[00:10] RESEARCHER:

Hi, PARTICIPANT 26. Sorry, I'm a little bit late. I'm really sorry.

[00:13] PARTICIPANT 26:

Hello. Don't worry. I was just checking if we had different time zones.

[00:19] RESEARCHER:

No, I'm a bit behind today with my schedule. I do apologize for that. If you don't have questions for me, we can start with the interview. Do you have any questions for me?

[00:34] PARTICIPANT 26:

No. I saw the document and I see what you trying to achieve. And hopefully I can be of some help with your research.

[00:44] RESEARCHER:

Okay, fantastic. How about we start with some introductions, if you can introduce yourself and talk to us a little bit about your experience?

[00:57] PARTICIPANT 26:

My name is [Deleted to preserve the participant anonymity] and my core, I'm a development programmer. My current career now revolves around software development, management of software teams, and scaling development teams to deliver web-based and mobile applications. I'm mostly in finance and gambling industry working with finance ministries and regulators of the licenses. Over the years, I learned how to implement in different scenarios, the Kanban, or the Scrum methodologies from Agile. The past four years, I was the CTO and head of research and development in the current company.

[01:53] RESEARCHER:

Okay, before we dive into the topic, let’s agree on a definition. How do you define quality in the context of agile software development?

[02:01] PARTICIPANT 26:

Well, there is the end result and the process to achieve it. The end result is the product or the software for us. The minimum is that it should meet the requirements and ideally less defects or no defects. That’s external quality. Agile is keen also on internal quality because we believe in responding to change. So, we like to have good code quality and a sustainable design to cater for future changes. Process quality is improving our software development activities and practices and the way we work. Software quality doesn’t happen without a robust and quality process.

[02:53] RESEARCHER:

Okay, fantastic. What is your opinion of Agile, what do you think of it?

[03:01] PARTICIPANT 26:

Well, from my perspective, it's very similar, like it is [inaudible], it's something that you should try to achieve on your best effort, if you want to incorporate it. And adjust it to the needs of your company because many, many people make a mistake that this system where you have to work exactly. While at the core, let's say like it Scrum they define sprints. They tell you a week, two weeks, three weeks. And it's your choice. So, it's a very useful methodology. Let's say it's a must if you still work in Waterfall model. So, but it has pluses and minuses, but it's because of the factor of the people. Because not everyone is faithful to different types of working methodologies. So, in general, it's my opinion, but very positive.

[03:10] RESEARCHER:

You mentioned something very interesting. The people factor. Can you elaborate a little bit further on that?

[03:19] PARTICIPANT 26:

Well, like over the years, I noticed that even stand-ups can be problematic if somebody is working on a task that takes four or three days. He's just repeating himself when he gets really bored with that. And people tend to forget because Agile is very idealistic for me. And they forget that you are standing here in front of everyone and you are describing, again, the same thing as yesterday, because not everybody is tracking your progress, like micromanaging you. So, you have to commit your time, even if it's repetitive, just to keep this mission going and this work going. And the people factor here is that they just get frustrated with some of the aspects. And it's not always good for them personally. And you need to deal with it. It's the management's problem to solve. Not the employees' problem. The whole Agile method.

[04:27] RESEARCHER:

You mentioned something I'd like to pick on. You said Agile is idealistic. What do you mean by that?

[04:36] PARTICIPANT 26:

It's like a contrary, if you approach Agile as a practical, complete, and direct interpretation of how you should work, you're going to end up frustrated. It's a guidance for you. It's an idea of what you want to achieve. The same as the ISO standard. Every single year in ISO, it's more simple to describe it than in Agile. Every single year, for example, if you want to keep the certification, you need to show proof that you improve. But the people factor also in those companies end up that, yeah, we made some two percent savings this year. People tried to find the shortest route to get the job done while those ideologists like in Agile, they try to structure the way how the management should deal with the workflow. But, you know, sometimes you just have to break the Agile methods because something happened in the company. And if you make yourself frustrated, like, oh, I'm a failure, we failed at Agile, many people do complain sometimes. Like, this is not Agile. This is improvisation.

[05:53] PARTICIPANT 26:

And no, it's life. Agile is a theory that you need because no company can exist only in theory. It has to have practitioners, but no good company can evolve from the practitioners without the theory. You can be a Waterfall type of old development company. And if you're not going to incorporate new trends of the theory, you're going to be left behind. You can have the best practitioners. But without this kind of idealistic I mean, ideology, like from Agile, you can get stuck.

[06:38] RESEARCHER:

Fantastic. You mentioned something very strong. We said like to follow up again. You say Agile is a must. This is a little bit ideological, isn't it?

[06:53] PARTICIPANT 26:

It is. But I would like to repeat in a contrary, if you work in a Waterfall model and you don't have anything else, like you improvise completely. Or you just manage without any kind of framework. If you have a better idea than Agile, and if it works better for your company, of your experience and you have something to measure and prove it to yourself, then it's a good choice. But if you don't have any kind of proof like a scientific approach, that you have something better than Agile then you are just anti-Agile ideally from the other side. Like you hate Agile because you hate it. So, in the companies I worked in, it was a must. We tried even in [Deleted to preserve the participant anonymity] one project, to do Waterfall, because we had not much management resource left. And it didn't go well. We even had this kind of empirical proof of that it doesn't work for us. So, you have to answer your question. Is it a must for you or not?

[08:13] RESEARCHER:

So, it did work for you at the end. That was Scrum, right?

[08:16] PARTICIPANT 26:

Yes, my Agile experience is mainly Scrum and some Kanban. Yeah, it was the best to our knowledge, a methodology that we could incorporate. Not the Kanban, but the Scrum version of it. But there are different companies that need Kanban because you cannot have break points in your world, but you need to continuously deliver your tasks from the pool system.

[08:43] RESEARCHER:

Why it did work for you?

[08:47] PARTICIPANT 26:

Well, first of all, we didn't have a situation that we needed continuous, in the contrary, for Kanban for example. The Scrum and Kanban. We didn't have the need to continuously deliver because we had audits and break points from the regulators. And we had to deliver a batch of tasks at the same time. Ship it to the staging environment. And then we got feedback. If it's approved or not. Is it violating any kind of points from the law or legal point of view or not. In the continuous, you cannot just keep changing it every single day, three days a week when you finish, because it just doesn't work that way. And let's say Waterfall model, it could work for [Deleted to preserve the participant anonymity]. But we have to go back in a planned way. Few steps back on result of the audits. Not in a forced way because you go back in a forced way in Waterfall model, like two steps back. You need to incorporate again, another block of your Waterfall planning. So, it kind of worked perfectly for the situation. I personally prefer Kanban, for example, if possible. But the factor of preference is to be taken out and you need to look at the reality.

[10:33] RESEARCHER:

Ok, great. The next question, can you describe to us your Scrum implementation? I'm interested in the process, how the process works and how do people work in this process.

[10:47] PARTICIPANT 26:

We were very close to the Scrum model. We had the stakeholders which were giving us, they had their own stakeholder to product owner log that was accepting to the development product log, in the sequence and the priority. And some of them, he was trashing to not be done because it had no business value. So, this process was exactly like Scrum. Then we had the ceremony of grooming that we were preparing tasks and estimations. And the start of the sprint people were commenting on what they can do within the two weeks, we had the two-week framework. In the beginning, we had during the MVP, our weekly model, which didn't work so much. So, we switched to two weeks. Especially when the task requirements grew over time. So, we couldn't even deliver a bigger task or a story within the one week period. And another thing, we had daily standups, of course, during those sprints, except for the ceremony of the sprint start. Because theoretically it wasn't needed. But sometimes it was. And at the end of the sprint, we had the retrospective, and here human factor comes in.

[12:25] PARTICIPANT 26:

They ended up complaining about something not related to the sprint. And it was a general chat and sometimes praising some good stuff or going off the topic. So, you just have to put it in your working hour costs that people just strayed away sometimes and sometimes they really focus on. We have both, like, you know, from the backlog to what's to do, and then finish. But in our situation, we had sprint backlog, we named it sprint backlog. Then we had a column, Returned and then QA. QA was retaining the task, and that was the highest priority during this sprint. If he didn't finish it properly or it was rejected by the QA, so what was on the fly. And then we had a Staging column where we had the information, what is exactly on stage. So, the stakeholders when they were checking stage because they were actively in the spring just checking if they could come up with some adjustments. We had a rule that when you commit, please leave the twenty percent empty for the slight adjustments. That was the policy of the company. And after the staging, we had the column which was Done. And when it was done, it was on the production. So that was quite an expanded Kanban board in our situation.

[14:09] RESEARCHER:

You talked about two things which I'd like to follow up, is the ceremonies. In your opinion, how do these ceremonies help in improving the product quality?

[14:29] PARTICIPANT 26:

Grooming, for example.

[14:32] RESEARCHER:

Yes, use grooming to start with you.

[14:36] PARTICIPANT 26:

Well, in my opinion, grooming is really one of the best meetings, the most productive, because we can actually plan and prepare tasks on how are they going to be delivered, when they're going to be delivered, and how many of them will be delivered in theory. Then we have the first sprint start ceremony that was the practice, the opinion of the developers. I decided to split it. On the grooming, only the leaders of the teams and only the management of the teams joined to save time. Because many of our developers I worked with, they didn't like too many meetings. So, we did the theoretical approach and it was quite accurate, especially that the leaders knew the capacity of the guys. We had very little number of new people coming in every year because we scaled up to fourteen people. And then the fifteenth person was four months away from the previous person. So, we didn't have this inaccuracy for the theoretical approach. And then on the sprint start, we had this period of the meeting that they were moving it and commenting that I'm going to definitely finish it at the end of the sprint.

[16:06] PARTICIPANT 26:

So those two meetings were really crucial and important. Stand-ups, well, it depends really from my perspective as a manager, it keeps me informed on what they do. But it might have a little value to highly independent developers that work for themselves. It goes for juniors and middle ones when they can quickly get information in that brief meeting with who they will talk about this task on how to solve it. And for people who have many ad hoc tasks, smaller ones that they finish within the day, and then they can bring something new to it. But it's really frustrating and it affects morning motivation for the people who are working on something that takes even months. We've had, let's say, our DevOps making an automated regulation of the whole company on the snap of the finger. Using docket technology. And it's not something you finish within a sprint or two. It took him seven months, of course, on the fly as an admin and DevOps, he was doing his own task. So, he was committing like twenty percent of his time for this. So, he got eighty percent of his ad hoc work supporting the developers and twenty percent of his own time. So, for a couple of months, he got nothing new to say on the standups.

[17:47] PARTICIPANT 26:

So, I wanted to start excluding people who don't want to join if they have been in this situation where he has nothing to say unless he really wants to come to the meeting. So, I followed it up with that I would, we had two testers in those fourteen people and one automated guy. They became the task owners. What happened is, every task, I've always had one developer and one of the QA responsible for explaining it from the product ownership perspective and testing it. And he was responsible for the correct delivery and understanding of the task. So, the product owner didn't have to message and interrupt the developer all the time because we were a whole department was here in Chechnya and the product owners, CTO, were in Israel.

[18:47] PARTICIPANT 26:

So, the Skype messaging interruptions were not working well. And when he was starting standups, he was calling the people. He is asking what they are doing, what's assigned to him. So, like four people, one stand up, separate for only for them. And if someone would see that he was called and then the other QA was calling the other the rest of the team of the assigned tasks. It shortened down the distraction of the meetings because they were never ten minutes or fifteen, when everybody was standing up. And it improved the communication between those four or five people, that they could quickly, even during that stand up, solve a problem with a quick follow up. So, I don't know if it's an idealistic violation of Agile, but from the practice reality, to solve the source of frustration of the team it worked really well. Because when our team grew to around ten people, stand-ups were terrible, ten people standing up. Then, of course, I incorporated on that meeting to have this projector and a tablet and we're showing the board with the description he was entering. The QA was entering on the tablet and showing to everyone the description of the task, and the person was saying at which stage he's in. So, I use also additionally visualization, because the theoretical is like an imaginary stand up what you're doing. It was kind of hard, especially in the mornings for people, before the first coffee to imagine what the person was doing. So that was a positive thing.

[20:36] PARTICIPANT 26:

And I find them essential. If you modified them to minimize those ceremonies for standups, the frustration of the people factor. If you just force everyone to stand up by the boat and talk about what they're doing, it's just going to waste a lot of energy and bring frustration. And the retrospective, I'm like half positive on that. If we have two week sprints, then every two weeks, a free for all forum about what we're doing is not a good idea. We did it once a month. Every second sprint or every critical sprint, because not every sprint is equal. Not every sprint has a huge importance. So, once a month rule, that minimum has to be done up to two months. So, you have to ask an Agile coach or a Scrum master, you have to understand how people want to work and try adjusting it for them. So that's why I say it's a way to try to achieve. And then there's the people who are around you. So, I find it useful, but not too frequently.

[22:09] RESEARCHER:

Yes. It's quite...it's very often, isn't it, because it's a reflection on the team and the process. And I did find it myself because I worked in the industry, it's not necessary or it's not needed every two weeks.

[22:28] PARTICIPANT 26:

Yes. Not every one week.

[22:31] RESEARCHER:

Yeah, or in one week. Because human behavior doesn't change in one week, it takes time. OK, I'll move to the next question. Do you think this Scrum setup is a good implementation of Scrum? And why?

[22:49] PARTICIPANT 26:

The general one or the one we incorporated?

[22:52] RESEARCHER:

The one you have in place.

[22:55] PARTICIPANT 26:

Still, there are things I would improve because, let's say, at the core, I continuously to want to improve whatever I do. Every manager should have it in his mind. I would say we got really close to, a very good to ideal for the company that we were in. Because as a C-level manager, you have to bring in the culture also. And if you incorporate Agile as a part of the culture and then modify it, then you will always deliver faster than any other company that doesn't have it. I could even do it with Waterfall model, if this would be something that people would like to work with. Because as long as you have people supporting the idea, and listening to them, then it will make sense. But if you will tell them, we work look like this, this and this, and you have nothing to say about it, they will not support you in any kind of, in my opinion, any kind of methodology will not work. So, we were close to both very good and close to ideal from what the company would expect from us.

[24:19] RESEARCHER:

Fantastic.

[24:20] PARTICIPANT 26:

It's hard to compare. I mean, I don't work in one thousand companies to have, like, quantitative analysis on it, and it's just one example.

[24:31] RESEARCHER:

Yeah. The next question, what do you do to assure software quality in this Scrum set up?

[24:40] PARTICIPANT 26:

From the QA perspective, in a practical situation, day to day, we're just testing and cross verifying it with expectations in the task. So, the quality comes from the planning and the description. What is expected from the developer to do? I have to work with the stakeholders inside the company to stop doing one sentence descriptions on what you want us to do. Because the QA slogan, I told them to enforce it always, that few hours of planning equals few weeks of development and save. So, the first rule of the QA, as a task owner in our company, was the responsibility to understand, fully understand what really needs to be done. Not even beyond the description. I told them, drill their heads, call them all the time. Message them, overcommunicate. So, the beginning of making a task is more important than just dealing with it in the middle. Because in the beginning, you can eliminate those return moving to the Return column.

[25:59] PARTICIPANT 26:

So that is the first step of quality assurance. If we know that we are prepared and we understand what we need to do, then we can follow up with the development process. And the development process was, in our quality assurance was, first of all, not only allowing the testers to have a staging environment to verify if the task was finished properly, but also to have his unique branch and you need the subdomain for just this only one task isolated. So, we're heavily using regression testing before having, let's say, end of sprint situation to deliver the tasks. So, our focus, in my opinion, the biggest source of good quality software is the regression process. If the task itself isolated is fulfilling the needs and then if you really know that it's fulfilling, then as a whole, you see if it doesn't work, then it's creating a conflict between other functionalities of the software. But if you straight away, like many companies do, deliver to the staging environmental, tell the testers and automation engineers to test it, then they don't know if it's only the task or as a whole, there's a conflict. So, they waste time. So, you need to have this pre stage step in the delivery process.

[27:37] PARTICIPANT 26:

This allowed me to reduce the amount of effort on checking the quality of the software on the release candidate environment. The release candidate environment, in our company, was an environment deployed on the same server where production is, but with an enclosed access. So, it was a complete simulation. You can have any kind of recreation of the production in staging, but you will never have this, there is always this at least one percent difference. There has to be always in some kind of a configuration file or in a load on your database, and networking might be also different. There's so many factors that you cannot predict, that still before the release, you need to have a cold freeze period on your release candidate also. And cold freeze never started before the green light of the staging. So now that was our quality process.

[28:45] RESEARCHER:

How about the people factor in the quality process?

[28:51] PARTICIPANT 26:

That's a very broad question.

[28:56] RESEARCHER:

Yeah, it is broad because we'd like to discuss it. For example, collaboration, the closeness. Because Agile is a highly collaborative process. Does it make people happier? Because the theory says when people are happier, they produce good software. Does it make people happier, the closeness, for example, being close to each other. Another thing is the QA are involved from the beginning. Are they involved from the beginning in your process?

[29:31] PARTICIPANT 26:

Yes. They are also included in the grooming.

[29:33] RESEARCHER:

Yes. So, it empowers the QA?

[29:37] PARTICIPANT 26:

Yes.

[29:38] RESEARCHER:

Yeah. That's what I'm looking for. These are the people factor that you mentioned. How do they help quality?

[29:51] PARTICIPANT 26:

Well, let's start with the issues I had in the beginning with QA's motivation. Because also in [Deleted to preserve the participant anonymity] and in previous companies, at the beginning in [Deleted to preserve the participant anonymity] they felt like they are just helpers. They just verify someone's valuable work with their not so valuable work. This was problematic because developers know that people pivot around them basically during the process. There comes my initiative to make them the task owners that they are the ones that make the decision that it can be delivered. They are the ones to decide what task will be done during the sprint by the developer by returning it to the Return, because automatically when you finish your task, you have to go to and clean your Return column.

[30:45] PARTICIPANT 26:

So, I brought in the importance for them, but also created this kind of collaboration between the QA and the developer that the QA is always available to the developer and the product owner, it's not. So, they create this kind of a trust. If I develop it the way you say it, and if I do it exactly like that, it means I didn't do the mistake. You take responsibility for the interpretation of the task. And this empowered them like they didn't become managers, but they became like they managed the responsibility. And that was [Deleted to preserve the participant anonymity] And they were really thrilled after a few months of trying it out. Because I always struggle with the QA's motivation because we really need them. They are really crucial, and they don't feel like it. They tend to be excluded, unappreciated, or even underpaid.

[31:47] RESEARCHER:

That’s a good point you brought up! So, how this QA developers collaboration helps with the quality of the software?

[32:06] PARTICIPANT 26:

Good question! Because it does and people under estimate its effect. The most important thing is the relationships improve immensely. They do not see each other’s adversaries but collaborators. When they collaborate they share their knowledge better. They share their understanding of the requirements and why a bug is a bug and how to resolve it. If you make your testers feel not valued, you will have poorly tested software, because the developers, they don't test so much. Because they love coding and you need to understand the nature of your team and then incorporate Scrum, not incorporate Scrum and force them to change their nature.

[33:07] RESEARCHER:

Well said.

[33:09] PARTICIPANT 26:

Thank you.

[33:10] RESEARCHER:

Next question. Do you think this Scrum produce quality software? You already touched on that. And how?

[33:24] PARTICIPANT 26:

Could you repeat?

[33:26] RESEARCHER:

Yes. Yes, sorry. Do you think this Scrum produces quality software? And why?

[33:39] PARTICIPANT 26:

I lost you on the quality software and...

[33:42] RESEARCHER:

And why? Yeah, should I repeat it?

[33:48] PARTICIPANT 26:

I understand the question. I'm briefly from time to time losing you.

[33:54] RESEARCHER:

Okay, let's drop the cameras so it can help.

[33:58] PARTICIPANT 26:

Alright.

[34:00] RESEARCHER:

Let me drop the camera, hopefully it will help.

[34:07] PARTICIPANT 26:

Can you hear me?

[34:07] RESEARCHER:

Yeah, I can.

[34:09] PARTICIPANT 26:

So, whenever I discuss this because it's a one million dollar question.

[34:17] RESEARCHER:

Yes, it is. But I'm not giving you a million dollars.

[34:24] PARTICIPANT 26:

First of all, Scrum or Agile does not produce software. It's the people who produce the software. And if your Agile methodology that you incorporate and the process is in place and in the end, you have a poor software, it means that either you are really, well, let's just name it, you are really a terrible manager that cannot lift his people up or change this Agile method to adapt to the team and motivate the team to slightly adapt. And I'm saying slightly because like the previous statement, you cannot change the nature of the person. And then become successful of it, or the boat should change the manager. Some people, of course, are not good at all in working in Agile. But I have a simple situation. I have fifteen people versus me. So, if the Agile didn't work and we are producing the software, what is the problem? Everybody is wrong and I'm right. That's impossible. So Agile does not produce software, it's the people who incorporate or work correctly with Agile to produce good software or not? You can blame the theory for the practitioner for not being able to execute it unless you prove the theory being wrong. But I haven't seen the proof of Agile being negative.

[36:08] RESEARCHER:

We coming to that in a second. Can you share with me a positive story about Scrum and he software quality, if you can mix both of them, would be nice.

[36:23] PARTICIPANT 26:

Two years ago, let's say in [Deleted to preserve the participant anonymity] when I took over and when I stepped down from the CTO to take over the R and D position and open an office and I hired all those people. We found a software house, adapted to deliver any kind of quality software that nobody would put it in his portfolio. It really trusted them. But it didn't work. So, they decided to scale a department for me instead of relying on mercenaries. So, but we took part in this software and it had no sprints. There was no Waterfall, no Kanban, no Scrum, no nothing. It was just poorly developed. And there was no testing involved, no quality checks, no unit tests, no automated UX tests, or integration tests, nothing. Just a whole REST API written in one file which had four thousand lines and that was it. And then I said, guys, OK, let's have any methodology, anything. So, what is the easiest one that we can do, and I said, OK, the most broad one is Scrum. Kanban is really hard to incorporate if you don't have a whole team.

[37:58] PARTICIPANT 26:

Let's say a whole team trained that knows each other and that you know them, and I need to find new people. So, it's the most popular. Because also it's really important that you have people who also worked in this model so it's easier to incorporate them in this. And we have this situation that with nothing, no documentation, no information about it, we broke down this whole one big REST API file and we started to chunk it and plan the deliverables over the next few sprints as our roadmap, technological roadmap. And then I told them, guys, put your product road map somewhere in the basement because you're not going to get it. It is not possible, you either do it the tech way or no other way. Because you are asking me to find ten to fifteen people within two or three months and work effectively. So that's not possible. We need to first focus on the tech and bring it up a bit. And then when I had first DevOps and four developers, I started to look for the QA and the QA got input on verifying on the fly, which we didn't have in the company before. But everything on the fly, all those tasks that were delivered during this one week sprints, which we switched later for two weeks.

[39:31] PARTICIPANT 26:

And the outcome of this story is that we slowly started to incorporate first, working in a start and stop, demo, go again cycles. So, we introduced a routine and people feel more safe and more organized in the routine. That's the first quality of the whole collaboration we have, an increase of the collaboration. And after that, we started to introduce everyday stand-ups. Like, it had to be amassed. So, we were talking with each other and not being enclosed between two monitors. It improved let's say, for the team, the information on what was happening. And in the beginning, we had a very little show of tasks. We didn't have tasks. We had this putting off the fire situation. And it was positive. We were much more informed. I was much more informed. And for me especially because the guys we were like sometimes reporting to me when I was standing like outside and just listening. And we thanks to those standups, we skipped on the small talk during the day because I was already informed, which was kind of sometimes unfair, because if I forgot during that meeting, they hold the credit that I forgot and that is my fault.

[41:10] PARTICIPANT 26:

But it gave a positive view on the whole workflow. And when we started, the last one was giving the retrospective and breaking down the Kanban board into workflows and empowering the QA more in the situation of the task owner. So, they were more motivated, and they were more detail-oriented thanks to this. In a nutshell, I would say that I personally learned that in Scrum model, I believe in Kanban it will also work the same way, is that you shouldn't micromanage. You should give the responsibility to the people. Your only job is to make sure that you surround yourself with those people who are competent, and the quality of the software will come over time. Even if you start with a huge fire and huge failure of an outsourced company that just didn't deliver, and things like those happen daily. Because in I.T., one of the things we struggle is outsourcing trust and remote employees, it's also about trust. So overall, that's our ...that's not even a story about quality. That's like making the company work.

[42:52] RESEARCHER:

Yes. It's a holistic story. Thank you. How about a negative story? You said it yourself. It doesn't work for everybody. Sometimes things can go bad. How about a negative story?

[43:09] PARTICIPANT 26

The first QA was the problem. What I mean is when she came...when I was giving you the story, I fast forward and skipped her. I mentioned straightaway that the next person that was after. But the truth about her was she was idealistic. Anything if you wanted to move the release, two hours later, or one day, she negatively approached, she said this team is supposed to work in Scrum and you're not good enough into it and she was openly criticizing people in an open office. She had more certificates and more, let's say training and even more years as a QA than I had overall in I.T. And then I realized that, OK, maybe I'm not right. Maybe she's OK. So, I tried to force it the way she tried to force it. And it backfired. Everyone was approaching and they said, we want her out. We don't want to work with her. And you need to stop behaving like this.

[44:22] PARTICIPANT 26:

So, this is why this headline with nature of the People first and then Agile as a second, it stays in my head because I did that mistake and thankfully, I didn't lose a single developer in the fragile beginnings of the company. Because I kept my mind open, through their suggestions and didn't interpret their arguments as a personal perspective. But because this is the good about a democratic approach is if you know that you have really good people around you and most of them are leaning towards some kind of a solution, decision or negative approach to somebody, then they cannot be entirely wrong. You need to listen and take a step back. And the moment when we stop working with that person and switch to more, let's say, laid off and less experienced but still a person that had a more open approach to Scrum, it really paid off because you really can't, like when you have a definition of something in real life, getting from A to B, like in mathematics, how long you're going to get from A to B, like delivery of the software. And your speed is forty kilometers per hour and so on, and then you have this X value of time. But in real life, you can trip over and go to hospital. So, you always have to plan the unplanned. That's it.

[46:22] RESEARCHER:

Fantastic. Thank you very much. The last question, it's a little bit provocative, but the purpose is not to provoke you, but to get your opinion. What do you think of this statement Agile produces poor quality software?

[46:41] PARTICIPANT 26:

Poor quality software? This statement is not provocative for me. When somebody says something like that based on his own experience, he most probably, if he was the management, he most probably made some kind of a bad decision. Either he hired not a good fitting team that they didn't collaborate well. Or he created the culture...it's not about the Agile. You can change to any kind of model and you will still keep creating software. Agile is a must if you want to name or structure something in some kind of a way that is best fitting for a company. If it's Agile, then yeah. But if Agile is not good for your company, then why did you make this decision? So, if you're producing poor software, then change the manager or change the framework. Agile does not produce software, it's people.

[47:48] RESEARCHER:

Yeah, well said. Thank you. I don't have more questions. Do you have any questions for me?

[47:56] PARTICIPANT 26:

Well, was it productive for your research?

[48:01] RESEARCHER:

Yes, it was very insightful. Thank you very much. It was very insightful and rich. Thank you very much. Yeah. So sorry I should have given you that feedback.

[48:13] PARTICIPANT 26:

Don't worry. But, the research, I have an idea but I'm not sure, but are you trying to improve Agile...or?

[48:22] RESEARCHER:

I'm trying to fit a gap in Agile. If you look at the manifesto, which is the abstract presentation of Agile, it left quality out. It doesn't call directly for quality. It calls for technical excellency. But technical excellency, you did touch into that, which is testing and code review and all of that. But it doesn't necessarily like you said it yourself, it doesn't necessarily produce quality. If we look at the Scrum guide, the Scrum guide left quality out as well. There is no direct call for quality. So, this theoretical gap is quite risky for somebody who's going to go and read these materials and try to implement these processes in place. It's something that needs to be addressed. We need to tell people that people achieve quality in Agile and this is how I achieve it. These values of Agile enhance or enable quality. The ability to collaborate, the ability to communicate, the behavior of people changes if you empower your testers, etc. We need to tell people this in order to fill that gap.

[50:00] PARTICIPANT 26:

Because it's also lacking culture management.

[50:02] RESEARCHER:

Yes. Yes.

[50:04] PARTICIPANT 26:

Because so the moment when I achieve eight people in the company, the very next person, I have to put a little effort in the culture because the culture was shaping that person's behavior already. He was adjusting, not changing his nature, but was adjusting some of the bad behaviors from the previous company was eliminated and some of the good behaviors were connected with our good behaviors. And that's true. There's no manager role also. It's like how much he's responsible for guiding those people in which direction they need to go with the quality and the collaboration. It's like, this is the cycle, get the task finished. That's the weakness, of course. I mean, it lacks a description.

[51:02] RESEARCHER:

Yes, it lacks detail. So, we are addressing that by doing the research on Agile because most of the software now is produced by Agile. It's either Agile or open source. So, if we're going to keep using Agile, we have to inject more research into it and address the details.

[51:31] PARTICIPANT 26:

Actually, complaining about Agile, what is missing and trying to improve it, is actually fulfilling. It's the main goal of continuous improvement.

[51:42] RESEARCHER:

Yes, correct.

[51:44] PARTICIPANT 26:

So even those anti Agile, their suggestions they actually pro Agile too more or less. I'm thrilled. I would love to, if you could add me to your e-mail list of publications to read about it, I would love to.

[52:00] RESEARCHER:

Sure, I will. I'm just going to check if I have your email. Yes, it's [Deleted to preserve the participant anonymity] It will be ready in July, in August, sorry. And you will get a copy.

[52:23] PARTICIPANT 26

Cool. I mean, I'd love to read it and anyway, I guess that will be all the questions I had for you. I really hope my input will help you.

[52:34] RESEARCHER:

Yes. Thank you very much. It was great input. Thank you.

[52:39] PARTICIPANT 26:

All right. It was very nice meeting you also.

[52:42] RESEARCHER:

OK. Thank you very much. Have a good day.

[52:44] PARTICIPANT 26:

You too. Bye.