Beyond the Commit, Episode 2: Gilberto Taccari - Transcript

In the second episode, we turn our attention to the evolving role of the Chief Technology Officer, the challenges of tech recruitment, and the dynamic reality of the FinTech industry. Together, these themes highlight how technology leadership influences both organizational growth and the broader financial ecosystem.

Our host, **Paweł Dolega**, talks with **Gilberto Taccari**, CTO at Tot. With a PhD in Computer Engineering and over ten years of experience, Gilberto combines technical depth with business strategy.

His career spans consultancy for major Italian banks, leadership at FinTech startups, and building high-performing teams in competitive markets. Guided by a "build it right" philosophy, Gilberto shares how sustainable growth, smart hiring, and sound technology decisions can influence the future of finance.

Beyond the Commit is brought to you by VirtusLab & SoftwareMill. Our podcast spotlights CTOs and senior engineers, sharing candid stories that resonate with technology and business leaders alike. Expanding on our popular technology blog (45 k monthly readers), the series adopts a more personal, conversation-driven format.

Want to listen to the podcast on Spotify, Apple, or Google Podcasts? Check out the Beyond the Commit website for details. Here you'll find the transcript of the conversation:

Pawel: So, uh, we are up. Uh, hi, Gilberto. How are you today?

Gilberto: Hi. I'm fine, and you?

Pawel: Not too bad. Thank you. I'm already excited to have this conversation. So, um, today we have Gilberto Taccari who has a wealth of experience in the FinTech and generally software industry. You worked in a bunch of places, and in the meantime, you founded your own company. You've spent a lot of time in FinTech, and we wanted to talk about it today, you know, to tap into your experience in the FinTech industry, in the software engineering industry in general. Maybe talk about it a little bit, and maybe more generally also talk about the role of the CTO and its challenges. Does it sound good?

Gilberto: Yeah, sure.

Pawel: Yeah. All right. So, um, before we do that, uh, maybe what we could do is, if you could give us, you know, a brief overview of your career and maybe main inflection points, so our, you know, viewers and listeners can have a little bit of a background story about, you know, yourself.

Gilberto: Sure, sure. I can tell more about myself. Uh, well, uh, I am Gilberto, as you said, and, uh, I am currently the CTO of Tot. Tot is an Italian FinTech that offers a platform to support small and medium enterprises in Italy with their finance management.

We offer a bank account, we offer cards, and we offer external services that configure us not only as a challenger bank but as a platform. So, we don't have just banking products, but we also offer some additional services such as the possibility to request multiple cards for collaborators or the possibility to check electronic invoices, making the linking between your expenses and electronic invoice, et cetera.

Pawel: And you are working... Most of your career, you worked from Italy. Is that correct?

Gilberto: Yeah. Yeah. Yeah.

Pawel: Yeah. So, is that like, you know, from your story, it seems like there is a quite vibrant community of, you know, startups and companies, especially in FinTech in Italy?

Gilberto: Well, uh, in Italy in general, I think, no. I would say no, but in Milan, yes. Milan is not only for FinTech startups but also is the, let's say, is the... I would like to say that it is the financial capital of Italy because it's the main city where you can find many banks or other companies related to finance. So, there is a great community related to FinTech in Milan. But yeah.

Pawel: Oh, got it. So, is it FinTech specifically or startups in general in Milan?

Gilberto: Uh, no, FinTech. I think that startups then are spread also in other cities. [laughs]

Pawel: Mm-hmm. Yeah. So, um, you mentioned... You know, thank you for this background story. And, um, you know, it's obvious from the story that you spent quite some time in the FinTech industry. And, um, from your perspective now as a CTO, or earlier as an engineer, what are the, you know, specific challenges in this particular industry, in FinTech, right, from the technological perspective?

Gilberto: Yeah. Uh, from the technical, technological perspective, I think that FinTech is, uh, well, since more or less 10 years or so, my career in FinTech, FinTech is a buzzword. I think that technology has been between the financial processes and companies since the beginning. We still have some companies that use COBOL as a programming language, that is pretty old.

But during my career, given that I had the opportunity to work on, let's say, incumbents, so, let's say, established banks, and also FinTech, there are some constants that I've seen. The first one is to test the application a lot.

Pawel: Mm-hmm.

Gilberto: Because in this sector, we are managing money, so it's pretty important to have a robust code and prevent errors because at the end there are some customers that are pretty, let's say, sensitive about something that doesn't work because it's about money. And so in all the companies I've seen a lot of testing, even manual or automatic.

In the, let's say, the legacy companies, maybe they have dedicated teams for manual testing the stuff, but when I moved to the startup sector and FinTech, I've seen many more automatic tests because the small companies, such as startups, cannot afford a whole team of manual testers. So this is one thing that I've seen.

And the other challenge that I see in this sector is to keep in mind that there are regulations.

Pawel: Okay

Gilberto: We are in Europe, so there are regulations about the personal data such as the GDPR. But in the financial industry, there are more regulations. And they can be, for certain people, challenges or limitations to innovation. I don't consider them limitations because you can innovate even with certain regulations, and you must innovate also because of regulation.

And so, I think that this is another, let's say, challenge because you have to keep informed and understand well the regulation and how your code can fit with it, how your architecture can fit with it, how your product requirements can work for both your customers and the regulators. And this is another challenge.

And this is the main challenge, I think, and the main advantage for startups because incumbents, players in the financial industry in general, had slower approval processes, very old teams responsible for regulation and compliance, et cetera. And startups are more agile. So, understanding the limitation and finding the solution without a whole hierarchy of approvals. And probably, one of the challenges that is the target regulations are also a key advantage or an opportunity for startups.

Pawel: So, that's an interesting thing because you mentioned in your answer this, um, you know, those regulations, also security concerns because you are dealing with money, because you are dealing with sensitive personal information. And I was wondering how do you, you know, deal with that, especially, for instance, as a small company, when you work at startups, and as in a startup, when you founded your own company.

Typically, and I'm asking this question specifically because typically those small companies can compete with larger existing companies because of the, as you mentioned, speed of innovation or speed of delivery, speed of testing the ideas with customers. And I was wondering how do you, you know, connect this and how do you deal with this in the environment when there is such a need, as you said, for testing, for regulation and, generally, for security?

Gilberto: Uh, well, [clears throat] I think that dealing with that is more connected to the way that you take care of this aspect. So, I think that in my personal experience, I think that when it comes to the regulation, there are people in the teams that are responsible for the product. And so they are the main, let's say, responsible for collecting all the requirements and are also able to summarize the requirements and tell what they are to the developers.

I think that the good approach to face regulation is to understand the laws and then be able to communicate them to the whole team. My best experiences in my career are the ones where I had the opportunity to work with teams where there was very good communication and also the developers were able to communicate with the product people. So, don't talk in

terms of programming codes, but in terms of domain experts, because I find that developers are also experts about the domains they work on, even if they are pretty good technicians.

And so, one approach to face the challenge of regulation is to share the knowledge and understand what the regulation mandates. And then I think that when you are able to understand the problem, the law, you are able to build the code and your architecture that works and fits with that regulation.

In terms of testing and what I'm obsessive about, automatic testing, 'cause I don't like to make something manual. And so I tried in all my steps, in all my career to push for automatic testing.

Pawel: Mm-hmm. Mm-hmm.

Gilberto: And it can also prevent some, let's say, regressions during the implementation and evolution of your product. There is a funny stuff about that because when I started building my own automatic tests in my first experience, and there wasn't a culture of testing, it was a nightmare for other developers because to evolve the code, they had to disable the test. It was a nightmare for me too because you see that something is deleted. So, remove the test to speed to develop fast. However, so for testing, I push a lot for automatic testing. It's not always possible or easy, and so sometimes we come back to manual testing.

And for security, this is also a cultural aspect. In this case, I think that the role of the CTO or other technical leaders is crucial because in a startup, unless the startup works in the security industry-

Pawel: Mm-hmm

Gilberto: ... you see security as a... I'm not saying a limitation, but I say, "Okay, we want to implement features for our customers. We want to push for new features."

Pawel: So, from that, you know, maybe from that angle, um, you know, given those characteristics of the industry, are there, from your perspective, any methods or tools? And I'm thinking about things like, for instance, functional programming or strong typing, for instance. Like, you know, in the Scala programming language or maybe even like in Haskell or something like that. I'm talking about the languages that put a lot of, you know, effort into making sure that you can build this compilation time safely, like that your business logic is encoded in types, so you can avoid certain kinds of errors.

Do you think... So that was just an example. Do you think there are some specific tools in the FinTech industry that from the perspective of what you said, you know, security, quality, and things like that, are better than others?

Gilberto: Uh, well, I think that's the way... There aren't some specific tools, but I give more weight to the engineering approach. However, there are some approaches that are better than others. And for example, in all my career, I used typehead programming languages.

And because I started with Java, and because it was the programming language for services in the banking sector.

Pawel: Right.

Gilberto: And it's still... that language is still the language for the banks. However, what I think is that there are some aspects that allow implementing more robust and secure code. For example, using a programming language that supports types-

Pawel: Mm-hmm

Gilberto: ... a strong typehead typing system. And also, here in Tot, we are using functional programming, for example.

And I have seen that with functional programming and TypeScript as a language we use, I see that there is a kind of better mindset in my opinion, compared to people that don't use such kinds of technologies, to think about the business logic.

For example, with a properly implemented application with functional programming, you have to consider all the paths because the data types, such as Either or other elements, force you to think about the whole business logic and not say-

Pawel: Right

Gilberto: ... "I can put on a null" or other weird stuff that exist in other languages. And I think that it helps. It also helps because of the composability of different functions. So, any developer or any company can build its own tool set of functions that can compose and reuse. And I think that this part can help to build a mindset that moves the developers to write good code or good product code.

This is what I think about functional programming and technology. So, I think in the FinTech industry, using typehead languages and using languages that are, let's say, stable and have a lot of libraries can help to face the challenges of the sector.

Pawel: Mm-hmm. Yeah. Uh, you mentioned that... Is this correct that your main stack is based on TypeScript, both front-end and back-end?

Gilberto: Yeah, it is.

Pawel: It's an unusual choice, right? For your functional programming in TypeScript, could you, you know, maybe shed some light on how you, you know, picked up this particular choice as a source of the language to be the backbone of the services that you build?

Gilberto: Uh, well, it's not common, but I think that's safe. I... I didn't find Tot, so I wasn't the first developer.

Pawel: Right

Gilberto: So, I found this... I found TypeScript.

Pawel: Mm-hmm

Gilberto: Functional programming, and I had experience with both of them in the past. And so, what I think about this technology is that it works in this sector. And the way that I

approached it, it's more about, "Let me understand how it works and how the people started working with it."

Because even though it isn't common, apart from the core developers that built the core part of Tot, of the-

Pawel: Mm-hmm

Gilberto: ... application, other developers who joined, they know functional programming or had a different approach with TypeScript. Some of them even use TypeScript as a first programming language. So, they know it.

Pawel: Mm-hmm.

Gilberto: They knew it, but it wasn't the first language. And so, what I found is that this approach worked because people can easily start thinking in function terms. So-

Pawel: Mm-hmm

Gilberto: ... thinking about the function. And in this way, I also find another interesting aspect is that, given that the approach is not common and people came from other contexts, languages and frameworks, I also found that any developers that joined Tot brought, from other experiences, their own knowledge, approaches, and way of thinking.

And moving them to, let's say, a new language at the end-

Pawel: Mm-hmm

Gilberto: ... even if they knew TypeScript, but maybe not fp-ts, that is the functional programming library that we use. I found functional programming, the... our code base as a place where they can express their own knowledge in a different language. So, without any previous constraint or limitation, they put the knowledge within our code base using this new language and new approach.

And I found it useful because I also experienced it, because I say, "Okay, in the past I used other languages." How can I put what I found useful in other languages in this code base without changing the language? And I also find this useful because the developer, the engineer, doesn't think in terms of the language, but in terms of the design of that software.

Pawel: Yeah

Gilberto: And it makes the... also the growth of the person more stable, constant and-

Pawel: Mm-hmm

Gilberto: ... steady. So, this is my way of thinking about functional programming. It's not common. It was a choice for Tot at the beginning, but at the end, I'm finding it working and-

Pawel: Mm-hmm

Gilberto: ... also an opportunity for people that join our team and people that still work in our team.

Pawel: Right. Yeah, I mean, I guess, you know, a lot of people know the JavaScript ecosystem and TypeScript specifically. I was just thinking that the mixture of, you know, TypeScript in a functional programming language is a little bit less common, although, you know, it's probably fair to say that in recent years, it's gaining traction, definitely. As you mentioned, there are libraries popping out. There are things happening and there are more companies focusing on this area.

So yeah, I mean, it's especially interesting as you mentioned that the FinTech industry is pretty much based on the JVM to a large extent, in Java. And you picked sort of a different, you know, approach. So, uh, yeah. Good that it's working for you.

Now, I'm thinking about that. You mentioned a couple of different things. One was those, you know, we just talked about this mixture of TypeScript and functional approach. But then you mentioned also this focus on automation and testing and basically, you know, quality, security. And also you mentioned that you require... You'd like to have your engineers or you have the engineers who have a lot of focus on the domain, so they also, apart from the technical skills, need to have focus on the domain they are working on.

How do you find it difficult to recruit the right people with the mix of all these skills? It seems like a lot, right? Like, there's a lot of different requirements and it's always in the software industry, it's difficult to find great engineers. How do you manage that? You know, how do you find the engineers to work with Tot?

Gilberto: Well, uh, finding the right engineers is always hard, I think. It doesn't depend on the sector or-

Pawel: Hard and expensive

Gilberto: ... or... Yes. I think that it's going to be... Over the years, it seems that it's always harder. However, finding good engineers in Tot is pretty difficult because of the technologies that we use. So, if we have to find candidates that know both TypeScript and functional programming, we are going to, let's say, not hire at all. [laughs] Um-

Pawel: And also that knows FinTech, right?

Gilberto: Yeah. I don't know. I'm bit joking, but it's really hard to find people with those characteristics. With the skills on those languages and practices. However, um, I think that the way to find the good fit with the company is to have a path for reaching candidates with posts online, with opening positions. But to find a good candidate, I think that the better approach is to provide an environment where they can grow. So, you can offer a good environment to work and you can offer good technologies, apart from salaries.

And to find them, I'm pretty focused on the way they think, because in our hiring process, there is a step that involves the front-end or back-end lead. And then, when I talk with the candidates, because when they interview them, it's just to understand the way they think.

To find the good ones, I think that what I look for is the approach to think about the problem and that it's not only about problem-solving, that is another often used word, but it's more important to be able to understand a problem, to think about possible solutions to the problem, to also share the ideas with another person. So, I look for that, and more than the technical skills that are important.

For me, it's important to have a good understanding of new topics and be curious about stuff. So, this is the way that I find the engineers.

Pawel: So, it's more about the engineering mindset that you're looking for rather than the experience in the specific technologies that you are using.

Gilberto: Yeah. Yeah. Because technologies can change, and what is more important is that you are able to face issues with... or build something, because I don't want to put everything as a challenge or face challenges, but it's more about building something new, building stuff, and it doesn't care about technology. What it cares about is the approach that you have.

So, yes. I think that the right engineer can study and learn a new technology, but a good technician isn't always able to approach a problem or deeply understand one domain.

Pawel: Right.

Gilberto: Probably there is also a kind of bias in this evaluation because I say that my career is 100% in the financial industry and FinTech because I really appreciated and I really love the sector. And it's a fun fact that I started to make some investments because at the beginning of my career, I worked on applications that moved stocks, so financial instruments for a bank.

And this is also a kind of bias, but I think that when you, when you understand a domain and you can talk in terms of domain concepts, without talking about domain-driven design, but when you're able to understand and explain the sector where you work and the problems with the right terms, so you also find what I was telling you before about communicating with the rest of the team, with the product managers or product owners and share the knowledge with the same language. So, it's important. Then the technology... A good engineer can learn a new language, a new framework, a new paradigm. So, it's less important to me to know a specific language, for example.

Pawel: Right. So, it's like a combination. The one thing is, obviously, the technology and knowledge in itself, but then what you mentioned is this engineering broader mindset and the skills to deliver something. So, there is, I guess, something like pragmatism, engineering pragmatism that is important. And the last thing is this communication, you know? Which is, by the way, often overlooked.

So, the one thing that I think is part of it is that, unfortunately, technical skills, at least from my perspective, are easier to test. When you start talking with someone, you know, during interview, during recruitment process, technical skills, like you're given some task, you ask some technical questions, there's usually some answer that is more right than the others. So, you sort of, you know, can assess it.

How do you assess those other aspects, though? It seems like it was always, for me, like a thing, and we are still, not saying struggling, but we are still evolving as a company to do a good recruitment process because we also found out that, you know, those other aspects, you know, how you deliver, how, you know, practical you are as an engineer, then the other soft skills, how do you communicate, how do you work with the team, these are important aspects, but they are more difficult to, to, you know, assess, especially if you have, you know, limited time, like one or two hours. How do you do that?

Gilberto: Oh, well, yeah.

Pawel: Any tips?

Gilberto: What I do, it's to... Let's start... During my interview, it's just I introduce myself and then I say, "Okay, we are going to talk an hour about what you do, what I do," and something like that. It's more, let's say, a friendly approach.

And to test that, I often ask the candidate, given that the candidate has already been interviewed by the tech lead. So, I'm not even the one that is more expert on some technological aspects connected to the language. But, when I interview them, what I ask them is to tell me what they do, on what they are working at the moment. Try to communicate what you are doing in terms that I can understand.

Pawel: Mm-hmm.

Gilberto: And this is my approach because, especially with, let's say, the mid-level or junior level, they often talk to you about the technologies that they use, the technological stack, et cetera. But with more senior people, so the people that we are talking about, if you want to hire a good engineer with the good fit, I ask them to tell me more about their job and the problems that they faced.

I often ask the sector where they work. And apart from candidates that came from the same industry I worked, other ones are from other industries that I don't know. So, I'm pretty curious to know about the new stuff. And I evaluate their soft skills trying to understand the sectors they work in.

I ask questions. "Okay, you did this, but why? Can it happen that this fact or this item can change the logic?" And I see how they reply to me. This is the approach. I impersonate the role of the technician that wants to understand a new domain, and they are the ones that have to explain their domains. And I really find this pretty useful because it's... It's not about, "I'm asking you questions to evaluate you if you are good or not on specific technologies or a specific approach in solving some issues." I also ask that, but it's more about, "Okay, tell me what you do and try to explain to me something that I should not know." And I can also ask stupid questions because I don't know the domain and ask them to explain that to me.

And the more they are able to teach me something new-

Pawel: Yeah, yeah, yeah

Gilberto: ... the more I appreciate that. I want to hire people that know more stuff than me. [laughs]

Pawel: Right. And that's, by the way, that's one way you can make this, you know, interview process also a learning experience for yourself, right?

Gilberto: Yeah.

Pawel: You learn something new every time. Yeah. And that's an interesting approach. Like, not only you check the communication, but you also check because you said that the other factor that you, you know, care about is the engineers knowing the domain. And obviously there are some engineers who don't care that much about the domain they work in. They just want to have technical, you know, challenges to deal with. And you are checking that because if they are able to describe, you know, the industry or, you know, the problem space they were working in, they have to know something about the domain. So, it also shows you, I guess-

Gilberto: Yeah. Yeah

Pawel: ... that they are interested in the domain they are working in.

Gilberto: But however, I am not saying that I will reject all the people that are good technicians. I also want to learn something new from those people. So, they use a framework, a library, a service that I don't know. "Okay, tell me how it works. I don't know. You have to convince me to use it." And you say, "Or so." "Teach me something new." And I, I, I see that. So, a good technician, I expect that knows very well some technologies, and she or he is also able to communicate, to explain, to tell me about the technology, the framework or the services.

So, I use this approach because I found it useful both for me and the candidate because-

Pawel: Mm-hmm

Gilberto: ... a good senior candidate should also be able to communicate effectively some concepts.

Pawel: Got it. So, it doesn't have to necessarily be domain, it can be-

Gilberto: Yeah

Pawel: ... something else. Yeah. So, you mentioned that it's becoming... Over the years, it never was easy to, you know, to hire good experienced engineers, and it's becoming more and more difficult.

You see, given that we had the, I think we had some sort of the, um, hype or at least, you know, the peak of the market. That's what people say in terms of, you know, in terms of the demand for engineering between I guess 2020, maybe 2021 to 2022. You see that it's... Is it more difficult or less difficult than it was back then to hire engineers? Or is it about the same? Do you see any trends?

Gilberto: Well, I'm seeing now that it's pretty difficult, and I think that partially is related to our technologies, even if they are not so crucial, but it's pretty difficult to find also types of developers that are good.

And I don't know if we... I know that the peak was in the past because of COVID and other stuff, we had all the companies working remotely and everything was digital. And what I see now is that it's difficult and there are valid candidates. In the past, also in those years, I was more lucky with hiring, but it was in Fair AI that we used Java. So, I think that there are many more Java developers than TypeScript.

But I think it's pretty much the same. I probably I don't see a lot of changes apart from the fact that during the... during the pandemic period, many developers were rising their roles, and probably now you look for experienced engineers that don't want to be, let's say, a senior engineer because they are preferring to have another role because they grew a lot in other circumstances.

And then what makes me a bit sad is that there are some engineers that say they are senior, but in the end, they don't have this great seniority. And I don't know if it's caused by the big hiring flow in the past years. I don't know. I really don't know, but I find it-

Pawel: ... maybe, maybe that was because of this, you know, the peak hiring that happened, and there was a lot of this inflation-

Gilberto: Yeah, probably

Pawel: ... of, uh, you know, of, uh, levels happening in various industries. And I think some of those people are, because of that, are a little bit stuck right now in their roles because if they want to move somewhere else, it suddenly feels like they are not, uh, you know, perceived as being senior. And yet, in their companies, they are senior, so they are stuck. And, well, it's the problem that it's going to solve itself. I think if, you know, people staying in this, you know, company sufficient time, eventually, again, if they will be learning, they'll gain enough experience to justify this senior position somewhere else.

But there's also the other aspect, which is, I think, happening, that a lot of... That's what we are seeing in terms of the recruitment. A lot of people had some, you know, salaries increased that happened over the last years. And for some of these people, it's difficult to find another, you know, another role in a different company where they would earn more money. And let's be honest, most people, when they are changing jobs, are looking for at least a 20% rise in their salary. You need to be very, very unhappy in your role to accept a lower salary or even similar sometimes.

Gilberto: Yeah. Yeah, I also saw that.

Pawel: Mm-hmm. So, you mentioned, you know, this peak of difference, a peak of combination of different, um, you know, skills, requirements, the way how you recruit. What I'm also seeing is that... I don't know how it's in Italy, but I see it in Poland and generally throughout Europe, there are more and more companies like Big Tech or Silicon Valley companies. And I'm talking about big names, not necessarily, you know, the Meta or Google, but even those companies like, for instance, Dropbox and, I don't know, Palantir, Big Tech companies.

I think that in the past, they didn't have that many offices throughout the world, and they were mostly hiring in the US. And what I'm seeing more and more is they open more, you

know, more centers, development centers in Poland, in Europe, in various countries. And that is something that definitely changed in the last few years, maybe five years.

And I was wondering, as a company, as, you know, where you work at Tot, how do you compete with, you know, those big companies? It's difficult, as far as I understand, to compete money-wise or the compensation-wise, because those Big Tech companies have a, you know, very large budget in terms of the, you know, full compensation package which goes through, you know, salary, you know, stock options or whatever they have.

How do you compete with that? What are the things that you focus on in terms of what you offer employees? And I'm not talking necessarily about money. I'm talking about the overall picture, like how do you, how do you make, you know, people wanting to, to come to Tot and work at Tot instead of Big Tech?

Gilberto: Well, I think that my interviews I had done in the past months... people that approach Tot do that because they look for a startup, for a company that is not static and-

Pawel: Mm-hmm

Gilberto: ... can evolve easily and quickly. And so we can compete on that because I, I know ex-colleagues that started working for Big Tech companies-

Pawel: Mm-hmm

Gilberto: ... with a very huge salaries-

Pawel: Mm-hmm

Gilberto: ... but then they left. I don't know if it's because they became rich or for other reasons, but they say that they didn't find there the same mindset of startups.

I don't know if it's true or not. I didn't verify that, but I sense that we attract people because we are a startup. And often people get bored in big corporations with long processes and no innovation or slow innovation.

So, you are able to propose new stuff and new technology, but you have to wait a chain of approvals before you put something-

Pawel: Right. Right

Gilberto: ... on the code.

And this is one thing that we offer. We offer a dynamic context where at the moment, any developer counts a lot because we are not so many. And so we offer that. We offer the velocity of a startup, the possibility to try something, make mistakes, but we can revert and go back with another, with the previous approach if it was better.

Pawel: Mm-hmm.

Gilberto: And then, what we offer is a very young environment, because all the developers are young and getting older. [laughs] And I want to say that we offer an inclusive

environment. A good percentage of developers are female. And I really love that because I've seen other companies where the tech department is completely full of males.

Pawel: Right. Tech, tech bros.

Gilberto: Yes. In general, the tech bros. The STEM sector is not very... I don't know about Poland, but in Italy, it's pretty dominated by males. And in some way, it also limits creativity or perspectives.

And here in Tot, I'm pretty proud of the fact that we also have some females in the team, and they are a good percentage in the tech team, and the product team in general. And there is a kind of right combination that leads us to a good way of thinking about the product.

So, a very inclusive environment where people are free to talk about, propose stuff. Also, the junior that joins Tot can also say something, propose new stuff, talk directly with the CTO. We are a startup, so-

Pawel: [laughs]

Gilberto: ... There is no big hierarchy. And I appreciate that. I communicate that to the candidates during the interviews. So, even the people that at the end decide for another job, another offer, I found that they were pretty happy to talk with me. And I can feel that because even if a candidate is talking with me because she or he wants a job, I can feel if she or he enjoyed the talk or not.

So, this is one thing that I communicate a lot during the interviews, this inclusiveness and the possibility of proposing new technologies, and that we are pretty fast to test something. And we are... I don't want to say I'm obsessed with customers because I don't believe that we should be obsessed in any way, but we take care of the customers. And I communicate that. I think that the candidates appreciate that, and these are the things that we can offer. And I think that this is the... those are the reason why somebody chooses Tot, compared to other companies.

Pawel: Mm-hmm. Since you're so successful in that, do you have any tips for, you know, these people to... how you can make sure or improve the odds of having a more diverse, you know, tech department?

Gilberto: Well, I don't know if there is a recipe, apart from the fact that consider the applications at the same level, and avoid biases. Yes. And consider the bias only if you see that in comparison to two CVs, a less experienced female in the CV could be comparable with a more experienced developer that is male, just because of their opportunity. I think that the main recipe is to consider them at the same level, so don't have a bias, and that's the bias that stops them at the beginning.

Pawel: Right.

Gilberto: And, uh-

Pawel: So, hold on for a moment, because I'm not sure I understood it. What you mentioned is that you have two CVs, you know, one from a woman and the other one from a man, and

even if they are not comparable... obviously, every CV is different, but what you're saying is that, often a woman's CV, even if it seems that the person is less experienced, you treat it as the same level as someone, you know, more experienced because of-

Gilberto: Well, it's slightly different. What I'm saying is that a woman's CVs maybe, or a woman's career, maybe it's more difficult because of the bias.

Pawel: Yeah.

Gilberto: Because of less opportunities. And often you see that... I see the potential to grow if they join a company that allows them to grow.

Pawel: Right. Right.

Gilberto: And I had this... I have had this feeling since probably... Well, I'm not saying the beginning of my career, but for several years, I had the opportunity to see women at work that started as a, let's say, junior developer, but grew a lot and quickly because they found the right environment. And given that, I felt that, and I've seen that during my career, I am pretty inclusive of that.

Pawel: Mm-hmm

Gilberto: If any developer, if you give them the opportunity to express themself and propose ideas and have the same opportunities to grow, I think that they can give the best they can do, they can give to the company. So this is my feeling, and this is the reason why I...

Pawel: Got it. Got it. I was thinking that one of the reasons is because I think I'd noticed it. I'm not entirely sure if it's an anecdotal evidence, so you didn't measure it in any way, but what I'm thinking is that in my career, whenever, you know, I was recruiting people, usually when these were women, I had this feeling that, you know, there was a question about something and, you know, women didn't... Like, when we asked them how good they are at something, let's say, hypothetical, you know, "How good are you at, you know, TypeScript?" So, compared to women, what I had in my experience is that with the same level of experience, men might say, like, seven and the woman would say four.

Gilberto: Yeah, this

Pawel: Same level... They have... they have exactly the same... like, just hypothetically. But I had this feeling that it was the case, and it's very easy in that case to sort of, you know, you go through a CV and you treat it a little bit worse because it's not as, you know, in a way bold, this CV. They don't, you know, exaggerate that much. And then you have the other CV. Of course, we are talking, you know, I'm talking statistically, everyone is different. So, it's not a, you know, a golden rule. But I had the feeling that as a general thing, it is the case and you need to...

And the funny thing is that in my career, I worked with great women engineers, and I think that one of the best that I knew was a woman. And it was always the case that it was like they were a little bit shy and you had to, you know... Over the course of their career, they had to learn to be a little bit more bold and courageous because they were not, you know, showing

all the... I think that they could be more expressive about their achievements. So, that was the case in my... Have you seen the same thing? Do you have similar experiences?

Gilberto: Yes, I had similar cases. And at the moment, I'm in a role that I can, let's say, encourage people to express themself. In the past, I had a very, let's say, sad experience where I saw that there was a junior developer... she was a woman, and she had a master degree in mathematics or something like that. And she's very shy, and I also saw that approach to being shy, and pretty... She hadn't a strong opinion on, for example, on the code when we had some code review and other stuff. And I think that because she was in, let's say, a new environment and not so inclusive. I don't know, but I pushed them. I was a senior back-end developer at that time, but pushed them to express herself and to... Don't be shy about that.

And it worked. I think that I'm following her on LinkedIn now, and she's writing blog posts and other stuff about AI and new topics. So, I think that she learned the right approach to be successful. But I agree with you that in some cases, there is this kind of bias in considering themself. And if you are a woman, you say, "Okay, I, I want to be sincere with that. My knowledge is four," but at the end, it's seven. [laughs]

Pawel: Yeah. Yeah. Yeah, that's an, that's an interesting thing, and, you know, the fact that you are taking this into account, I think it's a, it's a good tip. I think it's a good tip to be mindful of that and be aware during the recruitment process.

So, um, you also mentioned these blog posts with AI, so maybe we could explore this a little bit. So, obviously, AI, AI tooling in development in general, but not only in development, is a big hype right now, so GenAI is a thing. So, every now and then, like every week, there's almost a new model or a new product or, you know, something in that space. How does it affect fintech? Or maybe we could start, you know, from fintech, and then maybe you could talk about Tot itself. So, I'm wondering about two aspects. First is, you know, usage of the AI in the fintech industry, and the second is development tooling. Do you see this development tooling, you know, impacting the way you work? I mean, I'm talking about the code generation, you know, automatic reviews or things like that. Anything in the area of AI-assisted development.

Gilberto: Well, um, in the sector part, I think that with the GenAI models, now AI is everything, and you have to build something with AI inside just to be cool. And apart from that, what I'm seeing is that even in Tot I worked with AI, not with LLM models, but I worked with other models made within the company.

What is in the... in the past years, what changed with the new models is the interface that you use to communicate with AI. So, you are not familiar, let's say, my... Some people consider themselves familiar with AI just because they can interface with it and see the results and improve the results because they can communicate by texting to a model, to a statistical model.

I think that since AI is going to be in any product, in my opinion, also in FinTech, AI is used. It was used for certain other, let's say, less popular models-

Pawel: Mm-hmm

Gilberto: ... for fraud detection or other stuff. But now with the new models and GenAI, AI is used in many more scenarios. It, for sure, it's going to bring some, um, innovation or some more easy automation, because at the end, what I'm seeing now, the most successful processes or scenarios where AI is used is for automating the boring stuff.

Pawel: Mm-hmm

Gilberto: And it's pretty good if the problem is not so complex. And so, for sure, it's going to be used more and more for many aspects. Also, the fraud detection systems are now starting to use LLMs because they are... since they are good enough to prevent-

Pawel: Mm-hmm

Gilberto: ... to prevent the frauds. And it's going to... I'm not saying replace, but also use it together with other existing models that worked with other-

Pawel: Mm-hmm

Gilberto: ... logic.

Moving back to Tot and to the creation of code, I think that these tools are useful. I also use them for generating some proof of concept or parts of proof of concept because-

Pawel: Mm-hmm

Gilberto: ... if you know the language, you can see the difference between your approach to code, to the ones that the model proposed.

Pawel: Mm-hmm, mm-hmm.

Gilberto: And also here in Tot, we use some tools. We don't have a, let's say, a company-suggested tool, but we give to the developers the possibility to use the tools that they prefer. We are in a phase where developers are experimenting with tools.

Pawel: Mm-hmm

Gilberto: And what I can see as a leader in my role is that these tools are simplifying some steps and make them less boring just because there is a model that... But the human factor is still relevant. So, we principally use those tools for generating code or editing some pre-request summaries-

Pawel: Mm-hmm

Gilberto: ... that analyze the code, then make the summary to, to what are the changes that are pretty long. "I changed this file-

Pawel: Mm-hmm

Gilberto: ... and this file, and this file. Okay, I see the changes." [laughs] Now, they pre-request like, "I know which files. Do you think-"

Pawel: Right

Gilberto: ... that those were useful. Yeah, in certain parts, yes. But, um, I think that the human factor and the knowledge of the domain is still relevant.

Pawel: Mm-hmm.

Gilberto: And so, for this reason, I think that also for the upcoming years, we are going to use both AI and human factors. And I don't see that it is going to replace 100% of the developers. And I also don't think that in the short term, the AI is going to write the whole code for us. Because we have also to consider that some parts of the code are legacy.

Pawel: Right. Right.

Gilberto: And I think that's... if the most senior developer that joined us in the company found something hard to understand, according to his experience in the code, I think that also AI is challenged about that code. So, I think that we are still going to need people to play the role of supervisors of the AI results. But however, the tools are evolving week by week. So-

Pawel: Mm-hmm

Gilberto: ... of course, they are going to improve. But, as I said at the beginning, we are dealing with money. So, we prefer to not trust a statistical model to handle the money [laughs] of all the code.

Pawel: Right. Right.

Gilberto: And, uh...

Pawel: So, if you were to make a bet from what you were saying, three years from now, you'd say that developers would be still largely writing the code, right? Most of the code would be written by developers in three years?

Gilberto: Well, I see... All write the code or supervise the code. Yes. Probably you will be able to allow an AI agent to write the code for you, but the developer should take a look. And also, we have to find who is the final, who is the subject responsible for the application errors or-

Pawel: Mm-hmm

Gilberto: ... other stuff. And, of course, for certain boring parts, we wouldn't need developers probably anymore. So, the boring, monkey job, as we said, will be done by AI. But for other stuff, I think that the human factor will still be relevant.

And to connect to what I was saying before, I think that knowing the domain and having an analytical approach to the domain more than the programming language-

Pawel: Mm-hmm

Gilberto: ... will be a key factor for the new engineering personnels. So, because you have... The role of the engineer will be the orchestrator of AI agents and domain requirements and problem analysis. Or simply, the new engineer must do the job of collecting and analyzing and then analyzing the requirements. Because at the end, if you want that AI to write the code for you, you have to provide the pretty clear and-

Pawel: Right

Gilberto: ... instructions. So, in the end, the developer won't be replaced. Probably, they will change their role a bit.

Pawel: Mm-hmm. It goes back to what you said about the recruitment also, that you mentioned that you're looking for good communication skills. And it looks to me, it sounds to me from what you're saying, that if it goes that direction, communication would be an even more important skill, because the clarity, how you communicate your intent would be vital, as the code is generated more and more by the AI models.

Gilberto: Sure. Sure. Yes. You have to be pretty precise and you have to... Let's say, I don't know if you experimented a bit with prompt engineering, but you have to clearly state everything, put guard rails to prevent certain response, adjust the parameters of the model, for example, the temperature to prevent some hallucinations or something like that.

Pawel: Yeah.

Gilberto: So, yes, the communication skills would be more and more important, and communicating with a model [laughs] at the end.

Pawel: Right. Right, right. I mean, I like... And generally, we circle around this communication skill. And I liked what... I think that was DHH, David Heinemeier Hansson from Basecamp. Now again, I think they are called 37 Signals. And he once said that, you know, writing code is still writing in a way as you would be writing prose, like a book or an article. When you write a book on their article, clarity, and the way you communicate is obviously important, you know, because you want people to be able to read your intentions, to understand what you are writing, not be bored along the way, so they, you know, get something from the article. And a lot of those things, you know, apply in the same way to writing code, right? Because you need to have this clarity of intent. You want your code to be readable so that people can go through it and understand. So, I think there is something into this, these, you know, communication and being able to generally communicate well, both written and in a spoken, verbal form.

So, you mentioned that you also do experiments with POCs, generating the code for POCs. And I think what people mentioned, and that is also my experience, is that whenever we talk about greenfield, I have to do some POC, check some things. You know, using various AI tools, be it Cursor or Copilot, whatever have these agentic modes, is absolutely amazing. Like for me, personally, so I don't spend most of my days writing code, and in the past, it was very difficult because whenever I had time and I wanted to do a POC, there was always a lot of this plumbing, a lot of this upfront work that you had to do.

And I ended up, I felt like, "Oh, this Friday, I'm going to have four hours." So, I built something that I always wanted to build, and I ended up in six hours realizing that I barely

scratched the surface because I was connecting with some, you know, authentication provider or deploying it or something like that, working all those things, which were not the core of what I wanted to do. So, for me, this ability to use AI models for greenfield POC projects is absolutely amazing.

But what I see in our company, and when I talk with other people, it's that when we talk about brownfields or legacy projects, it's a different story. It's significantly, you know, more difficult to apply and generate reasonable code, as you said about the legacy that your most senior developer has difficulties to understand.

So, I was wondering about your... You mentioned this POC, but I was wondering about your core platform, your core product. How much, you know, of the AI developers use, use there... Do you already see some sort of efficiency gains that you could claim, like say, 10% more effective or anything like that?

Gilberto: I, yeah, I don't have numbers because I don't have a... I didn't measure that, but when it comes to the brownfield, working with existing code, the AI can help in implementing, let's say, new features, for example, that are not so tightly coupled with existing ones. But at the end, I see that at the moment, the AI just speeds up the implementation because it recognizes some existing patterns in the code, or it's like a suggestion of some parts of the code. This can improve the efficiency.

But what I also found is that often it helps to duplicate the code. So, if there is the code yet and you duplicate that part of the code just because it's working in another part, it's going to make the code less clean as you were saying before. And so, for the brownfield, the human factor is important, and is crucial.

Pawel: Mm-hmm.

Gilberto: Also to validate the proposals of the AI, because AI may be, let's say, right, but the right implementation to do is to use the AI proposal to make a refactor of the existing code, not add the AI part just because it's AI.

Pawel: Yeah.

Gilberto: And go faster because AI helps a lot to write a lot of code. When you don't have anything and you have to build a proof of concept, you can use it because you want to build something from scratch quickly.

Pawel: Yeah.

Gilberto: And you must consider that as a proof of concept, because if you want to engineer it, you have to spend more and more time. But when you are in a greenfield, any single part that you have to existing code is code that you have to maintain together with the existing legacy code. And so you have to use AI carefully because it can bring a code explosion just because you have more and more lines and files, and you have to pay a lot of attention to that. And so, for the greenfield, I still prefer relevant supervision by humans.

Pawel: Right. Right. So, um, yeah, I find the same issues, like this code explosion that you mentioned. I wonder if that's a question of, you know, carefully manipulating or preparing

the prompt. So, here's what I found sometimes is that, you know, I have this code explosion because the change that I wanted to have, indeed the model generated it, you know, somewhere aside, and a lot of code was duplicated. But when I tweak the prompt a little bit or follow up, like, make sure that it's reusing the existing parts of code and, you know, that it's not a completely separate code because there is a lot of duplication, usually I was able to get some results, although perhaps not perfect, but it was better than the, than the first attempt.

Gilberto: I have to try.

Pawel: Yeah. So, I was wondering, do you, do you have some internal in-company guidelines or prompt library or prompt templates that you use?

Gilberto: No, we didn't get to create that.

Pawel: ... because it can get pretty... Like there are books about that. I think recently every major model provider is publishing a short ebook, like 60 pages about prompt engineering, right, for Gemini or for, you know, for entropic models or something like that, so it's a topic. It's a huge topic, and there is a... I think it's a skill that you, you know, that you may need to acquire in order to efficiently use it as a team, right?

Gilberto: Yeah, sure.

Pawel: Do you look for this, by the way, when we talk about skills? Do you look specifically for this skill during the recruitment process?

Gilberto: No, no, no.

Pawel: Like generally using AI tooling?

Gilberto: No, I don't looking for this specific skill. I... Personally, I'm studying the topic to propose something to the team. And I also study prompt engineering or the patterns and other stuff, in order to propose some guidelines and to share with the team. It's not a skill that I look for from candidates because I said, um, that Tot has built a lot of code in the previous three years, and I think that to be critical towards the code and understand it, it's pretty important, and I look for smart people that can do that.

Pawel: Got it.

Pawel: So, we talked about the recruitment, and we talked about the AI models and usage, you know, at Tot. I was wondering if maybe we could switch gears to a different topic, a related one. A lot of companies have, you know... Different companies have different approaches to working with, um, external service providers or consultancies. Some companies have their core of business, their, you know, the most important part, the secret sauce of their platform or service. And some companies want to focus on that, for instance, internally, and then subcontract the other parts to, you know, to some external vendors. Other companies want to have a mixture, like, you know, 80/20 or anything like that.

Do you have as a, you know, a CTO with experience in various companies, do you have your, sort of, rule book or guide book on how to set it right, you know? When do you work with

external companies and, you know, what are the, the, the areas that you ask them to work on, if you work with external companies?

Gilberto: Yes, I'm still working with external companies. And there isn't one-size-fits-all, and any case is different. A few years ago, I also talked with the CEO of a company that said, "For my technological part, I outsource everything."

Pawel: Everything?

Gilberto: Everything.

Pawel: Okay.

Gilberto: Yes. And yes, there were some business reasons, just to be more conservative, with the business not going well. So, "I don't have internal resources for that part." From my perspective, I think that for a startup, when we talk about a startup, it's important, it's worth evaluating the collaboration with external companies, because it's a challenge to find the right people since the beginning.

Unless you found a good match among the co-founders... Some startups are born just because there is one person that is expert on the technological part and one person about the business and the other one about the product. "Okay, let's found a startup." If you are not lucky like that, in a situation like that where the, let's say, the lead engineer that can be the heart of technology or the CTO is also the main developer, because in a startup, that happens.

In other cases, I think that it's useful to involve external companies, also to get the skills that you need and that you need at least time to find persons with those skills. And from case to case, this depends on the skills that you need, and I suggest collaboration with external companies.

I don't have preferences from the area. Of course, it would be useful to implement the domain logic internally with the internal team.

Pawel: Mm-hmm.

Gilberto: But it doesn't mean that external companies cannot provide you with developers and people that can help to speed up the development of your domain. So, I consider external companies also for implementing the core domain part, just because they can help. They can help with people for a limited time span or also for long collaborations.

Here in Tot, we are using external companies. We used it for having a boost in the implementation of our features, and we still use it for the DevOps parts, so what concerns the cloud infrastructures and other configurations. I found during the years this kind of collaboration useful, and because when you are working in a startup, you are pretty focused on your product, on what you are building. And sometimes you don't have the time to have an external perspective.

More than in Tot, in Fair, I'm pretty sure that I had a bias to say, "I'm doing everything well," so focusing on what I'm doing, looking at the results from the technological part and saying,

"Okay, this is correct." When you work with an external company, you have expert people that can say, "Okay, there is also this other option that you can try."

In another experience, a consultant can also show you other examples, other experiences with other companies that they worked with, and bring up not only the people that can work on your project, but the people that can bring their experiences that they had with other clients. And this can... We talked about diversity. This is also about diversity on the know-how that you have in the team. With external consultancy companies, you have people that work in many scenarios and then help you to build something with more knowledge that is not the knowledge of the team alone.

Pawel: Yeah, there are... I think that is one of the selling points of consulting or software services companies. Even when you talk with, uh, you know, with candidates during recruitment, that, you know, you work with many different customers, with many different projects, you'll see a lot of different things. So, there are some... Perhaps if you work... Obviously that depends on the person, but some people prefer that. Other people, as they say, are those missionaries that prefer to focus on a single product and work on it for 10 years. So, I think it's a different, you know, different character of the person. But I think you're right about this knowledge of, you know, having different customers. You can gather different perspectives, and then you can share with, you know, with your customers what you, what you saw in different, you know, in different projects.

Do you work with... In that case, do you work mostly on a time and material basis? Or do you prefer sometimes some, I don't know, fixed price where the companies deliver, like, they build something, and they are responsible for the results and they deliver? Like, sometimes that is... You know, they work on it and they deliver it to you as a final project. Or do you work as a single team, you know, on a time and material basis when the company is basically working on billable hours?

Gilberto: In all my experiences, apart from one case where I was involved with one company just to implement one single part of the product, in all other cases, I worked only with time and material. So, people from consultancy companies were part of our teams. So, they worked together and they brought their know-how with them, together with our team. I prefer to involve the external company fully on our product development, not delegate to them some parts. Because when the collaboration concludes, then the other developers have to take care of the code.

Pawel: Right.

Gilberto: So, this is my approach, and this is the approach that I followed in the past, and it worked. If you want to delegate something to a consultancy, to an external company, you must be sure that that single part can be taken from an external company and has its own life.

Pawel: There has to be some sort of handover process.

Gilberto: Yeah.

Pawel: Or something like that, right?

Gilberto: It's required. And then it's more like... If you delegate something, it's more like, "Okay, I take care about this part, but not so much. So, let's bring this part to do that and then I will check it at the end."

Pawel: Mm-hmm.

Gilberto: I don't know. I would like to be responsible about that part, too. Not just relegate it and say, "Okay, it will go well, of course, if the collaboration is good." But, at the end, also because I worked on product companies in my recent career, any piece of code, any software component is part of your product, and they should take care at the same level as other ones that are implemented by internal teams. So, for this reason, I prefer to always bring external people to our team, in order to also share the knowledge, share opinions, and make the internal team aware that there are other perspectives, as we were saying. So, improve the internal team just because of the collaboration with external people.

Pawel: Mm-hmm, mm-hmm.

Gilberto: This is my approach, and I really loved it. And there is another funny part about my previous company, not this one. Because I hired a junior developer.

Pawel: Mm-hmm.

Gilberto: And he started working with very senior developers from the external consultancy company. And when at the end this guy left the company, I told him, "You were very lucky because you worked with very talented people from the beginning." So, given that I trusted the external company and also the developers that we had internally, I was pretty sure that this junior developer grew a lot. Because I put him to work with very talented people, seniors since the beginning. And this is the right value that also external consultants can give to your company. If they have the right mindset, because, you know, you can have the wrong consultancy company that you have just people that write code, such as a Gen AI model. But in some lucky cases, external cooperation brings you value.

Pawel: Got it. So, you are not looking exclusively or only on this working as a way to minimize cost? Some companies, what they do is, like, they outsource some stuff, and probably they, you know, they offshore it to, you know, some cheaper, you know, geographic areas to save some costs, right? But then, I guess, they are focusing on the clear, you know, monetary value rather than the, you know, skill or the outside perspective, that would be my thinking.

Gilberto: Yeah. My approach is just to... It's not about money because you have to pay... Good people are not for free, even if they are employees or external companies. So, what I look for is to build something with more speed, and build something good. Because at the end, you, you are... Even if you are working with internal employees or an external company, you are building your product, and you must look more for quality than for, let's say, quantity of codes.

Pawel: Mm-hmm.

Gilberto: This depends also on the strategy of the company. But in general, I'm working on companies that take care of the product, because it's part of the business. And so it's not so...

I think that it's not a matter of finding the external collaborators that are cheaper, but finding the external collaborators that are good for your business. And to build something with high quality.

Pawel: Got it. Now, when you look... Probably when you look on the webpage or marketing materials, all the companies are the best at what they do, right? That's what every company claims. There is no company that says, "We are mediocre but cheap." No one says that. Everyone is the best, right? And, um, how do you look in that case for a, you know, external company that you want to work with? If you were to need help in a specific area, like, um, I don't know, you said this area with this, um, you know, cloud platform or, um, cloud environment. How do you look for someone who can bring some quality to the table?

Gilberto: What I do... Of course, everything could start from the website. So, you see some use cases of the company and say, "Okay, this is the sector close to mine." And this is, of course, the first thing that you see. And when you have to choose, you look for companies that work in your domain. Because if the goal is to build, build the product quickly and not take care of the money that you spend... that is also a factor, but I hope that in most cases, is not the most important.

Choosing a company that is working in the same area is really important. As I was saying about candidates, the domain, the knowledge of the domain, the way that you approach it, it's important. And then for also guaranteeing the right knowledge of the technology, you also look at the stack that those companies and those consultants work on.

Pawel: Mm-hmm.

Gilberto: Because you asked me before why you should choose an external company. And I replied, "Because you need the skills that you don't have internally at the moment." And so, knowledge of the domain with the existing use case of the company and knowledge of the technological aspects. A strong knowledge because you pay for very talented people. You are paying for the aspects that you don't have yet. And-

Pawel: Right

Gilberto: ... for this reason, those are the main factors for the decision. Then there are other aspects that are more cultural, connected to the way that those consultants work in your company.

So, if you have an external company that gives you expert people and then they just write the code and do their work without, let's say, giving more to you, it could not be the perfect situation. In my experiences, I had the opportunity to have external people that grew, that helped me, helped the company to teach other developers new stuff, to improve the code, to improve the know-how on the technological stack. So, another thing that I consider when a company is involved is the approach of the consultants to the team.

I'm pretty open to listening to them, to having their opinions, to evaluating everything that they propose. Because if I was an expert on their sector, I wouldn't need them. So, in this way, I'm also really inclusive also for those people. And I consider them as part of the team, even if they're-

Pawel: Mm-hmm

Gilberto: ... temporary. And what I look for in that case is that they are... they can propose stuff. They suggest, as we were saying, other perspectives. They tell me more about experiments that they have with other companies with the same technology and something like that. This is something that I value a lot because you feel the contribution not only for the lines of code. And-

Pawel: Got it

Gilberto: ... This is important for the choice, in my opinion.

Pawel: Mm-hmm. That's something that you verify when you start working with the company in the very first few months, right?

Gilberto: Yeah.

Pawel: Yeah. And this... You mentioned that you are looking before you start working together, you are basically looking at the web pages, right? And materials that are published there. And I'm asking about this because that's an interesting thing. I always thought for some time that in this line of business, very few people look at the, you know, at the published webpage materials. People rely more on word of mouth, recommendation, you know, reference from someone else.

Gilberto: Yes. This is true. This is also true because I have been working with the same company in different contexts since 2017. [laughs]

Pawel: Right. Right. Sure. Yeah. When you find someone, I guess that's also what people say that when they find someone who they work with, and if they work well together, they usually stick to one another, right? For a long period of time. I think that that's how it is in the consulting industry.

So, do you have any guidelines for someone who is starting with working with an external company? Like, if you were to give advice to another CTO who is starting with a new company that they never worked with before, is there anything on your mind that is worth doing at the very beginning to make sure that this cooperation kicks off well and is long term successful?

Gilberto: I suggest what I usually do, and it's involving the external people in the work project. We make them as part of the team, as I said, and listen a lot to their suggestions and also push for discussions and to look for different approaches and understand better the whole context with the external know-how, not only the internal one. This is the main suggestion. So, I love to listen to people when they tell something and think about what they say. So, my suggestion is to make them part of the team.

And why would I say that? Challenge them about the current solution and find other possibilities to better build a product or the service. So, collect any contribution that they can give, and consider their contributions as part of the whole team contribution.

Pawel: Mm-hmm.

Gilberto: And these are the main suggestions that I would say, because probably I, I was lucky with my experience, but this approach worked for me for many years.

Pawel: Right. Right. Yeah, I mean, it's like... Since it worked for you, I think it's... it might be the way, you know, to go. I think some companies, what they do is they put the external consultant or external, uh, employees aside, and they don't give them full context. And I think it is very important, what you're saying, to, to, you know, involve these people from the very beginning. Because usually, the cooperation is longer, a longer time anyway. My experience is that even if we start working for a theoretically short period of time, eventually it turns out that there is another thing to be dealt with, another challenge, another project, and the cooperation changes into the long term anyway. Yeah.

Gilberto: Yes.

Pawel: You said from 2017, right, that you did the cooperation?

Gilberto: Yes, with the same company. At the beginning, I was... Probably I was a senior developer or the team leader. So, what I've seen at that period, given that I wrote code because I was part of the development team, I don't remember if I was yet the team lead, but I remember my first pull request when those external consultancy people joined the team. I remember something like 100 discussions on some lines of code that said, "Okay, I'm here just to learn something new." And I really enjoyed that because I've seen a great seniority from external people that I learned a lot from. And this helped me in the following years. And this is the process that worked, and this is the way that I suggest, because I learned a lot when I was a developer from external people mainly, because in my team there was...

Pawel: Mm-hmm. Got it.

Pawel: So, maybe, you know, going into a little bit different direction is... You have this interesting story that you worked at Fair AI, the company that you co-founded, but then you joined Tot as the CTO, and the company was already established, so the engineering team was there. Could you tell us maybe a little bit about, you know, characteristics or challenges that you are facing when you are joining as a, you know, CTO? So, the most senior technical, you know, leader in the organization [clears throat] that is already established, and that already has an engineering team, and that already has a, you know, specific stack and technology selected?

Gilberto: Well, it was a challenge, and a good challenge, because, as you said, yeah, in 2020 I founded a startup. And as a co-founder, I was the first developer, [laughs] and the situation that I told you before, where the co-founders had different skills, and you can build something from scratch [clears throat] with the co-founders. And I built my own team, the culture, and everything from scratch.

And it was amazing, and it was also error-prone because you are new, and you are the only one that... you feel the failures [laughs], and you are also the cause of the failures sometimes. But you learn a lot. And you also, when you... if you are a good leader, because I was the... [clears throat] Sorry, I have to drink water. [clears throat] I was saying that when you are a good leader, people can see you as a leader from the beginning, because you also hired them.

Pawel: Right.

Gilberto: And I'm talking about the technical team, and I worked with very skilled people, and I suggested my way of thinking about code, the culture, the, let's say, attention to automatic code testing, and other stuff. And it was a great experience.

When I moved to Tot, I was sure that it would be different, even just because not only the people but also the code, I'm not so skilled with TypeScript and functional programming, so my contribution, given that in a startup also the CTO contributes to the code in a certain way, could be less than in the previous experience.

Pawel: Mm-hmm.

Gilberto: But what I've seen since the beginning is that you had to join a new environment with new people or new existing code, with legacy code, and a different culture. These are the challenges. You have to learn how the company works, and you have to prove that you can lead the technological part. You must get the trust from other developers, and this is the main challenge in my opinion because you help other people to recognize you as the right people for that role, so in this case the CTO. And they must accept you.

And so yes, those are the challenges, and I think that I faced them by listening and learning from looking at how the company works, and listening to people, the developers, looking at the code, starting kind of reviewing the code or trying to understand all the stuff. And I think that I tried to face those challenges of starting a new experience with an established company.

I would like to say that, I don't know if it looks like I show only challenges, but for my personal growth, I think that this experience means a lot because when you build something from scratch, also the team, everything is easier. It's also easy to fail and make your mistakes and build your knowledge.

Pawel: And by easy, you mean that it's easy because you understand what is there, right? You have this intimate knowledge.

Gilberto: Yes, yes, because it's something like you have a green field. So, as for the code, you don't serve the people, and you can say, "Okay, I want this kind of person. I like this developer. I like that one." And so, from certain points of view, it's easier. But I think that for a CTO it's also important to work on an established company, even if it's a startup. So, yes, we are not so many, there is not a huge hierarchy, and we are agile. But I like this change.

Pawel: Mm-hmm.

Gilberto: And so, it's something that a leader should do in his or her life, at least once. [laughs]

Pawel: Mm-hmm. Do you have any, you know, recipes or guidelines that you learned along the way in order... Because you mentioned that it is important to build trust. Do you have some guidelines on how, you know, CTOs joining the established companies can build that trust as soon as possible?

Gilberto: Well, I don't know if there is a recipe. It depends a lot on the people that you find. I think that in these cases, you have to recognize the people you work with. So, depending on

the people that you work with, you have to deal with a different approach. For example, for good technicians, you can get the trust from them only if you propose something that they don't know or you propose a better approach and they recognize that you are skilled at that.

Pawel: Mm-hmm.

Gilberto: For other, different people, I think that other approaches work better, so more, less technical. So, let's say, "I am able to understand your domain and communicate with you about the domain, make questions, be interested about that, and contribute with my ideas," and they are appreciated.

I think that there isn't a general recipe. What I found useful for me and here in Tot is to, since the beginning, I was saying that, start listening to other people and take care of all the aspects in my role. So, the technological part, but also the product one. So, understand how the product works and understand all the requirements, constraints, et cetera.

I worked in the same industry in the past, but I wasn't an expert, so I started understanding how banking for small and medium enterprises works in Italy. And the main recipe is to get trust in the way that other people give you trust. So, for technicians, it's better to show that you are good enough-

Pawel: Mm-hmm

Gilberto: ... to propose something new and teach something, et cetera. Or also, accept other perspectives and try to discuss different opinions and something like that.

For people that are more interested in, let's say, soft skills aspects, for me, the communication is all, and the understanding of the project and the culture is really important. So, the effort should go in the direction of being accepted by the team, and this works, I am at the end.

Pawel: Mm-hmm.

Gilberto: And that's...

Pawel: So, there is this thing that, I don't know, when a new official takes the office and they start their tenure, there's sometimes something called the 100-day plans or things like that. It also is something that some CTOs or directors quote. Like, there is the three-month plan on what you have to do as you join a new company, or as you start when you are promoted, for instance, as a CTO or CEO.

What do you think about... You know, is there something like, in your mind, some sort of the game plan that you had when you were starting at Tot, "Things that I have to do, you know, in my first 100 days"? Is there something like that that you might be, you know, willing to share?

Gilberto: Well, I know, and I know this concept of 100 days, and I didn't have the opportunity to put that in place because I got overwhelmed by new stuff. And by nature, I was, let's say, pretty active and involved as much as possible from the beginning. So, I didn't

even take a long break before leaving the previous job. So, I was really active and said, "Okay, I'm ready to do something."

However, I don't have any kind of fixed recipe. My suggestion is to take the time to listen and learn about the company, because you are entering a new environment, and it might not be the good fit for you, and you might not be the good fit for the company at the beginning.

Of course, you had the opportunity to talk with other C-levels, maybe the technical leads. In that case, I had a very pleasant conversation with both front-end and back-end leaders. But the first days, you have to listen and understand.

Pawel: Mm-hmm.

Gilberto: And try to behave as a newcomer. So, you have to enter the company and not to say, "I want to say that, I want to do that," or some stuff like that. So, for me, it's that, enter and listen until you are able to contribute. Because if you don't know anything... In other contexts, that I know because I have some colleagues that talked with me, ex-colleagues that told me, the first 100 days of the CTO is to change the technical leadership, change people or something like that.

And it doesn't work in my opinion because if something worked well for three or four years and the company has customers that use the service and use the product, it means that it works. And so you have to learn something, and listen for opinions, and learn a lot. Listen to this process until you are expert, and you are almost that expert. Because in 100 days you cannot be an expert of something new unless you're working on a competitor in the same market.

And when you feel you are ready to contribute, contribute. So, make your plan and start proposing small improvements, changes, suggestions, and make a plan for the more strategic and with more impact on the company. So, learn...

Pawel: Okay

Gilberto: ... This is my general suggestion. I don't know if there is also a bias because of my previous career or also my academic studies, but I really love to learn a lot before changing something or proposing big changes or even small changes because you are a... there is like an ecosystem that works, and you are a new part, and you should know the ecosystem before saying, "Okay, I move this piece and everything will work." No. You don't know if it works, and you must learn beforehand if that move is correct or not.

Pawel: Mm-hmm. That's extremely reasonable. I mean, my question, like this 100 days, it automatically, you know, the action plan for 100 days sort of assumes that you, as you mentioned, that you want to do something, that you should do something, that you should make a change, that you have probably a bold change, like you said, like, changing the technology, you know, leadership or direction. And what you suggest is, in fact, quite, quite opposite. And I like it, right? I think that we often forget that, you know, it's sometimes worth learning and understanding before you start making some, you know, action or introducing any changes to the system. You mentioned, you know, the company was working for some time, so probably, you know, it has some, you know, internal structures and things,

how they work. It's probably better to learn first how things work, and only then when you get this understanding, start thinking about introducing the changes. So...

Gilberto: Yeah. I think that this is the right approach. And I think that a similar approach, it's the... I demand that from the people that I hire. So, learn how it works before proposing something new or maybe... that should be from a certain senior, let's say, engineers. This approach should be understood because-

Pawel: Mm-hmm

Gilberto: ... otherwise you're saying, "Okay, I'm able with Java. I want to introduce Java. And I don't care if we use TypeScript. Java is better." You don't know. You have to learn before everything works, which are the challenges, which are the pros and cons. And this is a general approach that I apply also for other roles, and it's what I expect from people that enter in a new context.

Pawel: Got it. So, if you were to give, you know, three, four sentences of advice to a new CTO at the company, apart from this, you know, "Learn first how things work," is there something on your mind like, you know, three, four sentences of advice for someone who starts a new CTO role?

Gilberto: Well, uh, pretty easy. Be friendly. I think that since you must be welcomed by other people, you have to be friendly. [laughs] And even if you are a leader, you have to have open conversations with others.

And then, meditate before changes. This is another phrase that I can say. And take the time to understand stuff.

And the last one, probably, acts quickly. When you have all the elements to make a strategy, make a good proposal, a strong proposal, and you have the trust of people, just act, because they will appreciate that you took a decision that they agree with, and you can move fast.

Pawel: Mm-hmm. Yeah. In a way, it actually sounds like good life advice in general, the things that you mentioned, in general context. So, yeah. I mean, I think that's, that, that's pretty useful. So, Gilberto, it was very great to have you here. I enjoyed the conversation with you immensely. I don't know about the, you know, future and other topics, but it sounds to me like there are probably also other topics that we could maybe discuss in the future. I would really like that if it happens. So, thank you for, you know, being with us and sharing your experiences.

Gilberto: Thanks to you. I really loved to answer your questions. Pretty interesting. Thank you. Thank you all.

Pawel: Thank you. Take care. Bye.

Gilberto: Bye.