[01:28] PARTICIPANT 2:

Hello.

[01:28] RESEARCHER:

Good morning.

[01:29] PARTICIPANT 2:

Hi Researcher. How are you?

[01:32] RESEARCHER:

I'm good, yourself?

[01:34] PARTICIPANT 2:

I'm good. Where are you located?

[01:37] RESEARCHER:

I'm in [Deleted to preserve the participant anonymity]

about yourself? You are in the UK?

[01:42] PARTICIPANT 2:

The UK, yes.

[01:44] RESEARCHER:

PARTICIPANT 2, do you have any questions for me before we start? I sent some instructions and clarifications. I'm not sure if you had time to read them, so if you didn't feel free to ask me questions.

[01:58] PARTICIPANT 2:

Yes, I have had a quick look. I would rather leave it up to you to drive the interview.

[02:08] RESEARCHER:

Yes. I will drive the interview. I just thought if you have questions from the start. Ok, let's start. OK. Can you please introduce yourself and talk to us about your experience?

[02:23] PARTICIPANT 2:

Yes. So, I have been working in the I.T. industry for 14 years. My first six years, I was a software developer. So, I was hands on, full stack. I'm not sure if you know about technology, about.

[02:42] RESEARCHER:

Yes, I do. I used to be a software developer as well.

[02:47] PARTICIPANT 2:

I worked with Microsoft.NET, MVC, C#, SQL Server, JavaScript, CSS, HTML. But then in 2012, I started taking on the roles of Agile Scrum Master. Since then, up to now my management experience around Agile has been evolving. So, I was a scrum master and for the last few roles I've been doing coaching, been doing project management around the agile and scrum master too because it's a role that you accidentally you end up doing.

Even if you are a coach or project manager, you are working with a team. You need to ensure they deliver and eventually you end up doing scrum must then. That's in a nutshell, my experience.

[03:52] RESEARCHER:

Yeah, that's sufficient. Just a quick follow up question. What motivated you or what were the circumstances that made you move from a software developer to a scrum master or to an Agile software project manager?

[04:12] PARTICIPANT 2:

Good question. So, when I was a software developer, I found myself working in very ineffective environments. We had to plenty of overtime. We had to cope with inefficient management, and I thought that things don't have to be that way. I had a chance to work here in London in a very good company who wanted to change. And they started introducing agile practices and applying Scrum. I saw that the collaborative spirit of Agile, the Agile values force cooperation, experimentation, and not a blaming culture, which is what I was hoping for up to then. I really liked the idea of their version of Agile so that's what motivated me. The fact that I could change working environments and make them more collaborative, avoiding over time, and avoiding management practices. That was my core motivation.

[05:37] RESEARCHER:

Ok, fantastic. All right. What do you think of Agile?

[05:43] PARTICIPANT 2:

Well, I think it's very misunderstood. Because is not a framework, it is not a scrum, it is not Kanban. It's not scaled Agile. To me, Agile is not that and that's what many companies don't

understand. Agile is more of a culture. A company can be very agile and not practicing scrum because Agile is about fostering a culture of cooperation, breaking down barriers between teams, making them more collaborative, happier, removing the blaming culture and make management more transparent. Address what you do with value and collaborate constantly with a client. One of the worst things that you may face in this world, in this profession is that the software teams start working on the product, they don't communicate with a client, it's not their fault. Often it happens that there be a lot features in the system, but the client hasn't had constant feedback. So, they create a lot of overhead in development in building the product, then you find months down the line, what they created is not what the client wanted. So Agile for me, creates that constant collaboration and feedback from the client. That's is what Agile is to me, it's a cultural thing. Things are there to support it, but that's not all there is to Agile.

[07:39] RESEARCHER:

These are quite interesting and superior values. As an agile coach, how do you coach a team, or how do you implement them in a team? Because they are quite abstract.

[08:01] PARTICIPANT 2:

Yes. Yes, they are indeed. When you join a company and assign a coach, that is a great indication because it means that a company is open to change. So, the fact that you are hired as an agile coach is half of the way. But still, there is a lot to do. And the first thing an agile coach must do work managers to ensure they influence the culture of the team. For example, as a practical example, I work in a company where the development team was eighty people in four teams. That means four teams of twenty people. So, they had their front end team, the backend team, QA, and DevOps. That means that there was a strong interdependency between the four teams, and they were blocked all the time between each other. And that's not agile at all. And I had to work with management to make them understand that a single team must be able to deliver an end to end feature - from the front end to the backend to the testing to the DevOps, instead of relying on independent teams for each one of the part of the system. You may have heard about this concept, cross-functional?

[09:40] RESEARCHER:

Yes.

[09:42] PARTICIPANT 2:

You need to work with management to make them understand these flaws in their overall organization. What are these flaws going to cost months down the line on what to do to avoid that? What we need, an example, is to restructure all the teams. We went from four team of twenty to ten teams of seven, eight people. Each of teams were focused on only one feature. So, we ended up having payments team, parking team this was for our car automobile app. So different teams for different parts of the system and management, they needed to understand this. Once they understand it, they are in the best position to influence culture and drive that gradually. Does it make sense?

[10:57] RESEARCHER:

Yeah, it does. But sometimes you do have an organization culture that clashes with these values. So, what would you do?

[11:08] PARTICIPANT 2:

Well, as I say it, if they brought you on board means that they are really quite open minded. If some of the changes you think must be applied, if they are closed to that, they don't want to do that, then you need to work things around and find alternatives. For example, I worked in another company who didn't want to do this. They wanted to stick to backend teams, frontend teams and so on. So, we found an intermediate point. We found a way to deal with this. So in between the backend teams, for example, we split the teams in payments, subscriptions, and overall performance of the backend. So still, we would have the backend team. By the way, that was like ninety people. A backend team of ninety people. We split the teams it in sub-teams. But apart, we had the frontend teams. So, see the difference. Instead of creating fully cross-functional teams with mobile, backend, QA, we had to create sub-teams within the large teams. So, the point is that if they don't want to drive the change that you are proposing, you need to work with them to find an intermediate point. The worst thing you can do is argue and create a resistance.

[11:50] RESEARCHER:

For the purpose of this interview. We obviously going to talk about quality. Then, my question is how do you define quality in the context of agile software development?

[12:01] PARTICIPANT 2:

It is at the same time complex and subjective concept. But a straight forward answer would be quality has three aspects: product quality, internal quality and the process by which we deliver the software. Product quality can be simply defined as being defects free and add a business value or at least meets the user needs. Internal quality has to do with the way that the software has been constructed. It is a much more concrete attributes like clean code, simplicity, component reuse and flexible design. It can be assured through predefined standards, linting tools, unit tests etc. The software development process is very important because it is the vehicle that drives achieving quality. In agile, we believe that the process is a continuous improvement job. From my experience, organizations do not get the process right from the first go. I think the best process is a robust process that is the product of continuous reflections, learning and adaptation.

[12:56] RESEARCHER:

Yes, that's a very good definition. I agree with you. Yeah. All right. Let's move on to some more detailed questions. For this question, you can choose any of the projects you worked for or any of the companies you worked for. So, can you describe your Scrum environment, the setup, the teams, the process, how they work together, etc.?

[13:26] PARTICIPANT 2:

Yes. So, I will go back to the company where will we split into cross-functional teams' example. Before any changed happened, there were eighty people, four teams, twenty people per team. They had something similar to an Agile board in Jira. They weren't writing user stories. They weren't doing daily stand ups. None of that was happening.

And when I worked together with management to encourage that cross-functional split, we ended up having ten teams with seven people. So, more teams, smaller. And we brought product owners, saw the product owners would create the stories for each one of the teams. We brought Scrum Masters to start practicing Agile within each one of the teams.

I was one of them and we had another Agile coach. So, we went from large teams, no process at all, no backlog, no stories, high dependencies to smaller teams, cross-functional and doing scrum. And able to communicate with the other teams. Once a week, we would do a scrum of scrums. So, one representative from each of the teams would join a meeting and we would talk about the progress of our team and about the dependencies across all the teams. And we will resolve the dependencies in the meeting and create a task in Jira. Does it make sense?

[15:30] RESEARCHER:

Yeah, it does. Yeah, that's very interesting. Can you take me through the journey of a requirement or a feature or a user story from its inception to the release?

[15:50] PARTICIPANT 2:

Yes. So, this can change across companies. Product management thinks that they need a specific to feature for a client. For example, the ability to login to an app using your phone. You enter your phone number, you receive a unique code. And with that code, you log into the app. So that's a feature. The first thing you need to understand from product management is how important is that feature. It's a must do. There is no way the product can go to the market without that feature. Usually you need to write the requirements, work with that team to make them understand the requirement. Create the user story or an epic. Let's say an epic because it is a big feature. Write it up in Jira and work with a team to break down that epic. And create the user stories from that epic. And the team does a first-time rough estimate on each one of the user stories. For example, login with a phone number, that's the general epic. So, the teams would say, well, first we need the user to enter the phone number. We need the user to receive the code by text message. We need the user to login with that code. Those are three different user studies. You put them in the backlog and then the teams, they estimate each one of the user stories in size normally. Have you heard about t shirt size?

[18:02] RESEARCHER:

T shirt size?

[18:04] PARTICIPANT 2:

Large. Very large...

[18:05] RESEARCHER:

Yes, yes.

[18:07] PARTICIPANT 2:

This is done as a first approach to estimate the user story. They say OK, this story is very large, and they define their label very large without details. Then they have the stories in the backlog. The product owner must establish the priority of those user stories. They give them absolute priority and put them at the top of the backlog. And they will be picked up on the next sprint. The team needs to estimate in a deeper way, these user stories. To estimate if they can actually do it in the next sprint or it needs to be spread it across different sprints. Does it make sense?

[19:11] RESEARCHER:

Yeah, it does. I'm listening to you. I will have a follow up question, but keep going.

[19:17] PARTICIPANT 2:

That's end to end before development. After that, the teams, they develop the user story. They show the functionality to product management, all the stakeholders. If the stakeholders are happy, then that story is done. And can be planned for release for the next deployment. It's a bit complicated in words. It would help to have a picture.

[19:54] RESEARCHER:

I worked in agile environment. I understand you. I just wanted to hear it from you. So where is the QA team in this process? I haven't heard you talking about them.

[20:06] PARTICIPANT 2:

The what, sorry?

[20:08] RESEARCHER:

The QA. The Quality Assurance Team.

[20:12] PARTICIPANT 2.

For me, the QA is a constant thing because everyone needs to ensure quality. QA starts from the beginning. I know that developers may say that they are not QA people or anything like that but QA starts by the developer doing a good job. And if we talk about specific testing, good practice is for the developer to do unit testing on every function he or she develops. That function must be tested. Then once that feature and user story is developed and tested, the developers release it to a testing environment. And there the QA team picks up the new features deployed to the environment, and they perform testing. Sometimes manual, sometimes automated. If it works, they deploy it to staging. And in staging, sometimes the stakeholders

they test in staging. Sometimes there is another additional step, which is UAT, user acceptance testing. That is a different environment.

[21:53] PARTICIPANT 2:

And the stakeholders test the feature from a user perspective in that environment. They approve it. If they approve it, then we can say that functionality is done. But if we see the release lifecycle as a set of steps in different environments, you have the local machine from the developer to testing environment to staging to UAT and then obviously in each one of these environments, there has to be a QA process so that they give the green light for that feature to be moved to the next environment.

[22:44] RESEARCHER:

Sorry to interrupt, but I would challenge you in here a little bit, but forgive me for doing that. It sounds a little bit of waterfall, because if I were to implement QA in this process, I would involve the QA from the story write up. So they can be familiar with requirements and they can be empowered and engaged from the beginning.

[23:17] PARTICIPANT 2:

You are right. I absolutely left that out. When I say the team gathers to talk about the story, an agile or scrum team must typically put down these details. It means developers, QA, the PO, scrum master, everyone. So you are right. QA must be involved from the start.

[23:51] RESEARCHER:

What have you observed from your experience when you involved QA from the start? How does it make them feel and their behavior? How does it change their behavior?

[24:05] PARTICIPANT 2:

I think there are more good things than bad things about getting them involved. Good things is that they know what's going on, they know what's coming. They know the functionality that they are going to have to test later on so they are not as strained in the future. Now, there is a roadblock because that sometimes they get confused when it comes to estimates. Many times the QA don't want to estimate because they don't know from the development perspective, they don't understand how big or small this story is. Sometimes their estimates are random. And it looks like they go along with what all the people from actual development say. Are you familiar with fibonacci series?

[25:07] RESEARCHER:

A what, sorry?

[25:09] PARTICIPANT 2:

Fibonacci series?

[25:11] RESEARCHER:

No.

[25:12] PARTICIPANT 2:

Ok. It's a way to estimate that story. It's a way to a size. A story can be talking about abstract terms. It can be very small, small, medium, large, very large. So, you would find sometimes that the developers say, OK, this story is very large. And the QA person says large. But the QA person normally can't formulate a solid opinion on why that story is large and not very large. Because he or she doesn't have a view on the depths on how to develop that story. That view is more solid from actual developers. So therefore, the QA person is involved in the sizing of the estimates. But you would find that normally they don't feel too attached to the actual process because they don't understand really. They feel like, OK, I will go along with flow but I'm not going to feel very strong about my estimates.

[26:32] RESEARCHER:

Ok. Fantastic. You talk positively about this setup, about this example. Why do you think this setup is a good implementation of Scrum and why?

[26:49] PARTICIPANT 2:

The setup as in the way this thing works.

[26:52] RESEARCHER:

Yeah. The implementation of Scrum.

[26:55] PARTICIPANT 2:

I think it's a good thing because first there is collaboration and forces people to meet and collaborate. Only because of the simple fact that you make people stand up in the mornings. That is what we call the daily stand up. It's fifteen minutes meeting every morning and it happens in which the scrum team, including QA, the PO, the developers. But you would find that development teams often don't do that. And the fact that you make them stand up and talk to the rest of the team about their progress on what they are doing, what are they going to do today, what are the blockers? Everyone gathers in a circle and they give updates to the rest of the team. And starting only for that simple fact, it makes everything better because people naturally start collaborating more. So that's the value in the setup.

[28:18] RESEARCHER:

You raised two important concepts, which is collaboration and accountability. And they make things better. I like that. Can you elaborate a little bit more on how they make things better?

[28:38] PARTICIPANT 2:

Well, because if you don't communicate, if you don't collaborate, then what happens is that they develop things that are wrong. When they find out it's wrong, they're not working well. They start blaming each other, ah but you didn't tell me, but you could've asked. And they are getting these into resentment arguments. So, it is great that once a day you force them to collaborate.

[29:15] RESEARCHER:

How about accountability? Does it change behavior?

[29:19] PARTICIPANT 2:

Of course, it does. Knowing that other people are expecting you to deliver, you may be more committed. If you give someone something to do, OK, go to your desk, nobody is going to ask you for this, no other knows who is going to need it, that person is not going to do a good job. If he or she feels they don't have a job at all, but after you know that other people need what you implement, they are going to use what you are doing, that motivates you. It fires you up. You know that you are working on something for you, you are working for other people. You are accountable to the rest of the team. See, the culture is that everyone is accountable for everyone. The rest of the team is accountable for you. And you hold accountability for other people too. That, to me is great. And if you can keep the size of the teams to ten people maximum, that is even better. They larger the team, the less sense of closeness you have.

[29:28] RESEARCHER:

How does this help achieving better software quality?

[29:32] PARTICIPANT 2:

Yes, it does. I'll explain more and use the developers as example. When the culture is that everyone is accountable for everyone, developers feel accountable to meet the team's expectations on quality. In other words, delivering substandard code quality is letting the team down. Accountability boosts the individual motivation to meets expectations including the quality of the code in the case of developers. I have seen it in practice, developers write better code when they feel accountable.

[30:58] RESEARCHER:

That sense of closeness is very important. Yeah. It changes peer to peer behavior, doesn't it?

[31:04] PARTICIPANT 2:

Yes, it does.

[31:06] RESEARCHER:

How does this collaborative environment help achieving software quality?

[31:10] PARTICIPANT 2:

Collaboration helps better quality by facilitating intensive exchange of information and feedback which help resolve and minimize bugs. Not only bugs, but I've seen developers helping each other's and learning from each other's how to write better code and resolve complex issues. Sometimes they are design questions or how to make a piece of code simpler and maintainable.

[31:16] RESEARCHER:

Do you have any example?

[31:18] PARTICIPANT 2:

Yes. In a company I was working before the current one, a payments company. We were developing a set of work services for implementing ASPSP imposed by the European Union around internet payments. We were implementing the backend services and the teams, when I joined, the team wouldn't talk at all with QA people, who were based in India. Right there is the first problem. We had development teams in London, but QA were in India. So, they were blaming each other all the time. When I spoke to the developers, they, they said the QA people don't a plan, they ask what to test. So, I spoke to the QA team and they said similar things about the development team. That was producing bad quality software and tested software. They would develop a feature and the QA team would give the green light, but it wasn't working. One of my roles as an agile coach is to bring them together and participate a daily standup, everyone together. And plan together. So, they would update each other. So, part of an agile coach is to help bring the people together.

[33:01] RESEARCHER:

How did you manage to do that in an offshore setup?

[33:07] PARTICIPANT 2:

Logistics wise, we were using Zoom. Every day we setup the standup using Zoom. We had headsets and camera. I encourage people to use their camera because it's nice to see everyone's faces. Logistics wise was very simple. You find a time that works for everyone and send out the meeting invites.

[33:44] RESEARCHER:

Have you noticed that this closeness and this collaboration that you've set up has enhanced the QA and the developer relationship?

[33:57] PARTICIPANT 2:

Yes, definitely. Well, it was like a day and night difference. The most important thing that QA could develop test plans knowing what the functionality was. I would organize the standups, the scrum meetings. I would organize when needed a conference call between the developer and the QA person and they would talk to each other on how the functionality works. So, QA could elaborate test plans that make sense. Whereas before, QA wouldn't elaborate on any test plan. They were assumptions.

[34:50] RESEARCHER:

And that creates clashes.

[34:54] PARTICIPANT 2:

Yes. So, QA people need to know what to test, what are the inputs on the expected outputs? They come up with their own assumptions otherwise and they're not going to test consistently. The problem is that if no one is there, it's so simple, if no one is there to facilitate that communication, it just doesn't happen. If found that people don't talk. They use Slack or email for some communicate. Scrum compels people to collaborate and overtime it brings them closer to each other's.

[35:31] RESEARCHER:

Ok, let's move to the next question. We already started talking about it, but we'll go back to it because we need to discuss this in-depth. What do you do to assure software quality in in this Scrum set up, for example, what did you do to assure quality?

[35:52] PARTICIPANT 2:

Well, a number of things. Improving quality in the development practices. In that way, you need to allow the development team to come up with a good development practices. It was just me that one would come up with that when I was on the technical side. Now, the development teams I work with, they do that. They need to have good development practices. I'm not talking about the detail guides on how to name the variables, because creating excessive documentation may delay the team. But I'm talking about general guidelines so that's something that I need to do. You can use software that detects automatically and evaluates the quality of your code.

[36:58] RESEARCHER:

You're correct.

[37:00] PARTICIPANT 2:

SummerQ is one example. You can use metrics as the number of tests that you run every time you deploy something. DevOps comes now into the picture. And you deploy something. It is a great practice to have prepared a number of automated tests that tests the feature that you are deploying. If all the tests pass, then green light and that feature is deployed. If it fails, it prevents

that feature from deploying. So, there's a number of good practices you can implement across the QA process. Obviously, the QA team needs to have a prepared test plan. Either automated or manual. I don't really get too deep into it because the QA people know better what to do. More often than not, they have part manual and part automated. It's not only about the automated testing that happens when you are deploying in DevOps mechanisms. If it has been deployed, the QA team gets their hands into a feature and run their own tests. Once it's all in, the users or stakeholders, they need to use the functionality and say, OK, this is what we expected. So that's ensuring good quality.

[38:53] RESEARCHER:

Ok. So how about the human side? How does collaboration, accountability, closeness, the client being involved etc.? How all these attributes of agile help achieving quality?

[39:17] PARTICIPANT 2:

Good question, again, because it is it is easier to foster collaboration within a single team. And it is to do it across all the pipelines. Your development team may be very collaborative between themselves. But they don't talk to the stakeholders and the stakeholders may not talk to the client. So that's why I was keen to do, what they call in Scrum terms, a sprint review. You don't need to call it that. You may say a biweekly demo or some demo or something like that. It's a meeting when the development team meets together with the stakeholders and they work together to show the functionality to approve it or come up with new ideas, come up with things that they expect to be in it, or were misinterpreted. The point here is to promote a predictable and repeatable event in which development teams and the stakeholders meet together. In scrum terms, that is a sprint review. And it happens every other week.

[40:57] PARTICIPANT 2:

What I find more complicated is the stakeholders meeting with a client. It's a bit more complicated because Agile would naturally tends to work more from the team side to the stakeholder side. So, what the stakeholders do with their clients, often it's a bit of a black box, even for the agile coach. And you do have the chance to be the one who would push for that collaboration, the stakeholders and client. And you may find yourself setting a monthly meeting or monthly conference call with the stakeholders, you, and the client. You talk about what you delivered, what the client expects, what's coming next. So, when it comes to the stakeholder communication to client to summarize things for you, that is where I found my biggest challenge. Because not always, I had the chance to be the one who promotes that collaboration.

[42:23] RESEARCHER:

Interesting. I agree. I'll move to the next question. Do you think this Scrum setup produce software quality?

[42:40] PARTICIPANT 2:

It's ensuring you follow good practices. It does promote good quality. Having said that, I found that most companies, they say they are agile, but they're not. Most companies think that because they have a Jira board then it means they are doing Scrum.

[43:08] RESEARCHER:

I've seen that. I've seen very interesting and pretty boards, and I've seen that messy and very, very chaotic boards as well.

[43:25] PARTICIPANT 2:

Exactly. Agile is not about picking up independent practices. It's not about cherry picking, things and putting them in practice. If you do that, then everyone is Agile. And that's what many companies do. They say we are agile, and we want to bring a Scrum master or an agile coach in to improve. But I do find that the only Agile thing that they have is a Jira board. An Agile environment must promote collaboration and transparency across everything, development teams, stakeholders, clients, management. That is Agile. If you do that, then you will produce great results. If you break the chain in some way or different parts, it will not work just as well. For example, I was speaking before, the QA team was in India, the development team here in London. They did standup, but only in the development team.

[44:42] RESEARCHER:

Yeah, I've seen that, too.

[44:46] PARTICIPANT 2:

This was the developers. They were sitting together and they're just talking to each other. But they didn't talk to the product owner, they didn't talk to the QA team in India. We had people in Spain in the same team and they wouldn't talk to the people in London. So, in reality, the picture was more complicated because I had to bring together people from London, people from Spain and people from India. So, yes it does work. It does promote good quality, but if you actually do it. It's actually having to implement the culture.

[45:35] RESEARCHER:

You've already shared with me some positive stories. Unfortunately, it doesn't go rosy all the time. You must have some negative stories to share with me. When things didn't go well.

[45:54] PARTICIPANT 2:

Yes. So, in my last company I worked for, we had a backend team in Sweden. And there were ninety guys doing only backend. In London, we had forty people only in the frontend. But obviously, this was not two separate entities, they needed to collaborate with each other. So, one of the experiments I was doing was to do some sort of stand up in between mobile and backend. But the problem is that we are struggling to identify how to optimize those standards because the teams were structured. There were two huge teams who knew structure. So, it was useless to do anything similar to a scrum between the two teams. Because one person or one team would welcome many different things. There are people in the backend who welcome many other different things that we found that a cross-functional team doing the scrum is not going to work. And that is why we decided to split the teams in between the backend team. And split the teams in between the mobile team. Does it make sense?

[47:36] RESEARCHER:

Yeah it does.

[47:41] PARTICIPANT 2

So, as you say, it's not always roses and rainbows.

[47:48] RESEARCHER:

No, no, unfortunately is not. And in your opinion, what are the organizational factors that make it go bad?

[47:57] PARTICIPANT 2:

It can be many. But to me, mainly, it's about the mistakes that are initially done, and are ever corrected. If you set up initially a team of five mobile developers in London and another team of five backend teams in Sweden, and you never do anything to bring those people together. Then you find yourself with a two huge teams in different countries. And who never talk at all, and they have never talk at all. Years and years of this mistake until that mistake produces a big ball of dirt that now you can't do anything and all you can do is be focused in that ball and do whatever you can do within that to rearrange things. That is why I would say the culture is very important because if you don't start fostering and influencing Agile culture from the very start, then things can evolve in a situation where there is no quick fix, or there is no fix at all. All you can do are hacks.

[48:08] RESEARCHER:

The, how the absence of an agile culture, like you said, influence the team’s ability to achieve software quality?

[48:17] PARTICIPANT 2:

Very good question! Simply people do not collaborate and this mean more bugs because the QA and the developers do not communicate efficiently. The developers make a lot of assumptions about the business requirements because they do not communicate efficiently with the end users and so on.

[49:37] RESEARCHER:

Fantastic. That brings me to my last question. It is a little bit provocative, but the purpose is not to upset you, but to get you to talk and to tell us your opinion and your perspective, because that's what we are after. What do you think of this statement aside: Agile produces poor software?

[50:06] PARTICIPANT 2:

I think it's a pretty vague generalization. It is true and it is not true. At the same time, yeah.

[50:16] RESEARCHER:

So, tell me when it's true, what happens and when it's not true, what happens?

[50:23] PARTICIPANT 2:

It's true in that normally companies don't know what it's like to be truly agile to foster a culture of collaboration, transparency, and pedicle environment. And pedicle environment means people are free to test and try and steer and learn without being blamed. On the other side, it's a culture that companies make the mistake of thinking that agile are frameworks not culture. Agile is scrum. Agile is not Kanban. Those are frameworks that are attributed to Agile in that they are putting practices, their values, and the culture. If you don't have the values and the culture right from the start, you don't have a solid foundation. You'll be in the building, you need to put a solid foundation first. And that is the culture, the values. When it comes on the top of the building, those stores, those are the frameworks. If you put the scrum and there on that very weak foundation. It's going to crumble. And it's not going to produce good quality software. That is why many people are becoming more skeptical about Agile, because they associate Agile with frameworks, with scrum, Kanban, XP programming, SAFe. Now, of course, this is not going to work if you don't get the people in the right mindset. If your leaders don't believe in these values, it sounds a bit abstract. I know, but it is true.

[52:29] RESEARCHER:

No, I understand.

[52:31] PARTICIPANT 2:

So that thinking is very true. Yeah.

[52:37] RESEARCHER:

There are two scenarios you've been talking about. When it's true, the second scenario, the statement is not true?

[52:47] PARTICIPANT 2:

Yeah, the statement is not true. It's the opposite. I have found that example what I told you where we had four large teams, eighty in total, four teams of twenty. We transformed that into ten teams of seven or eight. In that scenario, I found that is not true because I worked with leaders, with the CEO, the CTO, and they influence the culture of the company. It was it wasn't me just shouting around the table, or going around asking people to do things.

[53:28] RESEARCHER:

So, there is leadership, there's commitment from the leadership.

[53:33] PARTICIPANT 2

Exactly. If there is commitment from the leadership team that sets that field for our success for an agile implementation. I mean, it doesn't solve everything. It's not going to be easy. You still need to get to the people or the teams to work in a collaborative environment. But it does help massively otherwise it's just me shouting around and trying to change things. I can only change things within a set amount of people, not across everyone.

[54:14] RESEARCHER:

So, my summary is my statement is not true. When the implementation of Agile truly embrace the qualities and the values of agile.

[54:30] PARTICIPANT 2:

Yeah, yeah. When those cultural values are being promoted by leaders, there's the foundation, is the values and the culture.

[54:48] RESEARCHER:

That was my last question. That was a very interesting dialogue. Thank you very much. Thank you. Do you have any questions for me?

[54:59] PARTICIPANT 2:

Yes, this is some very interesting thing. The agile research is a very interesting thing you are doing.

[55:10] RESEARCHER:

It's a research project which is going to lead to an academic paper which will we'll try to submit or publish in one of those software engineering journals or conferences. Basically, we're trying to address the gap of Agile. If you look at the Agile manifesto, it doesn't make a clear reference to quality. It does make references to excellence, but excellence doesn't equate quality all the time. So, when we look at this Scrum guide itself, which is assumedly meant to be an instance of Agile or an implementation of Agile, it doesn't make reference to quantity either. So, we do believe that this is a gap that needs to be addressed in the body of knowledge about software engineering. But like this interview is a good example that agile teams care about quality. They do have quality measures and practices in place. So, what we wanted to understand is Agile itself as a culture. How does it promote quality?

How does it implement quality, et cetera? So that's what we set up to try to understand.

[56:56] PARTICIPANT 2:

And this response has been promoted by the university.

[57:03] RESEARCHER:

Yes. Yes. It's a project in my university. I work for the [Deleted to preserve the participant anonymity]. I'm a postdoc and this is my project. So, in a postdoc, you do a project research project, and this is my research project.

[57:22] PARTICIPANT 2:

Ah interesting. And I can have access to what you are going to produce?

[57:27] RESEARCHER:

Yes, you can. And I'm just going a put a note here against your name because I have a list of people I talk to. And once the paper is ready, I'm going to send it to you.

[57:45] PARTICIPANT 2:

OK, very interesting.

[57:49] RESEARCHER:

And any feedback is welcome once I've sent it to you. It will take a little bit of time, but you will get it. That's for sure.

[57:59] PARTICIPANT 2:

Ok, let me know if I can be of some additional help or something.

[58:04] RESEARCHER:

Yeah, just the last thing I will send the interview transcript to you if you can have a look because that's a quality and validation things, we do is to make sure that when we transcribe the interview we didn't misquote you. We didn't make you say something wrong or something you didn't want to say. And it's an opportunity for you to add or to withdraw some of your statement, if you like, it's just the quality of things we do. So, I will send it to you and if you have a look at it and tell me if you okay with it.

[58:43] PARTICIPANT 2:

Ok. Interesting. Thank you very much.

[58:47] RESEARCHER:
Thank you, PARTICIPANT 2. Have a good day.

[58:50] PARTICIPANT 2:
Have a nice day and weekend.