[00:00] RESEARCHER:

I'm doing well, thank you. Thanks for your time this morning. I do really appreciate.

[00:07] Participant 19:

Thank you.

[00:09] RESEARCHER:

All right. How about if we start with introducing myself. I will introduce myself and tell you what I do, and we'll take it from there. And we'll start with the questions after that. My name is [redacted]. I do research in software quality. I'm interested in understanding how software teams achieve quality in various software processes and methodologies. Currently, I'm working on trying to understand how agile creates software quality and doing that, I'm interviewing people. The reason why we do interview people because we believe in practice and the knowledge in practice bridge the gap between theory and practice.

[01:15] RESEARCHER:

We know little about agile and quality and people should tell us how in practice they do achieve software quality in order for us to bridge that theory gap. Basically, this is what I'm doing. I'm based [redacted]. That's all as an introduction unless you have questions for me?

[01:41] Participant 19:

Not right now.

[01:44] RESEARCHER:

All right, fantastic. And what we're going to do, we're going to start with some introductions. And if you like to introduce yourself and tell us what you do and what your experience has been throughout the years.

[02:01] Participant 19:

Okay. I'm a computer science graduate by education. I have eight years of experience, almost eight years of experience in manual testing. In my career, I worked with three companies. My last two companies in which I worked used tweaked versions of Scrum. There were small scale company, so had about 50 to 100 employees. We didn't work for any client because they were product-based companies, my last two companies. And then my first company, for which I worked, it was a multi-national company. And while I was working in that company, I was associated with three projects. So out of three, two were following the waterfall model of testing. And one was pure Scrum. And the project which was following pure scrum was a Danish company. [redacted]

[03:14] RESEARCHER:

Yeah, I know of it.

[03:16] Participant 19:

While I was working with [redacted], it was in the initial stage. So, I was part of that projects. And the last two companies for which I worked, they were into product development, so we worked for clients that were coming to buy those products.

[03:47] RESEARCHER:

Yes. OK. Fantastic. Thank you. That's a very interesting and rich type of experience. That's take me to the next question. How do you define quality in agile software development?

[04:14] Participant 19:

Very broad topic, but I'll keep it simple. Software quality means first and foremost no bugs. The business doesn't like buggy software. The software should add value to the business, this usually means addressing the needs of the business. This is my perspective as a QA. A software developer may tell you software quality is more code quality and good design. That's is also important and fundamental.

[04:47] RESEARCHER:

That's take me to the next question. In your current work environments or in any work environments you worked for, can you describe to us the Scrum setup?

[05:14] Participant 19:

Here, I take the liberty of choosing one of these projects, which was not the latest. If I choose the latest project, it didn't have many components which are using Scrum. So, we'll have not much to discuss on that, so I'll use a project which I was [redacted], which came to my company for getting a CRM application. So, a client relationship management application.

[05:49] RESEARCHER:

Yes.

[04:50] Participant 19:

So, I'll start with the initiation meeting, which happened between QA (Quality assurance engineer), a developer, a business analyst, technical analyst, and the client. Part of that meeting we discussed about the vision of the client. So, the client was only discussing their vision and expectations, which they had for the application. And so, it happened during the start of the month. And then towards the end of the month, the BA came with actual requirement. So actually, the development started after one month of that initiation meeting. So, when the BA came and she started discussing, we came to know that this whole development process will be divided into three phases. First, will have six modules, second phase, will have four modules and last phase of this will be maintenance. So, I'll go through the first phase, which had six modules. Each module had almost six or seven functionalities to be done and each functionality was called one user story. So, when the BA started actual knowledge transfer to us, she used wireframes instead of a requirement document because she did not have extensive requirement documents.

[06:31] Participant 19:

So, because she was the contact person with the client, she used to discuss with clients about the functionality what they wanted. And she used to make wireframes and using those wireframes, she used to give KTs(Knowledge transfer) to us. And based on that KTs, we used to create test cases. Are you there, Researcher?

[06:58] RESEARCHER:

Yes, I'm there. I'm listening to you. I'm just taking notes.

[07:01] Participant 19:

I'll move forward then. Based on that KT, and the discussions happening in the meeting, we used to create test cases and developers used to start developing. Once we have completed the test cases, we used to give it to review to the business analyst. She used to give comments based on her understanding. And like it was inevitable that while writing test cases, we used to get more questions related to the functionality and we used to write it down inside that test case. We used to use Excel. When the BA used to review it, she also used to answer the questions which were noted down in the test cases. Once the review was completed and the test case was baseline, we started testing. And during testing, we used to report bugs and so it was a continuous process reporting bugs, getting the fixes, retesting, and completing the whole testing. So, when we used to say that testing is complete and all the test cases were executed, there are no bugs present and there are no questions pending. As soon as the exit criteria was met, we used to give demo to the client.

[08:41] Participant 19:

So, what happened was for all the use cases we used to test, we would give a demo to the client , so it was a continuous process. I was part of a team that followed Scrum to the line. So, the use cases are bound by a time box that is called sprints. The only difference was that I don't think that tweaked scrum and actual Scrum with Sprints which help the product quality. When development and testing is time-boxed, we have certain control over the quality of the way we work and the quality of the product. So, following sprints, makes us more competent and makes us know more about our shortcomings and good things also.

[09:08] RESEARCHER:

Let me interrupt! What do you mean by “having control over quality” when using Sprints?

[09:14] Participant 19:

Yes! Basically, we test more often, we get frequent feedback from the PO and we rectify much quicker and bugs do not slip to production. And even if they do we can still correct in the next Sprint. Overall the quality is always better. More importantly, we learn quicker about the actual business needs, we make less and less assumptions.

[10:18] RESEARCHER:

Great to hear this! I do have more follow up questions. Do you want to keep continue or should I ask my follow up question?

[10:24] Participant 19:

No, go ahead with the follow up question.

[10:26] RESEARCHER:

I've noticed that there is an element of waterfall in the setup of the project. Am I right or wrong? Because what you describe is the requirement has been done and the requirement being brought to the scrum team for you to start working with. This is not really scrum, is it?

[10:50] Participant 19:

So, what I know from working in waterfall is that when the requirement was given to us, that is very less chances of discussing it and improving over it. When BA used to come with wireframes and discussing the requirements, we use to ask her questions. And many times, I've seen that asking questions actually altered the requirement to be more user friendly or to be more in line with the market requirement, which is not the case with waterfall. Because during waterfall, the requirement was written almost six, seven months before and person who has actually written the requirement was probably not present. So, there was no live discussion as such.

[11:52] RESEARCHER:

So that's dynamic and that's collaboration that helps you not only learn the requirement, but also empower you to influence the requirement? Correct?

[12:04] Participant 19:

Yes.

[12:05] RESEARCHER:

So, how does this active collaboration help quality?

[12:13] Participant 19:

It does! We become better knowledgeable about the requirements and we participate in the user stories. This collaboration helps us to know what to test and we catch more bugs. To be honest, I loved working in Agile because you have your own identity. People know that you know stuff then they appreciate you. So, it's instant gratification, I would say. Because the more you work, the more you are appreciated. And because of more communication, developers also appreciate you and you also appreciate the developers. And because I have been appreciated by the developers, that I work with scenarios which are rare. It's really quick, the turnaround time is short.

[13:00] RESEARCHER:

Yeah. Go ahead.

[13:05] Participant 19:

And while working on waterfall model for two projects, I did not have any communication with the client. I was not aware of their processes or nothing. I just used to work what was given to me. Having the PO on the team helps a lot to understand the requirements and other expectations.

[13:25] RESEARCHER:

You mentioned that the project was successful. What made you think it was successful or what made the project successful?

[13:38] Participant 19:

I think the most important thing, which I noted, was that since we were giving a demo to the client on a regular basis, they were aware of the functionality. Actually, they were seeing the end product. So, when the project was released, there were no surprises for them. They were already aware of the product. They were happy to accept it. Then the project went into the third phase of maintenance. They're actually using. It was not a surprise for them.

[14:26] RESEARCHER:

So, the engagement makes them familiar with the product, earliest as possible. And also, the accountability of accepting the deliverables is early as well. They are accountable for saying yes to the feature itself and accepting it, right?

[14:50] Participant 19:

Because giving a demo many times, because they were seeing the end product, they used to give comments if it’s not in line with their functionality or not in line with what they imagined. So, we used to go back and check the requirement, what was the reason why we didn't develop according to that vision. So, if it was a mistake, then we used to develop it, if not, then we used to go for a change request. This iterative inspection process allowed us to correct, adjust and at the end deliver functionality that meet the client’s needs.

[15:28] RESEARCHER:

Fantastic. OK, I'm just going to move to the next question. What do you think of Agile in general? You already mentioned that. You love it. You like it. Why?

[15:50] Participant 19:

Agile, apart from what I said, I also think Agile makes you competent because it makes you think, in relation with what the market wants. So, you need to keep on top of things. You cannot go into a requirements discussion, thinking the requirements on the basis of ten-year back technology. You cannot do that. You need to learn really quickly to be able to participate, actually participate in the discussion. Otherwise, it's of no use. Scrum is not forgiving. I have seen that flaws are really quick to be discovered when working in Scrum. If I'm working in a team of ten people, I suppose, if somebody is low on competence, it shows really quickly. It is not forgiving.

[17:02] RESEARCHER:

What do you mean it's not forgiving?

[17:05] Participant 19:

I don't know why, but maybe it's some personal scenario. I have seen three people sacked and all these people when they were sacked in an Agile environment. Because it's really quick, if you are not modifying and if you are not learning, it just shows. For example, in an example, the BA was sacked because she was the point of contact with the client and she was not able to understand the requirement. So, it just made the whole process chain collapse because of she was the starting point. So, it was a disaster in fact, so I've seen you need to learn really quickly. When I started working in agile, I used to complain a lot about the time spent on meetings. In Agile, there are lot of discussions, talking and meetings, I used to think that it is a waste of time because I came from a waterfall methodology and in which there were not many discussions and time spent in meetings. It was like work. In Agile because the work is part in small section, more of the time is spent on talking. Also, I find Agile it doesn't have any downtime, which I missed compared to waterfall. Because when you are working in Agile and working, working continuously and sometimes even with my colleague, we have discussed that it leads to numbing of your creativity because you are constantly working. With waterfall methodology, I have spent almost six months without doing anything. I was asked to write manuals and not test because development wasn't progressing and there was nothing to test.

[19:46] RESEARCHER:

That's brings me to my next question, which is a little bit more detailed. Can you take me through the journey of a requirement or a feature or a user story from inception, its creation to the release?

[20:02] Participant 19:

So, a user story was written by a product owner. And when we used to initiate a sprint. We all used to go and discuss each user story from the point of view of developers and testers. Many times, I have experienced that a user story may be easy to develop, but not to test. I was, in fact, appreciative that QA was included in this kind of meeting, because then estimations made by you were true. So, I'll come back to the requirement. So, in Scrum, I've seen that the requirement is written with three or four lines. Written in the form of like, I want my application to this, just this much. Upon those three lines the developer, the tester and product owner discussed. If we want this application to do this, we need these many things. It was just written in pointers. The documentation was not done. It was written in pointers and all the discussions were accumulated and the document was made and using that document, development was done, and test cases were created. The test cases were now attached to that user story. The user story, meaning the requirement.

When developed and tested, that requirement was now physically present in the system. So, this is how a requirement was discussed, developed, tested, and committed to the code.

[22:34] RESEARCHER:

So, there is some level of unwritten requirements. So, there is a minimum of requirement that is communicated. But the rest of the requirement is communicated collaboratively.

[22:50] Participant 19:

Yes in meetings.

[22:51] RESEARCHER:

Yeah, in meetings.

[22:52] Participant 19:

But meetings were always documented. Yes, it was documented not a heavy document, but pointers that was discussed. And also, you can say that the requirement was extensively explained in the test case. So, the test case is also a reflection of that requirement.

[23:19] RESEARCHER:

So other than testing, what do you do to assure quality? Because testing is quality control is you make sure that the software meets the requirement. It's a fit for the requirement.

However, we do a lot of things to assure that the process itself and the deliverables are of high quality. So, what do you do other than testing to assure quality?

[23:50] Participant 19:

So, in Agile, since QA is included in all the discussions, the quality starts with the requirement discussion and QA asking questions based on the end-user perspective. It starts from there. Because the test cases are reviewed, peer to peer behavior. I take an example in which me and a colleague used to work, and she used to be in the meeting present with me for discussing the requirement on which I was supposed to write test cases. After writing test cases, the test cases were then reviewed by my colleague.

So, again, the requirements were defined. So, the requirement discussion, test cases were viewed, and the major quality enhancing process which we followed was retrospective meetings at the end of each sprint. So, as I explained earlier, that because sprints are time-boxed, and we are aware that each sprint, the criteria that was used was story points. So, we are aware that each sprint is going to develop a set number of story points. Now using that as a baseline, we know that sprint one had a hundred defects. Sprint two has fifty defects, sprint three had twenty-five defects, so it was quantifiable like you can look and see and understand, why so many defects were seen in this particular sprint ? What went wrong?

[26:08] Participant 19:

Didn't developer get the requirements, or the tester was more competent and hence the test case was more probing. So did the retrospective meeting used the test cases which help us to improve and understand continuously. Because we needed to do root cause analysis on each of the defects found. We used to discuss the defects found during the peer-to-peer review of the test cases. We were not interested in what the developers did because if you do retrospective on defects, you'll get improvements for the developer, and not the tester. So, we used to do retrospective meetings for defects found in test cases. With each defect we used to ask five questions. Why? Why? Five times why. “Five why” analysis to get insight.

[27:25] RESEARCHER:

So, what I feel from your answer, in this process or in this instance you've been describing, you've been empowered as a QA professional from the beginning of the process. And this empowerment has enabled you to be more efficient and more happy in the process. Is that correct?

[27:50] Participant 19:

Yes.

[27:52] RESEARCHER:

Fantastic. I would like to follow up on something very interesting you mentioned, which is peer to peer behavior. Can you elaborate on that?

[28:05] Participant 19:

Yes, because by mistake, I told you that she was my friend instead of my colleague, but they are my friends.

[28:15] RESEARCHER:

That's fine. We make friends at work. It's normal. It's a normal human thing.

[28:22] Participant 19:

So, I found great collaboration with those people with whom I've worked in my Scrum experience. Because in Scrum everybody is aware of what everybody's doing. Like when I was working on a user story and I talk of some functionality with this application, we used to discuss. It was so easy knowing that I can discuss with my colleague. In fact, a lot. We used to exchange ideas a lot, so peer to peer behavior is very friendly. It is dependent also.

[29:25] RESEARCHER:

It is very interesting to me because I worked in the industry for 20 years and I felt that the relationship between the QA, I worked a lot in Waterfall, but when Agile came in, I worked in Agile for like five years, and I left and came back to academia. But from my experience, I felt that the relationship between the QA and the developer, it's always has this superiority element, which is the developer is always superior in the process. They get offended when a defect is a raised. They look at the QA as a less skilled team member.

Is this something you can comment on? How is it in Agile now?

[30:33] Participant 19:

I don't think Agile eliminates this, I don't think so. I'll give the answer in a few points. First point is that because there is a high communication rate between QA and a developer in an Agile environment, the differences are not much. Because they knew me, the developers with whom I worked. They knew me, so I do not think they were offended by me logging defects because they used to test also in some form or the other. I'm probably not the right person to answer this because I have an explanation, because during my first year of experience, we were given training. It was defect-driven training. In the training, we were given different scenarios in which why a tester needs to be more diplomatic.

[32:18] RESEARCHER:

That's interesting.

[32:20] Participant 19:

We were taught to write defects in a way that were less offending. For example, you should not say, it is not correct. You should say, I found this requirement not following the discussed functionality. Like that. And moving the blame...

[32:45] RESEARCHER:

...from the developer.

[32:47] Participant 19:

Yes, exactly. So, this has never put me in a position where I have offended the developer. However, I do have a feeling that they feel superior because they are developing something. They feel like they're developing, and we are just testing. So, I always say that maybe you are developing, but I am here to make it better. And I'm here to make you focus. As a QA, we do testing, but I think QA is also about following process. Because I'm making somebody follow a process, I have more power actually. And if conditions are right, if the person is right, they do understand that yes, I'm working to create a great application. But I've also worked with developers who are very hard to work with because they just do not accept that this is a defect. But it's a person to person thing.

[34:13] RESEARCHER:

Ok. Fantastic. Let's move to the next question. We already discussed a bit of it, but what do you do to assure software quality in an Agile setup?

[34:33] Participant 19:

Because the development cycle is small so you can get feedback fast. Because development is small, the feedback is fast and therefore the improvement is fast. It goes with the process because if you don't improve, you end up making an application which is low quality. But if you learn and improve then the quality is much better; this is what I experienced.

[35:27] RESEARCHER:

So, the continuous improvement makes the team more efficient and creates better quality, right?

[35:39] Participant 19:

Yes. And this continuous improvement is just because there is continuous feedback. It is not the same with waterfall. It is the process which actually enables us, because processes helps continuous development, this feedback is continuous improvement.

I think that is the essence of why Agile is so popular. And I also think that Agile is popular because it's the demand of the current situation in the world. Because products are so complex nowadays that you cannot develop an application based on one year old requirements. It's impossible. Your product will not work in the market. In Agile, if a product is launched, I think the requirement will be all most one month old. In the waterfall model, it's like one year old requirement that you're working with

[36:46] RESEARCHER:

But processes are not everything, are they? You need the people to make them work, right?

[36:54] Participant 19:

Yes. Yes, definitely. I'm definitely say that being a QA, I was testing in waterfall also and I was testing in Agile also. There was no difference in my testing.

[37:20] RESEARCHER:

So, what's made you a better tester in Agile then?

[37:27] Participant 19:

Freedom of asking questions and influencing the requirements. I feel more in control of work than when I was working in waterfall.

[37:49] RESEARCHER:

Fantastic. Can you share with me a positive story from your experience? A positive story in the sense of software development. And if you can share it with a little bit of details, please. Thank you.

[38:15] Participant 19:

One of the projects for which I worked, for [redacted], it was in fact one of the smoothest implementation projects in history of that company. And we were appreciated a great deal during that implementation. Because normally when an application was given to the client, there were very many iterations with the client and them being not happy with the functionality. In [redacted] it was not the case. And because it was the first time this happened, there was a great deal of discussion about why this project was successful. And many people said that it was because the people involved in this project were people who took ownership of their work. And another difference in the team is that three out of seven people had worked in Scrum. So, they were aware of sprints, which actually helped I think with the seamless implementation.

[40:01] RESEARCHER:

Ok, fantastic. We can move to the next questions. Can you share with me a negative story, something went bad or didn't go well? And why?

[40:17] Participant 19:

Just after this [redacted], I was assigned to a project in which I think I mentioned this earlier, the BA, she was responsible for communicating the client's requirement to us, and she couldn't explain the requirement well and hence, I was not able to write test cases because I always found that my questions are not being answered. So, what I found was that person didn't agree with the values of Agile. She was not ready to communicate. And she always used to say, go back to the first KT document that I gave. Listen to the recordings. She was not involved. I think she didn't value it and even knew it.

[41:21] RESEARCHER:

So, she didn't embrace the values of agility?

[41:24] Participant 19:

Yes. And it broke and I left that project because I actually got scared.

[41:33] RESEARCHER:

Yes, because it was heading to failure.

[41:39] Participant 19:

Yes, and I was scared because in Agile everybody is responsible because if I was not able to write test cases, which I was not able to do, people were asking me, what is happening? And then pointing out why, and I couldn't work within that project. Do I have a good BA because that is a gap?

[42:08] RESEARCHER:

And do you think that the setup itself didn't help? Because it was a client vendor relationship.

[42:18] Participant 19:

No, because same setup gave great results from [redacted] So, it was not setup, it was a resource issue.

[42:31] RESEARCHER:

Yeah. The efficiency of the person itself and embracing the QA values. Sorry, the Agile values. I agree very much with that. Fantastic. That's brings me to the next question. OK. It is a little bit provocative question, but the purpose is to get to you to talk and to tell us your opinion. What do you think of this statement: Agile produces poor software?

[43:12] Participant 19:

Why don't I agree it?

[43:14] RESEARCHER:

Can you share with us your argument? That's what we're looking for, is the argument. Yes.

[43:21] Participant 19:

First of all, I think that Agile produces an application which is in line with the latest requirement, which waterfall cannot do, and an older process cannot do. And also, because most of the development and testing is done by humans, so the mentality of human while working on a project also affects the quality. And in Agile I've seen that if everybody is onboard with the principles and values of Agile methodology, it is seamless, and it is empowering and hence you work to your best efficiency. It produces great results. But of course, at the end of the day, a product quality depends upon the people who have developed it and people who have tested it. No more than the process. But of course, it is a combination of process and people. You cannot say because we have used the latest process like Agile and Scrum, therefore, we are destined to be successful. That's not the case. But if the same team is working with the waterfall model and the same team is working on an Agile model, I think at the end, Agile will give you an application, which is more in line with the current market standards.

[45:17] Participant 19:

This is what I have seen, because when I was working with waterfall model, both of those projects were R&D type. So, it was research and development and they were developing projects for future. I don't think that both of those project’s applications are in the market yet. They still working on it. Because the requirement which was written for a waterfall model...are you aware of those applications which stores lacks and lacks of requirements for which you have a scientist who writes it and can visualize twenty years forward functionality.

[45:57] RESEARCHER:

I know those horrible applications. I worked with it.

[46:05] Participant 19:

I think Agile is fast. It produces an application, which is trendy.

[46:18] RESEARCHER:

Trendy, yeah. I don't have further questions. I think the end was well said. I liked it. Do you have any questions for me before we conclude?

[46:43] Participant 19:

Do academia, like people in research and development also think that QA is really needed? Because I have been with managers who are actually managing their development team, but they were forced to have QA and I have worked with manager who clearly say they don't need testers. Do you agree with that?

[47:26] RESEARCHER:

No. Of course. We highly believe in quality. And I do strongly believe personally that quality should be engaged from the beginning of the process. Like you said yourself and the testers should be having an equal position to the developer and the team. And we should empower our QA and testers to work in any process. We should empower them to be equal to every member of the team. We should empower them to have and voice their opinion. I've seen QA, they are more knowledgeable in my experience, and better knowledgeable than the business analysts.

I've seen QA teaching the business analyst the product. I used QA myself because I worked as a business analyst. I used QA as my product owner because they know the product better than the business sometimes. No, not at all. I think the QA function is underestimated in software development. And, if agile empowers you to do your job, then it is the most suitable method for software development. But I think it should happen in any method or in any process you using to empower the testers.

[49:26] RESEARCHER

Quality should be supreme. I've been studying in the last three years, part of my PhD open source software development and their attitude to work quality is very different. They see quality almost like a religion. It's about quality and what they do to achieve quality is far, far ahead from most closed source companies. So, their quality assurance practices are far more developed than closed source development. So, no, I don't agree with that statement and I will strongly defend against it.

[50:22] Participant 19:

I was not knowing that.

[50:25] RESEARCHER:

Do you have any more questions?

[50:28] Participant 19:

No, I'm good.

[50:29] RESEARCHER:

What we do is we transcribed the interview. And if you don't mind, I will send you the transcription and if you can have a quick look that we didn't misquote you and everything is OK. What you have said is what we going to use, and your participation is completely anonymous, and the names of the companies you have used during the interview are also anonymous. So, you don't have to worry about that. They will be in the script, but they won't be put publicly, or we wouldn't say anything about them publicly. So I will send you the script and if you can have a look at it and just make sure you are comfortable with what you've said and if you want to add something, feel free to send it to me by email and I'll add it to the script.