[04:18] Participant 24:

Hi. Good morning.

[04:20] RESEARCHER:

Good morning. How are you? Finally.

[04:23] Participant 24:

Yes. I'm good, how are you?

[04:26] RESEARCHER:

I'm very good, thank you. Thanks for your time and thanks for accepting to do the interview. I start with introducing myself. Just to tell you who I am and what I'm doing and why this interview. And then we'll start with the interview. How does that sound?

[04:50] Participant 24:

Yes, sure.

[04:51] RESEARCHER:

Ok fantastic. So, my name is [Deleted to preserve the participant anonymity]. I'm doing my research in software quality. I try to understand how teams achieve quality in various software development methods and processes. I'm conducting this research on Agile and how does Agile achieve quality in software development. We do interviews because we do believe with interviews we get to understand the practice, how do people practice Agile and by understanding the practice, basically we bridge the gap between theory, which can sometimes be very abstract and reality, which is the practice - what people actually do in real life. So, that's why I'm doing these interviews. Your participation is anonymous, so your name and the companies you work for and the companies and examples you share with us will remain anonymous during the research process. So, you don't have to worry about mentioning names and your name being mentioned somewhere else. So, that's who I am and what I'm doing. If you don't have questions for me, we can proceed with the interview.

[06:33] Participant 24:

Yes, sounds good. So, I understand that you're doing some research on Agile and how QA is part of Agile. My experience is mainly Scrum. So, I will be talking about my Scrum experience.

[06:48] RESEARCHER:

Correct. We interested specifically in Scrum.

[06:50] Participant 24:

Yes, so. Because QA in Agile is not I would so in one sense, not something which is separate from the team, it's part of the development life cycle. So, we can talk about it later. I think that should be enough.

[07:23] RESEARCHER:

Let's start with some warmup questions and perhaps introductions are good. How about you introduce yourself and tell us what you do and talk to us a little bit about your experience.

[07:44] Participant 24:

So, I am currently working in Sweden and I am originally from India. I have been working in software for the last eleven years. And my main skill set it is quality assurance, testing, slowly we are moving towards DevOps. And throughout this eleven years, I have been mainly, if I highlight, I've been part of three main domains of software or you can say industries, such as insurance, banking software and automotive software. So, and the banking also includes mobile apps. For this I have traveled across three continents, I have been to South Africa, I've been to the Netherlands and right now I'm in Sweden for the last two years. Throughout this eleven years, because when I started my career in 2009, the software I.T industry has already moved towards Agile. I would say I'm lucky that I'm started with Agile. So, of our projects were Agile. I have seen, and because of course, in 2008, Agile was still growing up in the industry. I have seen Agile grow from its initial stage to now where it is, fully-implemented Agile and another model of DevOps. Yes, that's a bit about myself. Do you have any questions?

[08:28] RESEARCHER:

First question, how do you define quality in the context of agile software development?

[08:34] Participant 24:

You will get different answers depending who you ask. I’m a Test Lead. My perspective is the outcome and the process. Our role in the software team is to ensure the software is free of bugs and meets the customer satisfaction and needs. The process quality is also important, because without a process there is no software. To be honest with you it is difficult to put into words what we mean by process quality. It is usually something we know it once we see it, same apply for code quality. From my experience, the best way to achieve process quality it continuous improvement. In agile, we have ongoing reviews and retros to evaluate our process. We learn and improve all the time. If you ask a developer. He may tell you the code quality first. But, if you don’t have a good process and practices in place like guidelines, peer reviews and definition of done then from my experience developers do not produce good code. But quality is a team objective; it’s the people who create quality.

[09:58] RESEARCHER:

Yes. I will move to the next question about your opinion. You started already talking about that. What do you think of Agile in general? What is your position about it?

[10:14] Participant 24:

Agile was implemented for various reasons, where you will the advantages, where mainly cited. That's my opinion of course. To counter the issues the waterfall model had. The waterfall model of software development, where you analyze, you code, then you test, then you deploy. So, this used to take around three months of cycle. Agile came up, which is rightly, what it introduced is business analysts, or analysts of the requirements, coding, testing, and deployment all should happen in one sprint and which should have a working software. A working piece of code, or working piece of component at the end of the sprint. This was primarily the main advantage that Agile was supposed to bring to the I.T software industry. So, yes Agile has minimized loss of time and results are earlier in the life cycle of the software. I agree, yes it has an advantage. This is also again my opinion, the practical opinion, I have seen that has also minimized use of resources. Human resources. That is, in the waterfall model there was hierarchical structure of the dev team. There was a development manager, there were developers under him, then you had test managers, then you had test leads under him, and then test team members. If you see this test lead, test manager, there were a lot of policemen in the waterfall model, which were unnecessary. At this moment, being in Agile, it sounds very unnecessary. Because you had this hierarchical structure in software development. Agile has replaced it with just a Scrum master and team members. Of course, product owner is always part of the team but the main team members in that sense is scrum master, which is one position, product owner and team members. These team members can be developers, coders, or BAs. Those these are three positions in Agile. So, in that sense I will say, it has reduced the resources, made it more efficient. That's my take in terms of advantage.

[14:06] RESEARCHER:

I have two questions. One is the efficiency of resources. What happened to the set of skills for example, a software architect is a set of skills that a scrum master may not have. A business analyst is another set of skills that the product owner might not have. So, when we minimize the human resources in the process, what happened to those set of skills?

[14:41] Participant 24:

That's a very important question and to be honest with you, there can be many answers, many debates on this but I'm seeing this every day in my work. You are absolutely correct, the business analyst position, a PO might not have. Due to this Agile being completely enforced in software in every medium scale or bigger scaled organization. What is happening is this multiple role playing is taking over. So, explain in a different way. I am a tester. I'm much more than a tester. I look into code, I fix the code, I take it into the production environment, which when I started, we had a dedicated person who used to deploy and do the migration. There was a dedicated automation tester, there was a dedicated coder, etc. Today, I almost do a lot more to be honest. Same thing, I see my PO, he tries to create requirements, he tries to detail this thing, which actually it is a dedicated business analyst job. At times it creates confusion. We keep raising questions to PO, what is this requirement? To be honest it has increased far more questioning and it has increased a certain amount of confusion in the requirements.

[16:53] RESEARCHER:

My next question is that you mentioned that in Agile that you do requirements analysis, development and testing at the same time. Now in software engineering, the requirements analysis and development they are dependent on each other. And testing is dependent on development, so you wouldn’t for example, start developing until you had an understanding of the requirements. You wouldn’t start testing until you had some code to start testing and the code has reached a level of maturity when you start testing. How do you manage these dependencies in Agile?

[17:41] Participant 24:

So, let me draw a parallel from my projects. So, in Agile, every requirements is discussed, unlike earlier, with projects that weren't Agile. In all starts with either PI planning or sprint planning. So, PI planning, I will talk about it later. Let's talk about sprint planning which is more relevant. A sprint in Agile is either two weeks, maximum four weeks. But I have more projects with three weeks sprint. Beginning of the sprint, the first day of the sprint, every Agile team plans for a sprint wherein some requirements flow in from the backlog and some requirements are also new, but requirements which can be done in three weeks after confidence mode by the team members. So, talking about requirements analysts, understanding how the requirements fit. Before the sprint planning, it is the duty of each team member to spend one hour before the sprint planning that is the last day of the previous sprint, wherein they spend one hour. Suppose we have six team members, five developers and me, one tester. We spend one hour and go through these requirements quickly. Like ok, I understand, I understand that. So, from a sprint perspective, these things can be tested, this cannot be tested, this has an impediment reject. In the sprint planning we discuss it and if there is further analysis required, they go to Sprint retrospection.

[20:30] Participant 24:

Another term is called Refinement. Refinement of the plan. There's an extra session around we didn't understand this. Fine we go to Refinement and spend another one hour and refine the requirement, so everybody has the same understanding. All the six team members, the product owner, and the scrum master, so everyone is on the same page, but I will say yes, this is how requirement analysis, but in that short period of one hour, one hour, one hour. It is not very...still requirements in a very particular sense, I would it is not clear. But what we have been doing is because of course we have delivery after three weeks, or two weeks, what we have seen in the way of working is, you start doing the implementation. You start doing the coding and the tester starts writing some kind of automation script because mostly in Agile, it's automated. So, I start creating test scripts with Java either using tools like Selenium etc. and that's how it happens. So that's requirement analysis which is done in a shorter, brief time and then we move to coding phase where we start writing code and then we start with some kind of scripts and then at the end of the sprint, whatever we have developed, we do a demo. Yes, there is still times where might not have achieved what we wanted. Most of the time we do achieve it because product owners and scrum master are involved every day where we present our progress in the scrum meetings in the morning. So, we all know and are on the same page, so most of the time we do achieve the small tasks we have kept for the sprint.

[22:59] Participant 24:

So, the conclusion, we do work in a very dynamic environment and everything is done in less time compared to the waterfall model where we used to spend two or three days in requirements analysis then two weeks in development and one week in testing. Compared to that everything happens together in a fully-integrated Agile team, which I've part right now. And there is slight risk involved because requirements are sometimes not defined properly because of lack of time, so we have a little bit risk.

[23:56] RESEARCHER:

So, how does this dynamic environment as you describe it help quality?

[24:02] Participant 24:

Yes, very much. With a fully integrated Agile team, we learn from each other’s about for example how to resolve defects. We learn from the PO what the accurate requirements. This shared knowledge help us develop better product.

[24:08] RESEARCHER:

Does this pressure have any negative impact?

[24:14] Participant 24:

This depends on where you are working. So, that's why I started listing the industries. It depends on whether your company is medium scale, small scale, or large scale.

Also, it depends on what are the hours lost, everything is related. For example, in India, work hours are nine hours and sometimes it goes up to ten hours. In America, it's similar. But in Sweden, it's eight hours and Netherlands also eight hours. If you say pressure, yes there might be pressure, but still you are working eight hours but coming to industry, large scale, medium scale, and small scale. In a large scale company, there might be actually be two teams, two different components. Right now, I’m in a medium scale company and I can clearly see a difference from the large scale bank in Switzerland. It was a large scale bank. We had two teams with six team members and today I'm in a medium scale automotive company which has six team members doing double the work, but we achieve it in eight hours with minimum breaks. My take is the pressure depends on which industry type you are in and also what is the labor loss.

[26:25] RESEARCHER:

We will move to some detailed questions. Can you describe the Scrum environment in your work? You could use an example from your current job or your previous job. Can you describe it for me?

[26:42] Participant 24:

Current Scrum environment?

[26:45] RESEARCHER:

Yes. Can you describe your Scrum implementation?

[26:49] Participant 24:

Yes OK. Let's start from PI planning that happens every three months. It's a bigger form of sprint planning. We start with the three months plan. This three months plan is a bigger form of sprint planning. So, what we do, in three months, what can we achieve. All the Agile teams in the company sit for two days in three months, in this particular company I'm working in. We gather information and do planning to understand what we want to do in the next three months, what is achievable and what are the bigger stories that we can breakdown, you could say what are the bigger requirements that we can breakdown into different stories. We put it in the board. In January we achieve this, in February we achieve this, in March we achieve this. So, this way we break it down. As a team we put an estimate. In January, this story might take five days. In February, for the story, it might take ten days and something like that.

[28:51] Participant 24:

We break down the stories further into small tasks during those two days of planning. So that's called PI planning of three months, for two days. And then starts the sprint. So, as I said, every Agile team in this company is of, the way of working as I say, part of Agile is a lot of other things as well. You have DevOps today, most companies are preferring DevOps. So, we have this Agile team that uses the DevOps model and the way of working includes, one PO, one Scrum master and the team member, how is a developer or tester. SO, in this team, that's the team setup. DevOps means you start coding and whatever you coded, you start implementing it, testing scripts and all this has to be checked into the service which should go to production. That's the goal of DevOps that you have a working piece of software which is in a production build. So that is also how this Agile team works.

[30:36} Participant 24:

So, that is in terms of the Agile and then if I talk about different Agile teams, so our team develops a piece of requirement and then we deploy it in an environment for the team in which we integrate our component to somebody else's component, which is another Agile team. They use our code to integrate with their system and then they deploy it in production. Sometimes we also do the deployment to production. So, is that what you wanted to know or is there anything else?

[31:28] RESEARCHER:

Yeah, I do have follow up questions. Let's start with my first follow up question. There are dependencies in this process, for example, imagine another developer and you are the tester and there is another guy, let's call him John. He is the product owner, so we started with the requirements, we have familiarity with the requirements and the developer starts developing. What does the tester do when the developer is developing? So there is a capacity usage here, which is not...because of the dependency on the developer to finish his or her code, what does the tester do and what does the product owner do when the requirements are clean and understood and the developer is developing? How are the capacities of the team members managed across this process?

[32:41] Participant 24:

The ideal place where I worked and I also I will give an example of Scrum where I'm working right now. In the [Deleted to preserve the participant anonymity] So here we had four developers and two testers, and we used to sit side-by-side. Every day the developer is working on one particular part of the code or the smallest part of the requirement. When he is coding this small requirement, then we two testers, we had two jobs to do. One job is, we have the tool from XP. Elm.

[33:55] RESEARCHER:

Yes, I know of it.

[33:56] Participant 24:

This tool is to track the test cases and if you find any bug, you track that. So, we used to create the manual test cases in QC when they are coding. Because ideally you as a team member, it is your responsibility to raise questions in sprint planning, as a tester or developer. If you are not clear about the requirement or you have the slightest doubt as a tester, you can't start writing test cases. You are supposed to raise it by raising a flag saying you don't understand this requirement. There is something called scrum [inaudible] you are supposed to raise that this requirement looks very big to me. And then you discuss, that is the ideal Agile but to be honest, I have worked in this industry for long and I can say that at one point in sprint planning, if you have a little bit of doubt, you can raise questions once, like OK, I didn’t understand then the team discusses. You can raise it twice but third time, nobody raises it because by this time, you are supposed to be clear. So basically, the sprint has started. Everybody has agreed that this we can achieve.

[35:27] Participant 24:

That means the tester understands the requirement. And then, I'm supposed to write those test cases in QC when the developers are coding and also it was our job to update the Selenium test scripts so we can understand, OK there will be a button which is going to be added to this requirement now. So, let's write a script for the button. So as a tester this was our job when the developer is coding. Also sometimes there were minute requests from the developer saying OK, I've written a small part of the code and I'm checking it in and you can pull the code out and can you quickly do a test because I want you to have a look. That's how Agile teams work. Earlier development was completed in a different environment, testing was different. Now the developer, he can ask can you quickly do a test. So, it’s not like the developer is going on and left alone. Every day, morning, there is a Scrum call and we are supposed to raise questions there and then, every day on the sprint. So, there are two weeks sprints. The product owner, developer, tester, everybody is in the standup meeting. So, developer says, yesterday I developed this button and the tester is supposed to say if he has any questions then and there when it’s his turn to give updates on Scrum. OK, you said you were working on this button, this link, did you check this link has all the necessary inputs? Have you seen the requirement of this page is the color should be red and not white, as usual? Even if the tester or the product owner is having less work, your job is to raise questions. That I will say is part of every Agile team irrespective of which place you're at or which domain you are. You are supposed to raise questions. So, to answer your question, that would be my answer.

[38:16] RESEARCHER:

I picked up two things that I would like to follow up to make sure my understanding is accurate. The first thing is the high level of collaboration in this team. So, these teams collaborate highly to minimize the dependencies across these deliverables or these artifacts, requirements analysis, development, testing etc. And the product knowledge, from what I've heard, needs to be deep. All team members need to have a very thorough and in-depth product knowledge. Is my understanding correct?

[39:04] Participant 24:

Absolutely correct. Across all industries. Yes, it increases a lot of collaboration. Earlier developers and testers had different teams not interact at all, now we interact every hour, every minute actually. We are together in a small team. We sit together and there's a high level collaboration giving input, input exchange. Also, the product knowledge exchange, these high delivering teams with Scrum in ideal conditions, we deliver every two weeks. So, product knowledge...even if you don't have knowledge about this product, you should be ready to learn quickly and adapt quickly. That's very important. I would even say, when I take an interview, they always asked me, check the human side as well. How ready is he to contribute? Yes, your understanding is completely correct.

[40:17] RESEARCHER:

Before I move to the next question, I would like you to elaborate a little further in two things you mentioned. The ideal conditions which is a very interesting concept. Can you discuss what are the ideal conditions for an Agile environment?

[40:36] Participant 24:

OK. What I mean by ideal conditions whatever is written... in Google and Agile this is all mentioned. I don't always see what we are doing and facing every day. That is ideal so to explain that further. What is ideal? Ideal would be a product owner having complete knowledge of what he is asking us to deliver. And to be honest, the requirements come so quickly in Agile and the market conditions are so high. Sometimes the product owner was asking us to deliver something that he does not have an understanding of what needs to be delivered in two weeks. From testing our quality perspective, I would say but do you know that this doesn’t work? Then he would say, oh really, okay let me check with the customer. Let me check with the person that gave me the requirement. So, ideal condition would be the product owner having an understanding. He may not have a technical understanding, but he should have a functional understanding of the requirement at least what is in the software. So, what I mean the practical thing is the product owner will neither have technical understanding or functional understanding. He will have some kind of idea that this has to be done in two weeks. He's the product owner, kind of a management, he pushes some requirement at some point. I'm not saying every sprint has this but some of the sprints yes. So, from the product owner perspective the ideal that he would have an understanding of what needs to be done, which is not always the case in a practical situation.

[42:56] Participant 24:

Scrum master. This scrum master role is very defined in Google and stuff but what is in practical, if you ask me, scrum master sometimes again, has either no technical knowledge or no understanding of the requirement. In my team six months ago, a scrum master joined, and she has never worked with a tester. She has never worked in a team that had a tester and this is a high performing team. So, ideal conditions for a scrum master will be to understand what each team member does, and the scrum master should have a deep understanding of Jira. The tool called Jira.

[43:55] RESEARCHER:

Yes, I worked with [Deleted to preserve the participant anonymity]

[44:00] Participant 24:

So, this Atlassian and Jira, these are important tools for Agile. Every day, morning, we use this tool to see what the developers are doing. Sometimes if I don't understand what this developer does, I can see in Jira, okay this is what he's done, and this is his progress. I can go to Jira and pick up a ticket and see okay this is his progress. So, the ideal environment for a scrum master would be, he or she should understand the tool, Jira completely. So, before you become a scrum master in Agile, at least go and check Jira completely in and out so you can help the team whenever they need it. There are many things, for example, in a story there are sub tasks. I expect the scrum master to know whether a story cannot be closed or moved to done in Jira unless all the sub tasks are closed. So, if you don't work with Jira, you don’t know. So that's the ideal environment for a scrum master. For a developer, an ideal environment, as you mentioned, it's a great understanding of the requirement. In an Agile team, all the developers understand Java and Angular, that creates a good team but how many teams in the world has five developers who know everything. It is not possible or also, I don't expect that. Out of five developers, there might be one person who is a dev architect who has better knowledge. Spending four years in an Agile team compared to four months in an Agile team, you are lagging behind, you have catching up to do. This also applies to a tester. So, there is some kind of knowledge curve which needs to be in an Agile team. It cannot be ideal that everybody understands and knows everything. That is my take.

[46:36] RESEARCHER:

That brings me to the next question. We already talked a lot about testing, but quality is more than testing. So, what do you to assure software quality in an Agile setup?

[46:53] Participant 24:

First thing is, with two weeks of sprint, I'll start going through the requirements completely. I have to understand a bit of architecture and I sometimes reference to the previous code, piece of code. That is an important point which I will say I've adapted over the years. As a tester in the Agile team, I have to change myself and move into programming as well and understand it. In a good delivering Agile team, it cannot happen that you are a tester who does not understand code. If you see every requirement these days for a tester, do you know Java? That is increasing day-by-day. So, this is the requirement, this is part of the change it is pointing to, I think I have an understanding so let me go check how it is done. I go to the code and this developer has worked on this before. So, I know that if something breaks by this developer, I know if I have to check the background. My requirement gathering and knowledge of this new requirement, I'm going through the code as my job. The second thing is, either I write the automation scripts myself or I help one of the developers with the test scenarios. These are the scenarios which I think are important for this requirement which we will have to test to make sure this requirement is fulfilled, what is expected and also the quality standards, not security because security is another part. But yes mostly, quality.

[49:40] Participant 24:

For example, this website we are implementing something called login functionality. Check what is the password combination. How many types of passwords are accepted, what is not accepted? Check the password compliance. Like NIST, the American password compliance. So, I have to make sure the developer understands this, and he has this as part of testing. I can do the testing. It's also expected the developer does the testing as well. That's the beauty of Agile advantages. The developer should know how to test, and the testers should know how the code works. So, one is understanding of code, second is automation and third is, creating functional suites as well. I do write some functional test cases and make sure every other team member can refer to it. When he deploys it and done with his work, if I cannot test it, because I am one tester, there are five developers and there are five things getting developed. It's not possible for one tester to test everything. It's important that everybody understands the functional test cases that I have written, and they can refer to it and test it themselves. So, that's about testing. What else do I do to ensure quality, that's one example that I already gave. For quality, I make sure that suppose we are creating new password for a requirement, I have seen the developer does not care what this password standard is. Sometimes he does not care because he has a lot of other coding to be done or some other requirement needs to be done. My job is to make sure this password creation is meeting the standards. That is company standards, like sixteen character password including uppercase, lowercase, and different characters. I have to agree to the NIST American standards of password. So, I ensure these things to increase quality.

[52:30] RESEARCHER:

OK, the last question is a little bit provocative but we just trying to get your opinion so don't be offended by it. It's a little bit provocative. So, what do you think of this statement: Agile methodologies creates poor software?

[52:54] Participant 24:

No, I completely don't agree with this.

[52:59] RESEARCHER:

OK give me your argument.

[53:08] Participant 24:

I think arguments are very riskier. As I said, whatever I said in the last fifty minutes about what the struggles of a PO, what are the struggles of a scrum master, what are the struggles of a developer or coder, this entire struggle is making each and every individual responsible for his job every day. Even if it is only an eight hour job, he is responsible for his actions today. Every morning, every person including the PO has to say what he has done yesterday. It has increased a level of accountability. Today whether you are working remotely, wherever you work, and you are part of an Agile team, you are connecting to this Scrum meeting every day. I would say if you don’t have to give any update as part of an Agile team member, then it’s not an Agile project. Then that's something else, maybe that's waterfall or some other model that would have failed. That is not an Agile team. A good Agile team cannot fail, so that's one argument. It cannot fail because you are accountable for your actions. Second thing, there are a lot of challenges in every sprint planning. Every sprint refinement, your knowledge is increasing. Of course, we understand the developer or tester's ability, of course we can see if he's lagging behind a four year developer or tester, we can't have the same understanding of the product from the first day or four months from the beginning.

[55:25] Participant 24:

One goes through a lot of infrastructural changes understanding domain knowledge etc. So of course, we understand everything that he's gaining some knowledge in this Agile dynamic environment. Third point, which you already covered, which is very practical, collaboration and trust me, nobody is left behind. when I started, I only saw my test leads...when I started out it was in an Agile environment, but not a true Agile. It was still moving from waterfall to Agile. I've seen the transition in front of my eyes. I've seen the test leads going to the meetings, the requirement analysis meetings, I've seen the development architect going to the meetings. But the developers who actually work under him and me, we had no connection with management. Today, every sprint planning, sprint ending or PI planning, we meet with the management. We set an understanding with the management and then a good Agile program cannot fail. That's my take of this. Nobody is left behind, everybody feels he is part of the team. Everybody feels it and if he doesn’t feel it, then it’s not a good Agile team.

[57:06] RESEARCHER:

Thank you very much. I really enjoyed it. It was really a great conversation. Thank you very much. Do you have any questions for me before we conclude?

[57:19] Participant 24:

I just wanted to switch on my video before we conclude.