[00:10] RESEARCHER:

Hi, Participant 1. How are you?

[00:18] PARTICIPANT 1:

Hi, I'm good, thank you. How are you?

[00:20] RESEARCHER:

I'm very well, thank you very much. How is New Zealand?

[00:28] PARTICIPANT 1:

Not bad, it's currently raining, but it's been beautiful weather in the past week. Thirty degrees. Can't complain.

[00:35] RESEARCHER:

Okay. It's a beautiful country. Lucky you.

[00:39] PARTICIPANT 1:

It is. I'm enjoying it.

[00:41] RESEARCHER:

Yeah, it's a lovely country. OK. Enjoy your stay. Do you have any questions for me before we start the interview?

[00:49] PARTICIPANT 1:

No, not really. I had looked through the PDFs.

[00:55] RESEARCHER:

Fantastic. OK. Let's start. Can we start with some introduction? Can you please introduce yourself and talk about your experience briefly?

[01:05] PARTICIPANT 1:

So, my name is [Deleted to preserve the participant anonymity]. I'm currently a senior project manager with Sentiance. I started out as a web developer, went through to functional analysts, and then became a project manager. Always worked in I.T. So, in the context of Agile, I touched upon Scrum for the first time when I was a developer and then helped implement it a multiple times over multiple companies. And currently, with the help of the VP operations, we implement Agile Scrum, Kanban, across the whole company on a regular basis.

[01:49] RESEARCHER:

Fantastic. Thank you very much. You implemented Agile in companies. That's what you've said, right?

[01:58] PARTICIPANT 1:

So, the first time we implemented it was with the [Deleted to preserve the participant anonymity]. It was for a specific project. We set up Scrum practices in the existing development teams. I also helped an [Deleted to preserve the participant anonymity]company. And now we basically help teams implement certain aspects of what helps them deliver the best products.

[02:24] RESEARCHER:

What's the difficulties of implementing Agile, which is a culture, actually? What are the difficulties of implementing such a culture in companies that sometimes are difficult to change?

[02:44] PARTICIPANT 1:

I think mainly getting everyone on board with what you want to do is probably the biggest hurdle. It also depends on what it is exactly you want to do. Agile is quickly used as a word, as a Scrum, as a Scrumban. It's differently understood by different people. And usually management wants one thing, the developers want another thing. They both have very different opinions about what is what. And we try to find the middle road or the best road towards delivering a product to the customers. So, I use the word Agile and Scrum and Kanban in different ways with different people. So, it depends.

[03:30] RESEARCHER:

Ok. Thank you. The first question. What do you think of Agile? What is your opinion of it?

[03:40] PARTICIPANT 1:

As I just mentioned, that depends on how you frame it. I think that the core values as stated in the manifesto are great ideas. I do believe in delivering products incrementally and working with people rather than processes. But there's also a lot of frustrations that come with it.

[04:04] RESEARCHER:

So, yeah, we'd like to hear about those frustrations.

[04:10] PARTICIPANT 1:

I think it's mainly everyone has their own opinion, I think by now it's very well ingrained in most media or senior professionals. They will have some experience with Scrum and that may be good or not so good, and they may have good opinions, bad opinions and same goes for management. But to me, I think the core ideas are good and they can be used effectively. They can be used to develop, deliver good products. But it's a slippery slope. It's also very easy to say, hey, we do Scrum. It's five rules. Follow this and we will do great. It's a very naive approach, but that you see a lot with managers that want to go fast, they have a new team and they just want to get going.

[04:57] RESEARCHER:

What's the consequences of going naive and about it, and implementing something wishy washy? What's the consequences?

[05:12] PARTICIPANT 1:

I think in most cases, for example, is to take the easiest way to strum. The easiest way to do is, is to read the manual really quickly and what hangs out with you is we all going to stand ups, we do a four hour planning every week and then at some point maybe we bring in the product management and you try to fly over all the things that make Agile good like talking with the customers, helping people do their best work. And instead you go to planning a lot of meetings. And I'm sure that then everything will be fine as long as everyone attends the meetings and we treat them the same as we treat all the meetings where one person talks, and all the others listen. That's just you fall into the same pattern that you had before, but now you're calling it Agile because instead of calling it the meeting, you call it the stand up.

[06:08] RESEARCHER:

I've seen that, I've seen that in a lot of places.

[06:15] PARTICIPANT 1:

Yes. So that's why if you ask me, what do you think of Agile, it depends.

[06:21] RESEARCHER:

Given the subject of our conversation is quality in agile, how do you define quality in the this context?

[06:30] PARTICIPANT 1:

It is a challenging question, because quality is a subjective concept. But for the purpose of a definition, I would say quality takes place in the process level and the software level. During the software development process, we implement and exercise practices that assure quality. The agile process itself is subject to continuous improvements. We use retrospectives to continuously reflect on the way we work to become better at what we do and ultimately better at delivering quality software. That’s the process. For the software or the product, I believe the key attributes is conformity to business needs and free of defects. I’d like to say, in agile we also focus on the internal software quality. The final product may confirm to business needs and have fewer defects but it doesn’t always mean the internal quality is acceptable. We have to add the desires to maintain, understand and reuse code to evaluate quality.

[07:01] RESEARCHER:

Let's move to the next question. You can choose any one of your experiences to answer this question because we want more specifics. We wants to understand the specifics. Can you describe to us your as our environment?

[07:21] PARTICIPANT 1:

So currently we have a development team of fifty to sixty developers, depending on the time of year. We work in an AI environment. So, it's data scientists and engineers, but also some developers. They're split up in teams of three to eight people and they are self-steering. So rather than saying top down, we're all going to do Scrum, we're all going to do all these things, we tell them everything that's available and we try to guide them a little bit on how to implement certain Agile artifacts. Like, do you want to do a stand up? Do you want to do planning? Why do you think that's helpful? Why do we think it's helpful? But in the end, teams choose themselves. So, the end result at the moment. But it is always changing. Is that every team may do something different? But in general, they default back to some more senior teams, to just Kanban while other teams, they really feel Scrum helps them and talking with product managers, talking to customers more and other teams, like DevOps teams, keep working on a very upper basis. So, it's a very landscape, but I think we're going in the right direction. The best parallel is maybe SAFe, scaled Agile framework. But again, we don't implement it to the letter, nor do we intend to. We're all about helping the teams do the best they can.

[08:17] RESEARCHER:

Ok. Fantastic. Next question. Can you take me through the journey of a requirement or a user story or a feature from inception to release?

[08:35] PARTICIPANT 1:

Sure. So, a user story or a feature can come from both ways, they can come from top down. So, from the management they say we need this feature because we think clients will buy it or clients have requested the feature multiple times. And that usually from both directions, that goes through the product management. Since we have project management then also delivery, we kind of get a say in how we prioritize these features. There's a whole bunch of parameters. I mean, that's product management stuff. In some way, they will order the backlog. They will order the features and they will request the most important feature according to them. And that will then be given to those certain development team, depending on the feature, depending on the requirements. And then someone from product management will be in the planning meeting. If it's, for example, a planning meeting every two weeks, they will be in there.

[09:39] PARTICIPANT 1:

The team will discuss what they want to do in that meeting, estimate how big the task will be. What is the kind of smallest demo that we can do to the client that will make the client pay for the feature? Because maybe they're just like, oh, we would like this thing, but we don't know if it's useful. And then they start work. If product management decided or the client decided we want this, the team is happy with the minimum specifications, they will try to get a first version out in, say, a week or two weeks. It could be just a wireframe, could be a working kind of terminal script depending on the feature. And then we work from there. If it's no good, we'd burn it as quickly as possible. If it is, how do we change it, maybe we further work on top of that and then we tried to bring it into production as fast as possible. It can be for just a specific client and then start testing it. Give it to the client, also give it to the data testers. So that's what you're after?

[10:45] RESEARCHER:

Yeah. So where does QA fit in this journey? You didn't talk about it. How do you assure the quality of these deliverables and the quality of the feature?

[11:00] PARTICIPANT 1:

In this company, we have an AI platform and there is a mobile component that sends data into the platform, so depending on where the feature is, it's a bit different. We're going to take an example from the platform, let's say. So, if a client wants a new data point set from the platform, there will usually be one cycle behind of development. There will be a QA team or there will be a canary release to clients that have signed up to testing or beta testing. So canary release that it only goes out to specific clients, one at a time or two at a time. And then we also have a test team that works one cycle after development teams, so say, development is one week cycles, then the QA guys will start testing it right afterwards. And then keep working in that way.

[12:03] RESEARCHER:

You don't engage the QA from the start?

[12:07] PARTICIPANT 1:

It depends. Currently, we have a very mature platform. So, they already have their processes set. If there is really experimental stuff, usually it's a very close collaboration with the client. They're creating an QA team. But most features currently are built on top of the very robust system. So, the QA team has their set scripts. They have all their things they want to do. For example, we output segments for customers like, a lot of kids, you sport a lot, or you eat very healthily. If there is a new one, they kind of know what to do. They know what to check. We will warn them, or we will inform them as soon as we start on the feature. But then they kind of know what to do. And there is always one QA guy in the team.

[13:07] RESEARCHER:

Ok. I'm just looking at my next question. Do you think this setup is a good implementation of Agile? And why?

[13:19] PARTICIPANT 1:

It's good. There's always room for improvement. I think we do deliver software and value quickly. I think for me, that's one of the important points. I think that people are happy with how it works. They feel like they can do the best work they can in the team, at least from my perspective. I can't read their minds, sadly, but they seem pretty happy with the setup. We managed to push out products a lot faster than before, even if they're in a little bit less finalized states, the clients that do get in touch with those features are aware of that and they're fine with it. But there's always room for improvement. We have a lot of biased managers, a lot of biased people. I mean, this is a point that I keep bringing up again, that they have their own opinions about how things work in Agile and what they want to do and how they want to work. Some just want to sit at their desk and they don't want to be bothered by standups, client request, changing features. They just want one road map for the next half year and then sit and do that work. But I think looking at the results, looking at the happiness of the clients to happiness of developers and the work produced, I'm thoroughly happy with the way it's implemented.

[14:44] RESEARCHER:

Do you think that the maturity of the product that's helped?

[14:50] PARTICIPANT 1:

Yeah. So, I have helped a company that has a mature product and I've helped one that has scaled from startup to a more mature with a lot of clients. And that is by far the most difficult part. It works really well, I think, in small teams, startups, where everyone is very motivated. They know everything changes. They can adapt really quickly. But trying to implement and keeping on top of Agile together with growing a company, growing teams, changing directions is very challenging. So, in this case, yes, it was helpful that we were just after the initial growth. So, we had the bigger team. We know where we were going. We had the bigger clients set. So that made it a bit easier to make it more robust to kind of fine tuning to make sure we speed things up. But I've been in the mess that is scaling a company as well, and that will always be a challenge.

[15:57] RESEARCHER:

Okay, next question. We touch a little bit on that, but I will ask again, because we want a bit more details. What do you do to assure software quality in this Agile setup?

[16:14] PARTICIPANT 1:

So, the main thing, as I said, is that we have testers and we have multiple environments. So, we have a test environment, acceptance environment, canary environment. So, the very close customers can get access to the features early. And then we have a production environment and all of these environments have rigorous testing. There's also unit tests and there is someone from QA on board of every team, having feature requests from the start to the end. There is a pretty rigorous checklist, if you will, of everything that a feature has to go through before it goes to production. Is there anything specifically that you want more information about?

[17:03] RESEARCHER:

For example, once the backlog items are decided or groomed, the QA is involved, for example. Do you call for the QA to be present? That's presence of the KIA is very important because you empower him or you empower her from the start to be engaged, to be collaborative. Does this happen, for example, at this Agile setup?

[17:40] PARTICIPANT 1:

So, in some teams, it definitely does. So, again, as I said, they kind of choose how they implement it. But the QA person is always involved from planning to delivery. So as soon as the team hears about it in the team, so when the team hears about the feature, he will be involved, he will start thinking about how to test this feature from start to end. You want to make sure that all the quality aspects are taken into account. He is also responsible for maintaining the quality over the iterations. So, the first iteration may just be a bit rough, but then he takes full ownership of the quality of the output. If we expect eighty percent accuracy and he some doubts if we can reach the accuracy, if he can measure it, he or she, then we will be in discussion with them to see how we can tackle the testing process so we can deliver that quality. And if he is unsure of then quality, then have to steer our metrics so that he has confidence in the quality as well.

[18:52] RESEARCHER:

This engagement from the beginning of this early engagement of the QA team or the QA resources, does it change behavior?

[19:05] PARTICIPANT 1:

Well, that depends on the team. I think, sometimes it feels to some developers that it slows them down, that they have to stick within boundaries, that they have to be more rigorous. They can't really experiment too much because QA will be there right away with, this is not the way to go. You have to reach this target and stick on it. In other ways, it can push them a little bit to fine tune to reach the quality assurance. I can think of one team where quality assurance is a bit less involved. They're more about trying to find new features to trial and error. They're a bit more free flowing, the wild card team, and they always have problems with QA. So, they claim they have a nice new feature. Now we ask for metrics. How accurate is this? How often can we expect this to work? How do clients make money out of this and then they kind of throw it over the fence and say, well, we'll figure that out later.

[20:15] PARTICIPANT 1:

So, while I think the developers in the team may see it as a bit of slowing them down in development. I think definitely from project management, everyone else can see the added value of the QA. And in the end, also developers do as well because they know the target to hit. They know how to hit it, they have someone there that has the expertise to help them reach the quality assurance. It may sometimes give some...I'm looking for the English word...not fighting, but these are disagreements. But that's fine. That happens in every team, happens in every development.

[21:05] RESEARCHER:

Ok. Can you share with me a positive story about Agile and quality and software and delivering software quality?

[21:16] PARTICIPANT 1:

A positive story. So again, about those segments, if we have clients that request segments or they want to know a specific thing about their user base. They want to know who is a dog walker, so who goes out to walk his dog every day. That was one of the feature requests and we didn't really know how to tackle that. It sounds like an easy problem to solve, but it turns out to be rather complex to find out if someone has a dog or not. So, I don't think I really mentioned that. But the company kind of analyzes people's profiles based on cell phone movements, so you just keep your phone in your pocket, you go about your day and buy GPS location, we can figure out basically a lot of aspects of your life. One of those is, well do you go and walk with your dog?

[22:13] PARTICIPANT 1:

And the company needs to be eighty-five percent sure that if we said this person is a dog walker, that he was actually walking with his dog. And I think before this specific team settled on an Agile trajectory, it was very hard to get some stuff on the product backlog. It was very hard to get them to move in a different direction because they had a fixed path. We're talking about a data science team here. If they start on something, they have to research and they're gone for a few weeks and then they come back and maybe they have an idea, maybe it doesn't work, and they have to start all over again. So that kind of steered themselves without too much guidance. So, we decided to get in there and really work with

them, try to explain to them why we think Agile has some added value even in data science. And in the end, we were successful in finding a way for them to work with experiments.

[23:17] PARTICIPANT 1:

So, there's no way to deliver a working machine learning algorithm every week or so. It's almost impossible, but you can try and make experiments with certain notebooks. I'm not sure how much technical detail you want here.

[23:37] RESEARCHER:

The more detail, the better.

[23:40] PARTICIPANT 1:

OK, sure. So usually the team would have to build a whole machine learning pipeline to build a whole algorithm to output certain results for a segment. And then it would take weeks. I mean, you need training data, you have to label it, you have to train the model. Wait for the output to tweak everything, and go over and over it again. And there are kind of these mindset of doing it this way because we've always done it this way. But the client really wanted some feedback a bit faster than six months. Maybe we have something, we want to be involved. That was the right moment to get the team to think about how we can deliver something, some value to the client a bit quicker. So, we change it up, not even massively. It turned out they were already internally doing a lot more than they showed on the outside. I think once you start probing development teams, you'll find out pretty quickly that when they talk amongst themselves, they're always talking about the things they just did, about the results they managed to get. And in this instance, they were already working with Python notebooks, for example, and sharing intermediate results. But they didn't share with management or with clients because the results were hard to explain.

[25:09] PARTICIPANT 1:

The results weren't good enough. They didn't want to commit to anything. So, in the end, we basically just found out that there was a whole box of, not deliverables, but interesting information, added value for the clients just sitting there that was not being shared, just because they thought it had no value. And by just reframing a few things here and there, by making them think about, OK, how can we explain this over and over again quickly, efficiently? How can we show these results in a quick way? Like just print them out to a PDF, add two lines of explanation and give it to the client. We enabled them to share more with clients, internal teams, and management on a weekly, daily basis about their progress. Whereas usually there would be a request for a feature that would go into the team and six months later, we would hear it didn't work. And then we had to start digging about what they were doing. We now had a good cycle going on every week. We would get a status report, some graphs, some insights on how this worked, how this didn't work, and this is the quality we're getting now. We are aiming for two percent more accuracy by next week by using these heuristics. And just everyone was way more involved in the development process rather than five data scientists mining through data and approaches by themselves in the logbooks.

[26:45] RESEARCHER:

Ok, fantastic. It doesn't always go rosy and happy. There are certainly some negative stories. Would you like to share with us a negative story?

[27:02] PARTICIPANT 1:

There was another team where we kind of had a junior, so I also try to help some junior project managers to guide the teams. And we put a junior project manager on a certain team and he just read the Scrum guide and decided to give an hour long presentation about how to do Scrum to a team of senior developers. One of the most senior developers of our own company, he had fifteen, twenty years of development experience. And they all kind of sat through the meeting and then the guy started shooting out, stand up meeting invites, planning invites, basically booking the whole team's agenda full of meetings.

And in the beginning, I think they joined one or two daily stand ups and then nobody started showing up anymore because apparently, he was just talking the whole time with the team. I think that's fine for the project manager. It's fine to walk with your head against the wall. I think that was a textbook failed implementation of how not to give Scrum to developers.

[28:36] RESEARCHER:

Yes. And the lack of engagement.

[28:39] PARTICIPANT 1:

Yeah, very much a lack of engagement. And there was already an existing distrust of any buzzwords related to Agile and Scrum and Kanban in that team in the first place because they are very senior. They already have heard about it a thousand times, about how it's going to magically fix all the development problems and make everything great. And that is the point where you have to talk to the developers and work with them rather than give fancy presentations about the latest Scrum guideposts you read about. So that was one about the implementation. There is also a story about a client where he did the same thing. So, as I said, we kind of label your life based on your G.P.S. location from your phone. But we also made a new branch in two predictions. So, if we know that your routine is usually, you drop your kids off to school, you go to work and then you go home and something minor changes. Maybe we can then start to predict that, hey, you are expecting another kid, or you are changing homes, you are going to live somewhere else. And there was a client who was very interested in this feature. We thought we explained clearly about what they could expect from our weekly updates so that we were going to let the developers take the lead.

[30:11] PARTICIPANT 1:

And that was the same case from my previous companies. It would be like an account manager or a client services person. They talk to the client. They put everything in nice fluffy clouds and give it to the client. Well, we are very clear and said, look, you're going to talk to the developers. They will be a bit more direct. They will show you the numbers. The numbers may look a bit less than what you want in the end. But know you are working towards it. And we did a few sessions with them. So, they were in the review sessions for the feature and they got to see the development plan. This is what we did this week. We took this trajectory because we think we're on the right track. This is the quality we have at the moment, this is the accuracy, this is the failure rate. This is what we can expect by next week. And there are pretty quiet in the first meeting because that's kind of normal for clients or contact people that aren't used to the process, they don't really know what to expect. Well, we thought it went fine in general. And I went on for about two hours in those meetings a week. We found out that they had been trying to contact their CEO very aggressively and complaining about how everything is falling in the water.

[31:44] PARTICIPANT 1:

Are we going to miss the deadline now that everything is going wrong? How would they feel the team's not on top of things? Just complaining about the lack of results in general. Even though from our perspective, we saw that we were going to make it. We were going to make the deadline and we were going to make the accuracies. It was all going to be fine. The indications we gave to the product person from the client, he sent it through to some people internally and so on and so on. They saw accuracies that were way too low, that they could never use to make money. It all kind of started rolling where everyone was yelling at everyone just due to a lack of misunderstanding. Not really knowing what it is to get a weekly update directly from developers without a lot of rose clouds. So, in the end, I think it all ended up okay. We sat together and explained it to them again. We did decide to just make an executive summary of where we were at every week rather than take them to the review sessions of the feature itself.

[33:06] RESEARCHER:

Okay. Fantastic. That was interesting. What do you think of this statement? This is a little bit provocative, but I think you are a very mature professional, so you would understand what we're trying to do here. We're just trying to get you to talk. So, what do you think of this statement: Agile produces poor software?

[33:36] PARTICIPANT 1:

No. I see where it comes from. But I think it's a poor excuse for poor software. I think every system, every team, every approach can deliver poor software. So where would this come from? I would assume that a lot of people that have been in contact with Agile, as the client I just mentioned. They see software in between the delivery dates. They think it's not good. Where else can it come from? Because certain aspects of Agile are overlooked. They just think it's all about standups and planning meetings. But in the end, we're just doing what we always did, which is Waterfall and deliver a product after six months that the customer hasn't touched, that QA has never seen. That basically came out of nowhere and just popped up. And then everyone is like, oh, what is this, I've never seen this before. It's not what I asked for. It's totally different from the requirements that I expected. I wouldn't say that Agile by itself delivers poor software, but it may be misunderstood.

[35:00] PARTICIPANT 1:

What it will deliver on a frequent basis and due to that, it's interpreted as being better or superior to other systems, because I think there's a lot less sugarcoating in Agile, in my opinion. You should try to involve, the businesspeople that are not used to seeing the low numbers, seeing all the failures, seeing everything that goes wrong, and then thinking that all that's all because Agile and they tried to go too fast. They're not planning enough where they're sitting in meetings too much. I think I've heard everything from all sides, either it's there's no plan, there's no thirty page document of how we're going to tackle this project. And I've heard the other side, they are doing a four hour planning every week. Why are they wasting so much time on planning? So, it's kind of funny that you're hear all the excuses from both sides. And then you've got people that are very happy with all of the frequent updates, which usually are more technical people. And then the people who say, what is this spam in my e-mail box? What are all these meetings I'm asked to attend every week just to say that, yes, this looks good or no, this is terrible? So does Agile deliver bad software? I don't think so. I think poor teams, poor management, poor everything can produce poor software. And using Agile by itself is an excuse for software is misguided at best.

[36:53] RESEARCHER:

That's what well said. I think it has to do with the performance of the team and it has to do with the people in a great extent. Blaming the methodology, it's easy. It's easy to blame the process or the methodology itself when you are not capable of implementing it, right?

Yes, I agree with you.

[37:25] PARTICIPANT 1:

So, I think in the end, it's all about the people and culture. Hopefully, you recruited great developers and you want to make them do great work, as fast, as easy, as fun, as good as possible. I don't care what you want to call it. I don't care if you want to call it Waterfall, Scrum, Agile. I don't know what else. Tribes. Any kind of buzzword that you want to put on a slide as long as you have people front and center and you don't try to hide behind the processes. That you're clear about the product increments, the features you're delivering. You're always telling the truth, you don't try to sugarcoat then I think you're on the right track. You can use whatever methodology you like, if there is layers of sugar coating and hiding behind emails and processes and it's not going to work. But some people seem to disagree.

[38:30] RESEARCHER:

Because they experience the failures, I guess. So that's why they disagree. So, we shouldn't blame them. Yeah. Yeah, it is. It is normal. It is acceptable. But it's usually not the process failure. It's again, the people and the team failure. It's not the process. The process maybe in some instances not a good fit. Because as I said, it's also a culture. It's a way of work and a way of behaving and the way... If you don't like to collaborate, so why would you work in an Agile environment, for example? You're just an obstacle to the whole process. So, you may want to work for somebody else.

[39:27] PARTICIPANT 1:

And then you get it to the whole thing of change management. There's someone, some manager has read a book about Scrum and all of a sudden there is a box of post-its and there it is. We go in the Scrum and they're surprised that three months later and everyone's running around confused and thinks everything's broken. Now, there is a lot of things that can go wrong in a lot of places. But I think that's true for everything. You can do the Waterfall that we've been doing for hundreds of years, tens of years, and I still do it wrong. And the same is true for Agile and the same is true of the next thing. If you haven't talked to everyone, if you haven't thought about how we're going to do this and is everyone on board, you can mess it up. At least in Waterfall you can hide for six months and then you have to explain why that didn't work in, In Agile, you don't have the luxury of hiding for six months.

[40:29] RESEARCHER:

You are accountable.

[40:31] PARTICIPANT 1:

Yeah, you are accountable. Even in the best case, you only have like three or four weeks before someone will start ringing the alarm bell. So, I think if it is only that, that you gain, you've already gained a lot. At least to get you talking about process and systems.

[40:53] RESEARCHER:

I don't have a further question. Thank you. That was a very interesting chat. Do you have any questions for me?

[41:02] PARTICIPANT 1:

Yes. So where is this going? I mean, I read that it's going in a paper and you're kind of researching about QA and Agile.

[41:11] RESEARCHER:

Yeah. So basically, I'm interested in software quality in general, how software teams manage to achieve quality. So, if you look at theoretically, there is a gap. What's this gap?

The gap is come from the Agile manifesto itself and the Scrum guides. They are very abstract, and they don't make a direct goal for quality. The Agile manifesto makes a call for excellence, but excellence is very broad, it is very vague. What we wanted to understand is how teams that use Agile handles quality. And not only that, you've been talking about software testing is fine, but software testing does all the quality control. It's making sure that what you are delivering, fits the requirements and fits other standards and other best practices, et cetera. But there are other social and human factors to all of this. You've been talking about collaboration and lastly, you mentioned accountability. All these are enablers that help us work better. When we do work better, we do deliver better quality. We are happier. The peer to peer behavior is enhanced. And we do behave better, and we are happier, we produce better software. So, these are the elements or the enablers we're looking for Agile too. We talked about collaboration, we talked about accountabilities. With other interviewees, we talked about the loop of learning or the ongoing loop of feedback and learning from the feedback and feeding back the feedback and making sure you continuously improve in Agile. That's a quality assurance practice to enable to be transparent and sending feedback.

[43:50] RESEARCHER:

So, what I'm trying to understand is to pick up these enablers of quality that are in the process itself other than testing. Because testing itself, it doesn't achieve quality. Like we discussed at the end, there is the human aspect of it. At the end, the people who are making the software. It's not Agile that is making the software. It's the people. So, when you give transparency, for example, is another element that appears in most of my interviews. When there is transparency and everybody is able to see what's happening, people work - I've been told that the work is nice. So, I'm assuming that we produce better deliverables when there is transparency. Why? Because you are accountable for what you do. Everybody sees what you do. So, these are the enablers I'm looking for to understand in these environments and to counter these statements that Agile produces poor software. Well, Agile produces poor software, because you don't collaborate, you don't have transparency, you are not accountable. Your QA team is not engaged from the beginning. It's not empowered. It's sitting somewhere in India and you don't talk to them at all. So, we're bridging that gap that is missing. And the body of knowledge, which is how Agile enables an environment where we can achieve quality.

[45:44] PARTICIPANT 1:

Very interesting. So, can I send you an e-mail and then get the paper once it's finished?

[45:51] RESEARCHER:

Yes, I think I do have your e-mail from your resumé. Just give me a second. I'm just going to confirm your e-mail. Its [Deleted to preserve the participant anonymity], right?

[46:11] PARTICIPANT 1:

Yes right.

[46:13] RESEARCHER:

I'll send it to you once it's ready. It may take a few months. It's not a fast process. Unfortunately, academia is very slow. If you don't have other questions for me, we can conclude. Thank you very much.

[46:33] PARTICIPANT 1

Thank you, have a great day.

[46:34] RESEARCHER:

Enjoy New Zealand. It's a beautiful country.

[46:37] PARTICIPANT 1:

Ah yeah, will do.