[00:07] RESEARCHER:

Hi, how are you?

[00:09] PARTICIPANT 8:

Hi. Hello. Hi, I'm fine. And you?

[00:15] RESEARCHER:

I'm doing very well, thank you very much. Thanks for your time and the opportunity to talk to you today. I really appreciate.

[00:22] PARTICIPANT 8:

You're very welcome.

[00:26] RESEARCHER:

Do you have any questions for me before we start the interview?

[00:31] PARTICIPANT 8:

No, I have read the instructions and the questions, so I have a picture in my mind it will go, so I'm fine for now.

[00:40] RESEARCHER:

Ok, fantastic. So, let's start with the interview. Can you please introduce yourself and talk to us a little bit about your experience?

[00:50] PARTICIPANT 8:

Yes, well, I am currently working as a delivery manager and in the company called [redacted]. I have broad experience working in various Agile environments. I've been working in [inaudible] cloud and they were using the less Scrum methodology and before that in STS-365. This is betting industry and there organizing and introducing the process to the organization. So, this is briefly about me. If you want me to go into more details or something particular.

[01:40] RESEARCHER:

No, it's fine. I would have follow up questions. So, my first follow up question is you worked in different Agile environments. What is the difference in various implementation of Agile? How did the organizations take Scrum or Agile in a different way?

[02:02] PARTICIPANT 8:

Well, from my experience, nobody actually takes it by the book. Everybody has some adjustments. And I think this is perfectly normal because of the nature of the business and the needs that every particular organization had. So, I found a question on your list about whether Agile is a good practice to be implemented for companies. And I'm just paraphrasing. Yeah. I think that it is good to get the elements, you know, that can be our views for an organization. You just can't take something and implement it straight as it is the book and just follow all the rules even though this doesn't bring you anything good and harms your organization, it just doesn't bring you any results.

[03:15] PARTICIPANT 8:

So, it's just not a practical thing to get and stick to something that is just written and said that you should do something every day. But there are some good elements and I think it can used to help the organization grow and have structure and stuff like this.

[03:28] RESEARCHER:

Let's get some definitions first. We will be talking about quality, it is important to agree on what we mean. How do you define quality in the context of agile software development?

[03:34] PARTICIPANT 8:

My understanding is you want to understand how we achieve quality not the means by which we achieve it. Right? Then, it is product quality. But process quality is one of a number of contributors to product quality. You can’t achieve a quality product without a process. Product quality is how well it conforms to the product requirements, specifications, and ultimately customer expectations. But for software it is important to mention free or fewer defects. Process quality focuses on how well the software delivery process is working. In agile, we keen on continuous learning and improvements. The go together. We always try to identify how we can improve the process.

[03:48] RESEARCHER:

Alright. Let's get to your opinion about Agile. What do you think of Agile in general?

[03:54] PARTICIPANT 8:

Well, I think it's a good framework. I think there are some very useful practices that can be implemented. I think it's good to be familiar with all those practices. And as I mentioned before with Agile, you don't even have to apply all of them. You can just take what can be applied to your business. And something that you can benefit off.

[04:32] RESEARCHER:

What makes it a good framework, you qualify it as a good framework? So, is it a good framework for software development and what makes it a good framework?

[04:45] PARTICIPANT 8:

Yeah, I think it's good if it's applicable to your business. If you want to have an iterative approach and if you have the opportunity to take the feedback from your customers or from some of your users, then it can be a very good framework. And if you know how to apply it. So, I think it's more about the mentality of the team. And how is it structured? How is it team cemented in order to call something that is a good or bad framework? It just the framework. It's something that gives you high level guidelines. Nobody says you have to do something. You can adjust it to your needs. It's perfectly fine. For me, it's commonsense, you know, to take something and just adjust it in a way to give you the results that you want.

[06:00] RESEARCHER:

So, you touch onto something very important, which is the mentality of the team. Can you elaborate on that a little bit?

[06:09] PARTICIPANT 8:

Well, if you come to the team and just tell them, OK, now we are going to practice Agile. We're going to have meetings every day and you're going to make Jira tickets in such and such way. And they just don't understand why we're suddenly changing the way we're working. It can bring confusion and resistance. and stuff like this. But then again, you have another approach to explain why are you working, why are you doing something, what do you want to improve and what results you are hoping for. And then you have a team that is open to accept new things, to experiment and try to be better. The team that is open to a critique or suggestion of how to improve themselves, like individuals then work is a lot easier. And this is something that I consider the mentality. You know, the openness to try new stuff and experiment.

[07:29] RESEARCHER:

I agree with you, if the openness to try is not there, it's a recipe for failure.

[07:37] PARTICIPANT 8:

Oh, well, yes. There are other things that influence, like business priority of things that are on the road map, the market, so all the pressures that are coming. Sometimes you just don't have the space to experiment, you don't have time. You have to just do something else in certain ways. But this is a whole other sphere where we can talk about this topic as well.

[08:19] RESEARCHER:

Ok. All right. Let's move to the next question. So, can you describe to me your Agile environment? We are after the process, how the process works, and how do people work in the Scrum process? You can choose any one of your experiences either the current one or the previous ones.

[08:41] PARTICIPANT 8:

Ok, let's go for previous ones. It's more, I think, more by the book. Well, it was contained of three months' planning and prioritization of requests that are in backlog. And this was done by the principal product owner. And my role in that organization was to communicate with customers and gather the requirements and evaluate how important those are. Meaning what are they bringing to us and then a comparison of what would be if we don't do something. So, in such a way, priorities were determined. And then when we built the backlog, there were few of us working as product owners. When we have a bigger common backlog then those lists of requests for a project were prioritized and elaborated with the development team? Because their feedback was needed in order to have the full picture of how much does it cost to build something in developers' time. Also, security and QA are involved in these processes since there feedback also influences the priority of particular things.

[10:31] PARTICIPANT 8:

And then once the plan is done and we have a road map for three months, then we would start with the implementation of those requests were that were planned for the period. Then this three month planning period was divided to sprints of two weeks and at the beginning of every week, we'll plan what will be done in this two weeks. The rule was to finish everything with development three, four days prior to the sprint end so that QA can test everything.

[11:22] PARTICIPANT 8:

This was on the paper. So, in practice it wasn't always the case. And will it go round with this process for those three months. After which everything that was built was deployed to staging and then some integration tests were run upon the code that was deployed to staging, and then afterwards it was deployed to production, if everything is right and communication of new features that were implemented.

[12:11] RESEARCHER:

Ok, great. You touch a little bit onto the QA and being engaged. At what stage do you engage your QA in the setup?

[12:22] PARTICIPANT 8:

Well, I was engaged as from the beginning. But QA management because they were needed to give feedback on the complexity of things that are planned to be done. So, they're giving their opinion and they're giving the estimation, which leads to why issue later on, when a QA person will be working, writing tests, and testing the feature comes to start working on a particular feature. Those estimations are not always matching because of those people have different stage of knowledge and expertise. So, this is a point where we were facing challenges now because somebody gives you one estimation and then it comes to you who are actually supposed to work on something and somebody else gives an estimation for you. And you just need more time. And this is where we had some challenges that we need to resolve.

[13:49] PARTICIPANT 8:

So, QA was like a separate team, but every development team had their dedicated QA. So what QA were doing during the sprint, it was planned on dev team level. Then they had that another manager.

[14:18] RESEARCHER:

So, I've noticed something very interesting that the QA are engaged from the beginning and they are given a voice. The QA provide their opinion from the start. This is in contrast to the traditional method where the QA are engaged at the end, most of the time. Does this enhance the cohesion and the collaboration and the team? Does it also change behavior of the QA and the developers?

[14:54] PARTICIPANT 8:

I believe it really enhances cooperation. It's still a struggle to have them fully involved. Exactly because of the case that you are mentioning, because traditionally QA is taking more testing when development is finished. But they are just used to be quiet and just work on something that they're supposed to. But this led to a problem because sometimes you have deadline and you have like two or three days dedicated to QA, but it just is not enough. And you have to meet the deadline and then you put all the pressure into QA to finish everything or the case when you agreed to finish in three days before the deadline, and development is late. They are going through this deadline of three days and then QA have to compensate for that. So, this is another challenge that we're facing.

[16:22] PARTICIPANT 8:

So, involving QA at the beginning and just hearing their side of the story of how much they need, how much effort they need to finish a particular task or their valuable feedback that you can get from them as to what can go wrong. What can complicate things during the way, then this really serves for somebody who is planning the development, product owners and developers in the end. And to have in mind what can go wrong and pay attention to things and to be aware that they need to finish something and leave time for a person from QA to finish their work. Because if not, the whole thing is going to be a problem because they're one team, you know, QA is part of one dev team. And I think this really empowers this collaboration and enhances this teamwork.

[17:26] RESEARCHER:

So, how this early participation helps achieving quality?

[17:30] PARTICIPANT 8:

They become intimate with the requirements. This allow them to efficiently test the software. Not only knowing what to test but they make less assumptions about the requirements, which also means less bugs and faster delivery.

[17:46] RESEARCHER:

Very true. I agree with you. Can you take me through the journey of a requirement or a feature or a user story from its inception to release?

[18:03] PARTICIPANT 8:

Yeah, there are two kinds of requirements. There are internal requirements that need to be done in order to keep the system functioning and improvements that you need for work time. The other type of requirements are the requirements that are coming from a customer. So, which one do you want me to elaborate?

[18:36] RESEARCHER:

The one that comes from the client.

[18:40] PARTICIPANT 8:

From the client. We had clients that are working with us. The product is such that you just have cooperation for a long period of time, like two, three years. So, you're constantly working with a client. And during their usage of the platform, they come to some obstacles. They need some improvements, etc. They are communicating this with product owners. And then we are interviewing them, understanding exactly what they need to be improved. And then we're sometimes taking the lead developers with us on a meeting, or sometimes we are talking with the lead developers about the features that need to be developed, to gather their feedback on what is feasible and what's not and how much effort is needed for something to be implemented so that it can be communicated back to a client.

[20:11] PARTICIPANT 8:

Sometimes you have efforts that are covered in agreements like small improvements and to make everything work as they as it was agreed at the beginning. But sometimes they just want improvements to adjust their platform for their systems. This is like improvements that it's not agreed that it's something like change requires something additional. Then you negotiate and actually I negotiate with them and say, okay, we need two months to develop this. This is one hundred hours of development time, which costs this and this. And if they agree, then we take those requirements and put them into the backlog for planning for next month. Those requirements that are paid are always like with a bigger priority because you have to deliver something that you signed the agreement for.

[21:35] PARTICIPANT 8:

And the other requirements, just go into the backlog and get prioritized with all the rest. So once the requirement is, of course, this is for the documentation, we are writing the Confluence pages, explaining exactly what needs to be done from a user perspective, what are acceptance criteria, what needs to be covered. And this is linked to Jira tickets and epics so that

development can be organized. Then once those requirements are explained in Confluence, the development team can start analyzing and cascading those requirements into particular tasks, subtasks. And start organizing their team that is going to work on this.

[22:56] PARTICIPANT 8:

Ok. Once all this organization is finished, then they start implementation, and it is for those two-week sprints and we're making sure that the whole effort for the whole feature and how it is divided into sprints. We're making sure that we're on track so that because it's something in the chain is late, it pushes the other thing. So, it means that we're at risk to be late to go through that line. And we're resolving those conflicts as soon as they appear because it's easier to resolve them at the beginning when you have means and greater possibilities to do such thing. Under assumption that everything goes well, the feature is developed on time, its deployed to production.

[24:06] PARTICIPANT 8:

Then we are organizing demos for a client explaining and actually showing the feature that is implemented on a platform and how they can start using it. Of course, in the meantime, during the process of development, if something comes up and needs to be cleared up, we are in constant communication with those clients and we're clearing up concerns that may appear during those this development and making sure that we will build exactly what we have agreed that the beginning.

[24:55] RESEARCHER:

I've noticed something in this journey that Agile calls for the customer to be part of the cross-functional team and your customer was external, right? How were you developing the requirement in the absence of the customer?

[25:18] PARTICIPANT 8:

Well, you mean the internal requirements.

[25:22] RESEARCHER:

No, the external requirements that come from the customer, right? And the customer is not part of the cross-functional team, right?

[25:35] PARTICIPANT 8:

No.

[25:37] RESEARCHER:

How were you developing the requirement? Who was communicating the requirement to the team? Because Agile is a collaborative environment where the user story is very brief. And the requirements are communicated collaboratively.

[25:54] PARTICIPANT 8:

Well, those are product owners. They're like a glue between the development team and customers, or business or anybody who is requiring something. So, we're like translators between those two parties.

[26:17] RESEARCHER:

So, the product owner needs to have a great knowledge of the product?

[26:24] PARTICIPANT 8:

Yes, of course.

[26:25] RESEARCHER:

Yeah of course. I've noticed that in this journey, you didn't talk much about the QA. And how do you ensure quality in this journey?

[26:38] PARTICIPANT 8:

Well, as I mentioned before. Yeah, I was concentrating more on the business side.

[26:44] RESEARCHER:

Yes, thank you.

[26:47] PARTICIPANT 8:

Well, as I mentioned, that beginning QA is part of the planning team so as soon as requirements come to the backlog, the QA is giving the feedback and their opinion on the feasibility of those requirements and what could be impediments on the road of development. Of course, this is a process on paper, but we are also talking with them like when you have a call with a client. And there is something new that they're requiring. You will go talk person to person with QA because when you are long in the organization, you know who has expertise for which field and then you can just consult if something just sounds like mission impossible. You can go to a person and ask whether they think that this is something is feasible or is it too complicated, are we capable of delivering something that the client has required. And then you get first feedback and you get an idea where you stand with this requirement. And then later on, when you enter process then QA is formally giving their opinion of feasibility, effort, and some possible impediments that you can come to.

[28:59] RESEARCHER:

Great. Thank you very much. Do you think this setup that you've been describing to us is a good implementation of Scrum and why?

[29:12] PARTICIPANT 8:

Well, it's not bad, but it's not the best out there. First, something that is really wrong is those planning cycles of three months. But I did not hear of companies that are big and having small, small development cycles. It is just something that is applicable to smaller development teams. And yeah, I think that there is a big struggle. Because when you're in big companies, management is used to get the prediction when something is going to be finished. And this is in collision with Agile where you don't have deadlines, big deadlines when some feature is going to be finished. It's going to be finished when users are satisfied. You know when you don't have any negative feedback from the field, or you're decided just not payable for you to work on something. And then in these big organizations when you have big client thing, lots of money, they just want to know when they're going to have their new feature that they're required.

[30:56] PARTICIPANT 8:

And then you just have to get some part of Agile that would work for the team organization. But then you cannot avoid some elements from Waterfall. You have to keep them because of somebody who is paying you to do the work. So therefore, it depends from which point you are looking. If you're looking from the point of software development, it's not good to have three months planning period because it's impossible to predict what is going to happen in three months' time. Priorities might change. Maybe something will break, and you will need to spend time fixing it. Maybe some people will leave the company and you have to hire new ones. All these impediments, it can be easier with Agile to overcome this because you have this two week planning cycles and like a big backlog with prioritized tasks. And it would be just fluent changes. So, changes would be easily adapted. So, I think this is what is missing in this implementation of Agile. Yeah, I think this is it.

[33:01] RESEARCHER:

Ok. Thank you. I was taking notes. You already explained that so it's fine. You already talked about what you do to assure quality in this Agile setup. We already talked about that. Do you think this Agile setup you've been describing to us, produces quality software and how?

[33:38] PARTICIPANT 8:

Well, no. Because I was thinking...and no.

[34:00] RESEARCHER:

Why is that?

[34:01] PARTICIPANT 8:

Because maybe, I don't think this is about Agile. I think this is about management setting priorities. Because it's not up to a framework to let you work on quality. It's about business priorities. When you want to have quality software then you just leave the engineers time to build something that is really future-proof. Something that is not going to have bugs, but something that will last for a long time. But usually, this is not the case because it's always

about having something as soon as possible so that you can sell it. Somebody is waiting for something, money is depending on this. And it's a pressure in such a way. I don't think that any framework can solve this kind of problem. But if you understand that there are important stuff that need to be done in order not to leave technical debt. Then any framework can help you. You can always leave thirty, forty percent of time for engineers to decide what is most important for this software architecture and to leave them just to build it better and improve and fix bugs and stuff like this.

[36:19] RESEARCHER:

So, it comes to organizational issues and the people at the end. It's not the methodology or the framework itself.

[36:30] PARTICIPANT 8:

Exactly. Exactly.

[36:33] RESEARCHER:

Fantastic. That's really good. Thank you very much. Can you share with me a positive story about Agile and its ability to produce quality software?

[36:48] PARTICIPANT 8:

Well, yeah, working on a project that is defined upfront. Meaning all the requirements are defined. The definition of done, is there acceptance criteria, is there QA involved from the beginning. They are working along with the developers on creating testing plans. The communication is constant with the team. Everybody is aware of what other people are working on. And work is synchronized. So, nothing comes up as a blocker and you finish on time. You run the tests and bugs that come up are resolved before going to production. And then you deploy. So, I would say this is a very successful project.

[38:19] RESEARCHER:

Very good. How about a negative story? Doesn't go always good.

[38:27] PARTICIPANT 8:

Yeah, well, you start late. This is a very negative scenario. You start late. Meaning that the development of a feature that was supposed to start a few months ago but it didn't because it was not prioritized. Client is now nervous already. You need to calm them down. Development teams start working but they're not actually sure what to expected of them and what they need to work on first. And what is actually the whole picture and the story behind the feature they're building and there are change requests coming up all the time. And they start working on something then they stop and start working something else. QA is working simultaneously on a few projects. And then the deadline comes to this feature, you have to show something, and nothing is working, and you have to explain why nothing is working or finding something that will take you out of this really uncomfortable situation. And then you'll have software with a lot of

bugs, it's slow. It needs to be completely refactored after some short period of time. So, I would call this really bad example of a project and feature development.

[40:28] RESEARCHER:

I agree. I strongly agree. Fantastic. This brings us to the last question. What do you think of this statement, Agile produces poor quality software?

[40:44] PARTICIPANT 8:

I would disagree because it's not Agile's fault. It's up to people who are leading the team and how they will implement it Agile. You can just take fragments that can be useful, like you can have your own setup, as you'll see. And you can take retrospectives from Agile and you can just accommodate it to how you work and say, OK, I want to learn from my work. I want to implement retrospectives. This is part of Agile, I'm not going to do daily because they're all sitting in the same office. And I know perfectly what my colleagues are working on at each moment. But I'm going to take retrospectives because I want to learn, and I want to improve myself. We need iteration, and this is what I'm going to implement. So, you can think in this way and as more as you know about Agile, not just Agile but other frameworks, and there are a lot of methodologies in Agile. And you can just mix it. Nobody says that you have to do everything, as it is said. And the more you understand how something can be applied in practice and how it can be useful to your particular case, then you will be better at what you're doing. So, I wouldn't think that Agile produces poor software. I think people do.

[42:46] RESEARCHER:

Fantastic. That's what I said. You mentioned something very interesting and I want to follow up on it. You mention you become better at what you do. How do organizations and software development teams become better at it?

[43:05] PARTICIPANT 8:

I can go back to the example with QA.

[43:12] RESEARCHER:

Yes.

[43:13] PARTICIPANT 8:

You saw that QA is like separated from the process when they are delivered a piece of software to test it in the end. And then like, let's imagine this situation. You have a traditional organization, developer is working ten days. Then they give this piece of software to create, test it. And then he realizes this could be developed in some other way that is maybe more suitable to some other element of the software. But it's too late because this is already done, and developer has gone onto something else. And then you organize retrospectives and you say, OK, what do you think, what can be better. Then I say, OK, I could be involved in earlier phases because I could give you some useful information from the beginning and maybe influence the

flow of development. Then you start doing this and reflect on this decision and say, OK, how we improve the way we're working, having QA involved at the beginning. Is all OK?

[44:45] RESEARCHER:

Yes, sorry. I got a notification in my screen, I wasn't expecting it. I do have another meeting. It scares me. I'm tired.

[44:55] PARTICIPANT 8:

No, it's fine, I was just wondering. Maybe the sound was interrupted or something.

[45:00] RESEARCHER:

I got I got some notification for a meeting, which I wasn't expecting, but I do have another meeting. Keep continuing. Sorry. Sorry about that.

[45:10] PARTICIPANT 8:

No, no, no. So then you reflect and say OK, so having improved the way we're working and you can see that you did improve the way you are working because you started to do something earlier that was causing problems and now you don't have those kind of problems. Then you're going to say something else and you'll find a way how to face it and improve. So, in each iteration, you can be slightly better.

[45:47] RESEARCHER:

But if you in place those learnings, of course. Because every retrospective, there is a learning from it. And you have to put those learnings into actions and put those actions into reality. So that's when you improve, of course.

[46:14] PARTICIPANT 8:

Yes, but then those are key parts. That is how you are going to improve. You have to reflect once again and say have we improved with this change. If you haven't perfected it, then you're going to keep that way. But if not, then something else is a problem. You need another part of this.

[46:42] RESEARCHER:

Yes. You need to keep iterating. So next time you would say, well, it didn't work. We have to look at something else. It must be something else. Yeah, I agree with you. Thank you very much. That was very insightful and very interesting. Thank you very much. I enjoyed it. Do you have any questions for me before we conclude?

[47:05] PARTICIPANT 8:

No.

[47:07] RESEARCHER:

OK.

[47:12] RESEARCHER:

Yeah. Thank you for that. Just one procedural thing before we conclude, is we transcribed the interviews and we send them to the participant for validation. So, if you can have a look at the transcript and make sure everything is OK. And let me know.

[47:35] PARTICIPANT 8:

OK.

[47:35] RESEARCHER:

I will send it in a couple weeks' time.

[47:38] PARTICIPANT 8:

Okay. No problem.

[47:39] RESEARCHER:

Thank you very much, PARTICIPANT 8. I have a good night, bye.