these days, in a way, every company is a software company.

Everybody is developing software. It doesn't matter what sector you're from. And so, we're looking at this as its own sector, which will eventually just eclipse the sector that exists now, which is essentially proprietary software and SaaS.

[00:42:57] JMC: I tend to share the same vision. Also, I'm skewed – I'm biased to think in the same way because I've worked mostly in open core companies, in open source companies not all of my career. I'm not so bearish and – I don't have a big problem with close source alternatives. I just think that the open source is better. But I don't think closed source is going anywhere.

But anyway, I don't have strong arguments, to be honest. I'm not the biggest expert in that sense. But Pablo, you might want to add something to that.

[00:43:32] PRM: Yeah, I think– I mean, Heather's moto, like open source is eating software faster than software is eating the world. I think, for me, it would be easy to enrich that saying that design is eating software faster than software is eating the world.

[00:43:48] JMC: Oh, okay. Elaborate. That's really interesting.

[00:43:52] PRM: I think it's obvious that design has become a key differentiator per se. That more and more technology needs good design. That more and more different audiences do require thoughtful good design. And I think design is now – it was always like that since the Xerox innovation around design and interfaces of course.

But I think people realize that design is a key element in innovation, right? Any Innovation. Anything that technology will provide us with in the future. Perhaps not flying cars anymore. We don't want that. But all the other things. Design is going to be a key part.

I think – well, more than I think. I strongly believe that closed source design tools will not help that. You need accessible – universally accessible design tools for everyone on Earth to access. And we have powerful force on such innovation, whether it's innovation for global audience or for their own communities. But you can not just consider design as any random category in technology, particularly design, cannot be treated like neutrally.

Penpot, if anything, yeah, it's about collaboration between designers and developers and making sure that Scott doesn't receive that response about, "Yeah, just do a pull request." I'll say, "Okay, we are already –"

[00:45:27] JMC: There's a feature for that.

[00:45:29] PRM: I'm using Penpot. He will say, "I'm using Penpot. I can actually integrate my design to your source code." Like, fast forward five years from now. And that will not be an issue. But also, more people in the world will be designing, will be designers with the right tools.

[00:45:48] HM: Yeah, absolutely. One trend we're seeing in software is the low code, no code trend. And that makes a lot of sense. Because, I mean, every software programmer knows that you don't want to do all the – Like, all of the nuts and bolts. You want to actually probably do things at a higher level usually. And the world Is still lacking all the software engineers they need to create all the software that people want to create. So, the more that we can push that up to higher levels and also make it work in tandem with design, the more productivity we're going to get out of the software industries. That is also a really significant trend. You're seeing more and more pressure sure to allow people to use more powerful tools at a higher level. And also, to get people who are maybe not willing to sit around all night looking for a semicolon. Just the quintessential programmer experience.

[00:47:03] JMC: [inaudible 00:47:03] syntax and all – Oh, yes.

[00:47:08] HM: Yeah. I mean, to get more people involved to make programming more inclusive, it helps to have better tools.

[00:47:19] JMC: Scott, you wanted to add something to that?

[00:47:23] SJ: No. I think that was well said. I was just giving a strong thumbs up.

[00:47:27] JMC: Okay. Okay. Apologies.

[00:47:29] PRM: I would like to.

[00:47:30] JMC: Yeah, go ahead.

[00:47:30] PRM: Okay. Now, Heather just mentioned this low code, no code. I think it's a big trend. I tend to see that more on the category of the individual that needs extra powers, extra superpowers, and feel that they can be more productive in areas where they don't have the expertise. In a way, sometimes you would democratize design. Sometimes you can say you commoditize design. I think we have to be not fearful, but rather like understanding that these tools are mostly meant for the solo player, right? For the lone wolf that needs to cope with more stuff when there's no other team.

But there'll always be need for the high-performance teams. I think Scott was mentioning them. Like, you need collaboration. Not only having extra superpowers yourself. But collaboration is the key to really scaling up and being able to sorting out the big challenges, right? Cross-functional teams, like big teams, to be able to do that.

And I think we are in the space, like Penpot and Taiga, in the space of the latter, of the high-performance teams. That will only get bigger and in bigger numbers despite the fact that you might have more people joining the ranks of the low code, no code. And probably there will be some conversion from those areas into the high performance.

We are interested more into the conversation between different stakeholders in a team rather than giving some individual like extra capabilities for a short period of time so that they can do more just by themselves. I'm not saying that is not important. I'm saying that I feel it's important to understand the difference in the approach. And that is why we're discussing collaborative design or design collaboration. Well, collaboration requires more than one person, right? So, just to underscore that.

[00:49:34] JMC: Yeah. I'd like to actually do a shout out to two people outside of this conversation that support the views expressed here in the last few minutes. One is Tyler Jewell a analyst/managing director of a fund that supports what Heather just said. The output, I'm not sure if that's the correct word, of software engineers that universities and other trades are providing the world with is not enough for the demand of – So, the supply is not enough for the demand. Therefore, platforms, developer tools, no code, low code, whatever tools to create software need to be intelligent so that the output of those is bigger, right? Because the demand is going to grow faster than the supply of software engineers or anyone producing software, which is not necessarily needs to be nowadays a software engineer.

And I think Pablo mentioned about something that design is fundamental. I come from KubeCon. So, that would be the first shout out. The second one is to actually Redmonk, the industry analysts focused mostly on developers and in open source, which I think today is the 20th anniversary. Congratulations to Stephen O'Grady, and James Governor and everyone, Rachel, all the all the people involved in it, because it's a fantastic small firm. And they actually evangelize a lot about developer experience.

KubeCon North America last year – Apologies. Last week. KubeCon EU in Valencia, Spain in may. Developer experiences all the way. Again, products that have a lot of research, user research. In this case, developers behind it. And then the output of which is a collaboratively work through and implemented in the product is actually the trend that is defining, in this case, the developer tools industry in my view. The cloud native industry. I just wanted to highlight to sort of like third-party validation of point of view that have been expressed here.

And I think that, with that, we can maybe wrap up. I mean, I actually would like to throw out a question. Have we missed anything that the broad space of design collaboration is – have we missed anything fundamentals? Scott? Heather? Pablo?

[00:52:05] SJ: I just want to reiterate, which is new to me, which I learned in this conversation, is that I'm not thinking of design and software development both along a continuum. Where we've got this kind of hyper-card-like technologies at the bottom where we're encouraging people to do no code, low code types of things and then this high-performance type stuff that Pablo referred to. And they're both good. They're both, but it's interesting to think of it as a continuum and what each level needs. And that has been very helpful for me to think of it as that continuum.

[00:52:39] JMC: Heather, any last thoughts? Or Pablo?

[00:52:44] HM: I would just say that – Sorry. I would just say that there are a lot of other issues in the creative design space around licensing. It's kind of a quagmire kind of. It's really a quagmire. And that is one of my pet peeves. Obviously, much broader than the topic we're talking about. But –

[00:53:10] JMC: Do you want to open that can of worms at this point –

[00:53:13] HM: Yeah. But that's a problem that really needs to be solved and one that interests me a lot. Like, how do you clear rights to use materials and figure out whether you have the rights to use materials and stuff? Actually, people waste quite a bit of effort on that.

[00:53:31] JMC: I'll reach out to you to see if we can have a further conversation in future episodes about that and with other –

[00:53:38] HM: If you want a very opinionated discussion.

[00:53:41] JMC: Always. Always welcome. Pablo, I've got a question for you. Everyone joining Penpot is forced to learn SVG? Become a very – what is it? Verbose? A very fluent person?

[00:54:02] PRM: No. No. Not at all. First of all, let me say that I was hoping to cover – Like, Heather would cover that issue. She just mentioned, like the issue is an understatement. For many reasons, 20 years also, more than 20 years with Creative Commons and still clearly we have – there are some challenges. Looking forward to any other episode where Heather comes and explains her views on this space. Really, really important.

Now, going to the question. No. Absolutely. Just knowing that design is already code. That design, anything you are creating on Penpot, is SVG instantly. Actually, if you inspect the code on the browser, you will see that SVG is there. We're actually using browser technology, open standards.

No. What we are doing is that – and this is very important. And the community has supported us on this very much, very strongly, is that designers are borrowing vocabulary from developers, so that we don't have this lost in translation between what you think and design and what actually gets into the code. That is one of our little nasty tricks. To make sure that developers onboard the design tool. Because they will find themselves using their own vocabulary, which is in the end what gets implemented into code, right?

We are sort of – this is a win-win. Not 50-50. It's not like that. But everyone concedes a bit. Like, gives away a bit. In the case of designers would say, "Okay, my own vocabulary is not that important. Okay? If in the end we get rid of this lost in translation, I will use your W3C CSS vocabulary." And then developers will say, "Okay, then I'll be an active part of that."

But the only thing we missed in this – I mean, besides Heather's view on content and design licensing issues is that there's one key – one thing that's missing to finally give design the power they deserve, right? And not only that design is code. Code is design. And so, we have this merge in teams. But the next level is that scope is design and design is scope. How much can we blur the lines between the Agile process and the design process so that what a good user story looks like is not mandated by 2002 document by some developer, right?

When it's ready, or definition of done, or what you really need to everyone understand, "Okay, this is what we're going to do." How much design can be there? Instead of just like what a developer would like to see just because they're used to? Again, I'm saying just for the audience, I'm a developer. I really like structure long text words that leave no room for imagination, right? And it's super deterministic.

But with the right tools, we could have an interactive prototype actually being the definition of a user story. That is the scope. And I think, for ourselves, that's a huge opportunity to merge Taiga and Penpot. Those are the tools that could bridge the gap more profoundly, right? So, have scope, design and code as a closed loop. That would be great for us. We didn't touch upon that. But if anything, that is like something I'm very much into it right now.

[00:57:52] JMC: Scott, as the only designer in this conversation and, literally, as the final – as the closing point, are you as optimistic as Pablo is with this tradeoffs that you are so willing to give away and stuff like that?

[00:58:08] SJ: Oh, I'm absolutely optimistic. I mean, Apple was spearheading design for decades. And yet, the PC mentality maintained itself for decades. Just because the world is going to be slow to follow doesn't mean we're there to lead and have these insights. And it will win over time. Ultimately, I'm very enthusiastic.

But it is always going to be that case when, like, there's an agile user story, there's executing to that story and there's asking, "Why is that story even there?" And leveling up those questions. And that's where Pablo's comment about scope, I thought, was so powerful.

I do believe that we're always going to be wanting to level up and asking these harder questions. And the more companies, like Kaleidos that is doing that, the more people will follow their example because it works for them. We just need to have more success stories.

[00:59:07] JMC: Well, we're at the top of the hour. It was splendid. Actually, in a very collaborative way in the spirit of this same conversation, I will let the Software Engineering Daily decide if they find the two open doors that we left open just a minute ago, if they find those interesting and they want to explore the licensing conundrum that Heather just touched upon briefly at the end. And/or the future of design, plus code, plus scope. And the long shade of scope creep and other problems.

I'll share with the community and see if they want to explore that and then reach out to you. But other than that, I'm extremely thankful that you've joined and spent an hour with all of us talking about this fascinating problem and solution. I'll add the links to all the relevant resources that we've mentioned throughout the conversation in the show notes.

And, please, Heather, Scott and Pablo, let us know where can anyone reach out to you if you've got social media and you're open to discuss any interesting ideas with anyone that listens to the episode. Can they reach out to you, Heather, anywhere? Do you have any Twitter handle?

[01:00:33] HM: Oh, yes. Easy to find me at heather@oss.capital. And I also have a blog at heathermeeker.com. And it's really easy to find me.

[01:00:49] SJ: And you can find me just at jenson.org.

[01:00:55] PRM: You can find me on Twitter, Mastodon and other social networks with a handle of @diacritica. And also, if you are into archery, you can go to our java.net and find wonderful resources on traditional archery. It's in Spanish, I'm afraid, most of the content. But Google translate does a decent job. Yeah, that's another thing.

[01:01:16] JMC: Funny enough, my brother, hopefully, he won't listen to this. He probably maybe won't even listen to this ever. But **[inaudible 01:01:25]** so that you know. Thanks to Pablo. Your Christmas gift is going to be related to archery. You'll know in a couple of months. Thanks, Pablo, for those resources. They're fantastic.

Take care, all. Bye-bye.

[01:01:44] SJ: Thank you. Bye-bye.

[01:01:45] HM: Thank you. Bye-bye.

[01:01:47] PRM: Thank you. Bye.