[00:21] RESEARCHER:
Hi, PARTICIPANT 27. How are you?

[00:28] RESEARCHER:
Hello? Can you hear me?

[00:34] PARTICIPANT 27:
Can you hear me?

[00:35] RESEARCHER:
Yes, I can hear you. Hi, PARTICIPANT 27, how are you?

[00:38] PARTICIPANT 27:
Good, how you doing?

[00:39] RESEARCHER:
Good, yourself?

[00:40] PARTICIPANT 27:
Good. You're in [redacted]?

[00:42] RESEARCHER:
Yes, I'm in [redacted].

[00:44] PARTICIPANT 27:
Awesome. I always wanted to go there.

[00:48] RESEARCHER:
Don't go in winter. Don't come in winter.

[00:52] PARTICIPANT 27:
It is a very snowy winter.

[00:55] RESEARCHER:

No, it just cold and gloomy. And the weather is really depressing.

[01:09] PARTICIPANT 27:

Got it. OK.

[01:11] RESEARCHER:

Come in summer. It's very nice. It's the opposite. Do you have any questions for me before we start the interview?

[01:19] PARTICIPANT 27:

No. Let's go ahead and start the interview.

[01:21] RESEARCHER:

All right. Fantastic. How about if we start with some introduction? Can you introduce yourself and talk a little bit about your experience?

[01:33] PARTICIPANT 27:

Sure. My name is [redacted], I'm in Washington, [redacted], in the USA, I started out as a draft technology major back in '91 and actually started building websites. That was primarily how I got my start into the tech industry. From there, I realized that there was more work than I can actually handle myself. So, I actually employed developers abroad and within the states and I had them work on projects. So, I was actually running projects as a project manager, technical project manager. And then also as a Scrum master. I've been involved in Scrum teams where I've been the developer on one side, and then I've also run teams as a Scrum master. So, I kind of have both sides of the flavor, a little bit. So, I'm kind of a little bit well-rounded, a little bit different than typical project managers that come in.

[02:34] RESEARCHER:

Yeah, these project managers comes from without a background of I.T. at all.

[02:40] PARTICIPANT 27:

Is that you or me?

[02:42] RESEARCHER:

No, I'm talking in general these days.

[02:46] PARTICIPANT 27:

Well, typically, project managers are in all kinds of facets of industry now, including construction, finance. But the one that Scrum works really well with is software.

[03:03] RESEARCHER:

Let's start with a definition question, how do you define quality in Agile software development?

[03:10] PARTICIPANT 27:

The word quality can be defined in many different ways. Most of the time the word is used to describe product quality, which often depends on criteria such as user satisfaction, functionality, and free of defects. However, Agile Software Development takes the definition in another light. When speaking of software quality, it takes into account not only the delivery of the functional requirements. It has to ensure that the source code fits within the software architecture at the unit level as well as the system level. This means that the developers have to produce code that is clean and an architecture that accommodate future needs. We also need a process that assures quality. Quality is required not only in the product, but also in processes. Agile is keen in improving the process. We believe is fundamental to achieving quality.

[03:13] RESEARCHER:

What do you think of Agile?

[03:20] PARTICIPANT 27:

I think Agile is majorly cool. I think it's very effective with software development. I don't think it works for everything, but I do believe it works for software development.

[03:25] RESEARCHER:

So, what make it works for software development?

[03:28] PARTICIPANT 27:

Well, here's the deal. Software development, when you get a scope of work, if you don't have your processes or your development of your software in an Agile environment, what happens is that you have it in a Waterfall environment. The Waterfall environment actually requires you to have everything locked in 100 percent otherwise. Then what happens is, is that by the time you get to the end of the project, you have something that's useless because there's been so many more exploration. There's been discovery built into the into the actual development of the software. And then also there's been a lot of I don't know what the right word is, there's been a lot of changes that have occurred. A lot of decisions that have been made that probably could have been a poor decision. So, with a Scrum environment or with the Agile environment, I keep referencing Scrum if that's okay.

[04:31] RESEARCHER:

Yes. Actually, the study is more specific on Scrum. Yeah, which is good.

[04:36] PARTICIPANT 27:

What's great about Scrum is Scrum builds a completely software in two weeks. So, the sprint is two weeks sprints, two to four weeks sprints. And the thing about Scrum is that Scrum actually develops a... How do I explain this, Scrum actually develops a system for checks and balances. So, when the sprint is done, you have a sprint review and the spirit review is with the stakeholder or the product owner or the people that have a vested interest into the software but don't want to get into the nuts and bolts. And they start building and coming up with different ideas, coming up with different stories. And so, everything's built off user stories. So, user stories is essentially when you get a project, user stories is kind of like, let me use this example. I want to log in screen. I want to be authenticated against who I am. It might have a little more detail, like it's going to have to have an e-mail address and a password. It's going to need to have a phone authentication with text messages. So, this is used as an example. So that's a user story, right. So, when we get into building the software, we start building this log in screen. Then meet the stakeholder and they go I really want to authenticate with Google. I to authenticate with Facebook. Well, that's the whole new requirement. So, in Waterfall environment, that is a change order. See what I'm saying?

[06:24] RESEARCHER:

You have to raise a change order and go through the whole processes, etc.

[06:32] PARTICIPANT 27:

Yeah. And so, in Scrum, it is not a change order, it actually becomes a new user story. So, the story goes back into the backlog. It gets reworked and it gets re-added back into the piece of software. So, what's nice about it is that in a Scrum environment is that by the end of a sprint, you should have a nice workable piece of software that you could use. That's what's nice about it. So, when you do the actual demo for the stakeholder or product owner and they give you feedback, the whole team is there with you. Not just you, not just you as a Scrum master, but the whole team's there. We're all kind of looking at everything together and we do the demo, and if there's something that the client doesn't like or the stakeholder does like, then it's not just myself, but the whole team hears from the stakeholder, what they like, what they don't like.

[07:33] PARTICIPANT 27:

And then with the next sprint, with the next sprint planning, we could incorporate those scenes into the next sprint. So, to me, it's a better way to build software because it kind of does checks and balances and then actually gets rid of this whole idea of change orders in regard to building a software product. So, to me, I think it's a better way to build software because nobody is going to have everything at a hundred percent with a scope of work. And then it just gets kind of messy when you have the scope working, you're kind of like, you are working on this data and then the client has a hair-brained idea or reads an article or something. And then suddenly, it's kind of like, hey, I want to have this incorporated into my software. Well, yeah, you can have that but it's going to cost you money. It going to cost you time. So, this kind of helps prevent that.

[08:26] RESEARCHER:

So, you made references twice to check and balances. Can you elaborate a little bit on detail on that?

[08:35] PARTICIPANT 27:

Yes. So, what happens is, is that when you're in a sprint, you have the sprint planning which essentially takes the backlog from a previous sprint or from the requirements. And you refine the sprint backlog and you take sprint backlog and you go to the team. So, the thing about Scrum that I really like is, it gives a lot of empowerment to the team. The team itself is the one that does all the work. The team itself is the one that's in the weeds. The team itself is the one that actually kind of has a great idea of the time requirements to get something done. So, to me, it makes sense that the team should be able to make those kind of decisions. In the old project management environment, in the good old days, a project manager is responsible for the whole project in a Scrum environment it is definitely a team mentality.

[09:35] PARTICIPANT 27:

And the team is the one that helps build the philosophy for the team. So, what happens is when I say checks and balances is that you have a lot of different eyes looking at the requirements, first off. Second thing is that because everybody was at the final sprint review and they're all hearing from stakeholder what the issues are, what the problems are and how to fix them or what decisions need be made. When it gets refined back into the product backlog. I'm trying to think of how I can explain it better. It's kind of like when you build something, like let's use Lego for example. You're familiar with Lego?

[10:19] RESEARCHER:

Yes.

[10:20] PARTICIPANT 27:

OK. You're building Lego and you look at the instructions and you build it the first time. And then your kid comes around and breaks it and cries is, and says hey, you know what I want? I want it fixed back up. It's faster to put it together the second time. You know that old saying. To me, that's what Scrum does. Scrum helps you refine the software so that all the little bugs, all the little quirks, all the little kinks get worked out in the process. And, through many eyes looking at it and many people taking ownership and responsibility for it, I think that this is a very good set of checks and balances. It doesn't rely on one person, the project manager, to keep track of everything. It allows the whole team, it has everybody to take responsibility and have accountability for the overall project.

[11:12] RESEARCHER:

So, I'm just taking notes. You touched on three values of Agile or Scrum. Transparency, empowerment, and accountability. Do you think that these values help the team to achieve quality?

[11:36] PARTICIPANT 27:

I believe they do. I believe that team members, once they have a kind of buy-in. So, here's the deal, is that the project management, let's say we're not in the Agile environment, in project management, you dictate to the developer what the hours are. You're saying, hey, you have this many hours, and this is what you have to work with, this particular piece of software. And it kind of becomes a little bit of a dictatorship in the way that it is run. And with Scrum, I feel like that the team member during the sprint planning, the team member actually says, hey, that's going to cost this main story points. You know what story points are, right?

[12:18] RESEARCHER:

Yes.

[12:20] RESEARCHER:

OK. So, they'll take the main story points and what they'll do is they'll say, OK, I think we'll take the main story point to another team member. And I say, I could do it in this amount of story points. And in a sense, they're kind of playing a little bit of story point poker with a particular user story. So, whoever takes it the less, there's going to be some eyebrows raised and say, hey, are you sure you will do it in this amount of story points. And they're going to put out different reasons why. And then from there, they're going to be saying, OK, if you can do it in that amount of story points, let's say the difference is between three and five, there's not going to be any more than that. So, he says I can do it three, but the other guy says I can do it five.

[13:06] PARTICIPANT 27:

And the guy who says he can do it in three has been pretty consistent in building his software, pretty consistent in completing his story points, pretty consistent in the demos that everything's been working out really well. So the guy who says he could do it in five and the guy that say he can do it in three, have to have a meeting of the minds and say, well, what can we really do it in because we don't want to overestimate. We want to have a nice, good line for the burn up, burn down chart. Want to have good lines. The goal is not to complete all the story points in the sprint. I mean, it'd be great if we could. But the goal is to complete...o have a consistent burn down chart that actually has a good line going down.

[13:55] PARTICIPANT 27:

I mean, it'll be great to complete all the story points, but I'd rather have quality over quantity. That makes sense. They feel safe to invest the necessary time to meet the quality expectations. So, if it takes a little bit more time for a developer to get something done because they got in over their head or whatever, then the burn down and the burn up charts really got to show those scenes actually in place. So, the other thing I like about a Scrum is that there's a lot of transparency. Like you said, I do like transparency. I like the fact that you do empower the developer. You do have the developers work out there out front. And if something doesn't go good with the demo or whatever, it is on the developer. It's something that the developer says, well, who worked on this particular portion of this project? And the guy raises his hand and says I did. And it's like, well, what were you thinking here? Why does this not work out? With the sprint, you kind of want to put together a nice portfolio for a couple reasons. A, it helps you kind of keep your job. And B, it shows that you have a lot of value in the team. So, because the stakeholders are the ones that are actually going to be able to remove you from the project or remove you from the company if they don't think you're pulling your weight. There's a lot of different pieces, but I think it kind of allows the developer to have a lot of empowerment and a lot of ownership over the software that's being built. And I think that they're proud of that. And they want to look better than their buddies for sure.

[15:45] RESEARCHER:

Ok, fantastic. Let's get a little bit more detail. Can you describe your Scrum environment? You could choose any one of your experiences and describe to me the process, the people, how do they work together, etc.

[16:00] PARTICIPANT 27:

So, in the Scrum environment I worked on before, it is primarily we've had three to nine members, the membership actually grows based on the business need. We had to have an architect, we have to have a UX/UI designer. We had to have an actual graphic designer. We add those people in early because they're actually laying out and designing a user interface or taking care of a lot of the user experience. I'm a big user experience guy. So, if it doesn't have a good user experience, to me I don't think your software products will be that good. That's a whole other topic, we could talk about later. But the UI is really, you know, building a UI, having everything all laid out and really important.

[16:59] PARTICIPANT 27:

Why is that important? Because the developers, they are not the best. They're best at building functionality. They're not the best at creating the user experience. They're not the best at creating a user interface. So, start integrating the team into the graphical aspect of it. Then I start building in a frontend devs to actually start taking the software, creating the user interface in a programmable fashion. Whether we're using React, whether we're using HTML, whether we're making a prototype or whatever. The idea behind the end of the sprint, though, is to have a functional piece of software. So, the demo cannot be like PowerPoint screenshots of what the software is going to look like. It's the actual functional demo of the software. So, if it's a compiled piece of software, it gets a little more tricky. But by the end of the sprint, you want to actually have something that you can demo. In the Scrum environment, you start adding on people, no more than nine, so adding on people as the business need grows. So, as you start having a frontend guy starting to run into some backend type developer work, database development work, you need to bring a database developer or a backend developer.

[18:24] PARTICIPANT 27:

And you have to incorporate these people as the momentum is going for the project. So as business need comes. So, what's great about that is that you can start small and you'll have to have a lot of overhead for a lot of different personnel. The other thing you need, and I totally forgot this was you need a business analyst. You need a business analyst to be able to actually work with the product owner. The business analyst will more or less come up of all the different, I'm going to say, logistics of how the software is supposed to work, where it's supposed to get its data, how it's supposed to be able to take the data, where it's going to put the data. And then what's supposed to be displayed in the user interface, for example. And what's supposed to happen when we hit the submit button? Essentially, the business analyst is going to take all those pieces of information, get very granular, get very into the weeds, get very in deep and probably even write some pseudo SQL queries. Actually, drop that data into the database.

[19:37] PARTICIPANT 27:

So, the business analysts is really a go-to...he's not a not developer, he's essentially a logistics person that kind of looks at the data from a logistics standpoint and also kind of shows what kind of data is allowed to be shown. There's legal involved with the data. You know, they're the ones that are going to actually kind of come up with different requirements based on the legal aspects of it. It needs to be kept compliant. It needs to be PCI compliant. It needs to be wherever compliancy has to be. They're going to be able to handle all that, come up with all that. But they're also part of the team. So as part of the team, they are working within the Scrum environment.

[20:23] RESEARCHER:

What makes this particular Scrum setup good?

[20:31] PARTICIPANT 27:

When we're developing software, you have to have people that are good creatively. You have to have people that are good building frontend user interface. And I'm using my examples based off of web development. And then you also have to have somebody that can handle the backend functionality. So, the reality is, is that when you use your app on your phone, your phone has a frontend experience. All the software is actually broken out in different layers. I'm not sure if anyone has told you this, but there's a presentation layer, there's a data layer, there is an application layer, in all these different layers, essentially could be Scrum of Scrums. They actually could work on the same piece of software. So, when Steve Jobs is having people work on the iPhone, they did not know they're working on the iPhone because they're working on a piece of something. But when they got to the end and they put the product altogether, all the different pieces of the product together then that's when they knew that they built the iPhone.

[21:46] PARTICIPANT 27:

This is the huge advantage of planning anything in a Scrum or in an Agile environment is that you could also kind of keep everything secret. If you have any kind of confidentiality. Not everybody is going to have the whole piece of the puzzle. They're just going to have only a portion of the puzzle. I can have somebody at the same time doing frontend development, I can have somebody doing backend envelopment development and I somebody doing design work. All the same time in Agile environment where in a Waterfall environment, I have to wait before I get into development on the frontend, I have to wait for the designers and have their stuff done. This is one of the hugest advantages is that the project can literally be...the Scrum of Scrums can actually literally be broken up into different application layers, whereas there's different layers of the software process, where there's presentation data or application layer and nobody can know what they're working on. They're just getting directions from the project manager on what piece of the puzzle that they're working on. They don't have to have the whole story. They don't have to understand the whole story. They just have to be able to perform what they were requested to do. So, in a silo, developers can work in a silo without even knowing what they're working on. So, to me, I think that's a huge power right there all by itself compared to Waterfall.

[23:22] RESEARCHER:

Can you take me through the journey of our user story from inception to the release in your Scrum setup?

[23:34] PARTICIPANT 27:

Sure. So, let's go back to the log in screen. Let's use the players that we've talked about. The product owner says, hey, we have to have a login screen. We have to have a way to authenticate the users that are coming in so user story might say we need a login screen and we need to be able to authenticate and have some user information that is pertinent to the user's log in that will display. So, the business analyst will say, OK, let me get all the information about this from the product owner. The business analyst works very close with the product owner. The product owner essentially gets this information also from a stakeholder. The product owner says, I want to log in screen. That's pretty much the information he's going to get from their owner says. Let's expand on that a little bit. Let's talk to the business analyst. So, they create this user story, which is I want to have a log in screen.

[24:38] PARTICIPANT 27:

Well, from a developer standpoint, as a developer myself, there's a lot of questions that I have. You want a login screen? Do you want an email address? Do you want a username? What kind of password? You have any password requirements? Are you requiring double authentication with the device? So, I'm asking all these different questions. As a developer, they'll be asking that against the user story. So, the product owner will make decisions right then and there to refine the actual user story. So, the user story will say, I want to log in screen. I want to have double authentication. And I want to be able to have a certain type of password, 32 character password with symbols and make it required. And then we need an email address. That's an example of user story. So then developer, and say this is all the information I need. So, they start. There's some things that you don't have to get very granular with or some things you do have to get granular with the developers.

[25:46] PARTICIPANT 27:

Because developers will go freelancing all day long and build what they want, and you have to give them some direction. So, what I'll say is, we have to have a login screen. And so, the first question they're going to ask me is must we have a create a login screen? And I'm going to say, yeah, absolutely. So, I'll go back to the business analysts and they will say to the product owner, do we need to have a create login screen. Do you have any special requirements with that? Business analyst will say no, or a programmer will say no. And then we go through and we create what we think we should be a login screen.

[26:27] PARTICIPANT 27:

So, we create the login screen. We have our business analyst. We have our graphic designer, we have the UI person, all involved. And they're all creating this beautiful log in screen. To create this beautiful login screen, they have beautiful styles added to it. They have the graphics actually upload to like Zeppelin or something where the CSS is actually exported. So, like, they use tools like Sketch. They can actually build out a login screen without even going to code and then upload to like Zeppelin and Zeppelin will actually export all the CSS files for it, the placement, and the association to where the design placements is going to be and stuff. Have you heard of Zeppelin?

[27:13] RESEARCHER:

Yes.

[27:15] PARTICIPANT 27:

So, it's a nice little tool to have in your arsenal. And then from there, the developer can extract the code and the CSS from Zeppelin and actually build a good login screen. The developer starts adding the login screen and says, OK, I have all the frontend stuff done. And then from there says we need to be able to take the login and we push a submit button. That information needs to go into a database. So, there's a couple of different ways this happens like it can go directly into the database. So, I might need a database developer involved, to be able to actually take the data and put it into a particular table or particular field, and or it could be stored on a session in the application layer.

[28:10] PARTICIPANT 27:

So, we have some kind of business process happening on the application layer where it's taking the information and doing some kind of validation, it's verifying it, saying, hey, is this somebody that's already logged in before? Have we seen this person before? If not, then it'll input the information into the database. If it does find that you've already logged in before and you try to create a user account before with the e-mail address. Then it will say this user already exists, try to find the password or the forgot password functionality. That would be on the application layer, which talks to the data layer and then we'd actually bring in the information based on what every user's actions are happening. That's a beautiful process that is just been taken care of in a two week sprint versus having everything all locked down or Waterfall. Does that make sense?

[29:13] RESEARCHER:

Yeah, it does. Thanks for the detailed descriptions. I've noticed when you talk about the process in describing the journey of a user story, you didn't talk about quality. How do you make sure or how do you assure quality in this journey?

[29:34] PARTICIPANT 27:

So, again, I think I've touched on this before is that developers are the ones that are in power. So, they are the ones that actually, all of them are coming together, co-mingling, talking. They're the ones that are actually saying, let's put this product together and if there's anything flaky...you're talking about quality in regard to well, is it going to do what it says it's going to do. Or is anything nefarious happening in the backend. So, my job as a Scrum master or even as a product owner is to find, I always try to find my number two guy. So, there's a person I need to work with and implicitly trust to make sure things are getting done the way they way supposed to be getting done. He's the one that will help me ensure the quality aspect of it from a technical standpoint. Because I have a technical background, I can see what is being done and when it comes to demo time, the stakeholder will give a lot of feedback.

[30:49] PARTICIPANT 27:

So, let's go back to the login screen example. We get to the end of the login screen and we're done with the sprint. And there's demo, and the login screen works, and we show all this functionality to the stakeholder. And the stakeholder says, you know I really want to have a Google authentication, an Apple authentication, and a Facebook authentication with the login. That requires a lot of different efforts, not really from the graphic design standpoint but from a frontend standpoint, the backend, and the application layer we have to go and capture the APIs for Facebook, for Apple and for Google. So, there's different APIs we have to tap into. We need each of these APIs so from there the authentication is going to happen. So, the application layer person, frontend person and the database person, because the database person is going to keep a token to authenticate with the actual login of the user. We go back and add that to the user story and the user story is then refined. This is what we call Refining the user story. So, the user story is complete but not necessarily fully complete. When you build a story, one of the things that is also kind of defined, is a definition of done.

[32:25] PARTICIPANT 27:

So, what's the definition of done? This is different in different environments. Some people say a definition of done is I can demo it, I can show it to the stakeholder, some people say it means it's uploaded to the repository, some people say it completes all the criteria. So, you build user criteria based on each user story. You say the definition of done for a particular user story is this, this, and that. And you ask about quality and that's how you can do your checks and balances on the quality aspect. Does it accomplish A, B and C? If it does, then it's a good quality user story that's been accomplished.

[33:11] RESEARCHER:

Who makes those checks? Do you have a QA team or the business or the product owner?

[33:21] PARTICIPANT 27:

In the sprint planning, we define the definition of done. The product owner works with the team and says, you know, let's create a definition of done for each of the different story stories. Create this definition of done. This is done if I can, let's use the login screen again, this is done If I can put in my user name and put in my password, I submit and it authenticates me and it knows who I am and I can use that information as a session, as a personalized session on the backend. Let's say that's the definition of done. And so, if it meets those criteria, essentially the login the screen is done. Now where the backlog refinements kind of comes in, now I'm talking about Sprint planning again after a sprint review, is that now the product owner or the stakeholder says, I want to have a new requirement added to the log and screen. That becomes a whole new story point, right? That becomes a whole new story. So then with that story, we still define the definition of done. And while we're developing the login screen, it might come out during the process to say, that do we want to authenticate with Google. Do you want to authenticate with Facebook? Do you want a social media authentication that actually might come out during the sprint planning and then those questions could potentially be asked and answered before it even goes to development? So that could be part of the definition of done as well.

[35:19] RESEARCHER:

Fantastic. Thank you. Thanks for those details. Let's move to some storytelling. Can you share with me a positive story about Agile and software quality? If the story can have both of those elements, it would be nice.

[35:38] PARTICIPANT 27:

So, I've been involved with different startups. The thing about startups is they are very time and cost sensitive. Everything is time and cost sensitive. And this is before I started doing things in Agile. I did a Waterfall methodology, I actually wrote a full scope of work, scope work was very laborious. A lot of details in the scope of work. There's a lot of if then analysis and pseudocode in the scope work. There's lot of dependencies in the scope of work. And looking back on that today, I think that that would've been a very good case for Scrum or Agile because there are so many different tendencies and there were so many different decisions that needed to be made that I couldn't possibly make all the decisions myself when I'm writing the scope of work as a product owner. Does that make sense?

[36:40] RESEARCHER:

Yes.

[36:44] PARTICIPANT 27:

So again, one of the advantages of Scrum is that each software layer could be its own Scrum of Scrums. So, you could have somebody that's literally a Scrum team and literally focused on the design aspect of the product. You could have a Scrum team, literally focused on the development aspect of the product or the frontend aspect of the product. And in Jira, I don't like referencing tools because tools are not what a Scrum team. So, I'll reference Jira because it's a little bit easier to understand. But from a Jira standpoint, there's what's called an epic. And the epic is an umbrella, right? An umbrella of the category of the task. Essentially, you could have Scrum teams working in each epic, in silos, and I don't know what they're working on and each other. And what is really is the genius of it is, the Scrum master takes all the pieces and puts it all together and has a well-rounded product at the end. That's one example.

[38:02] PARTICIPANT 27:

I think that Waterfall has its place. I think Waterfall has its place with construction projects or certain types of financial projects. But I think that they've seen a lot more Scrum now in financial projects even. So, financial and technology has come in, so it's gone full circle right there. In my experience, the Scrum environment is extremely successful. So, I built a product that was a website which had a lot of data. It was for a [redacted], an organization that keeps track of all the different countries and all the different age groups of the children and adults that are struggling with social issues like social, medical issues like diabetes, hunger, starvation, just different types of things. And they keep track of it by country. And so, there is this huge spreadsheet that's actually inputted by everyday people that represent their country, in regard to this data.

[39:28] PARTICIPANT 27:

And they input all this data. The data had to be parsed. One thing I told the client, I said, this is a definitely Scrum environment. So, when we built this product, I leaned on the head developers who really help keep everything organized. You can put all the data into a Mongo DB, and me as a project manager, Scrum master was just facilitating Scrums. I was facilitating the daily standups. I was facilitating the demos and I was facilitating the sprint planning. And we were actually kind of putting together a kind of put together a Scrum environment. So, in a sense, the daily Scrum was helpful because there was a lot of work to be done and there wasn't a lot of people working on it. So, the data was a huge thing. And when I told the clients that the data is going to be whatever the data is and whatever it displays as is what it is. And you can't change data to morph your...because they are wanted charts to actually show the differences visually through a bar chart or a line chart. But they didn't like the way the data came in. It wasn't going to prove their point. They wanted the data to be modified. I was like, no, I'm not going to do that. You have to have some ethics in doing things this way, too.

[41:23] PARTICIPANT 27:

The client wants to be able to show that poverty levels in a certain country are astronomical and through the roof for political reasons or whatever reason they want to have it done or for financial reasons or for grant purposes or whatever. If the data doesn't support that, then where's the line get drawn in regard to is the data real or is the data manufactured? Does that make sense? So, a lot of conversation with the client about data integrity. Some of the fields weren't actually filled in from the user so that was discontenting because then the client was saying, why isn't there anything showing here? Well, you have no data that supports it. And there's a lot of back and forth in regard to the data integrity in the data mining. And we ended up having a pretty good product that actually was very accurate, which is really important. I think that was the biggest important thing. The accuracy aspect of it and it satisfied the accuracy. And in a Scrum environment, I don't think we would have got there if we used Waterfall for that.

[42:49] RESEARCHER:

So yeah, I see. I agree with you. Yeah. So, it's not always rosy, unfortunately, there are sometimes negative experiences. Can you share with me a negative experience or negative story?

[43:04] PARTICIPANT 27:

If a Scrum master and a product owner are kind of hardheads, if you want to use that terminology, like they're kind of determined to have their way done and no other way. To me, that's not a good fit for a Scrum environment. So, all Scrum is about is empowering the developer and actually giving the developer a lot of empowerment to build the software products. Take ownership, to have empowerment, to have transparency on where they're at with the product. But there's also this other thing in the sprint. It's called Sprint retrospective. And a great retrospective is essentially where the team kind of gets together. They kind of do a postmortem or a debriefing of what went well to Sprint, what didn't go well. Why were there so many problems with this and those type of things and have it be non-punitive. Because the reality is, is that they need to be able to speak freely without management being present. They need to be able to speak freely about the product owner being present, and speak freely about what their struggles are.

[44:20] PARTICIPANT 27:

And then at the same time, you come up with resolutions for those things. In a Sprint retrospective, I've been involved in Scrum teams where that wasn't done correctly, and it was punitive, and it was so damaging. It was like a reality show, everybody gangs up on one guy and then all of a sudden, the guy doesn't feel like he's part of the team. Kind of feels like he's not doing a good job or whatever. And that person is going to get replaced with somebody else. And that's not the purpose of the sprint retrospective purpose. The purpose is to have the team take a deficient area, be able to actually build up that area and encourage that person to be better the next time around when the sprint happens.

[45:13] PARTICIPANT 27:

So, any kind of training or mentorship that can happen, will happen in the Sprint retrospectives. I've been involved in Scrum teams where that hasn't happened, and it's been very detrimental. So, Scrum really works when you use all the different ceremonies. You can't just negate something because it doesn't fit with your organization structure. You have to do all the different pieces. Cover your bases and actually make sure that the team functions. The whole goal of Scrum is to have your team have more velocity. So, what took three hours to do before, should only take one hour because the team has four velocity. So that's what you're trying to build as a Scrum master. The Scrum master doesn't really do anything except for facilitate and set up the situation so that it happens. So, to me, that's the negativity of Scrum. The other thing, too, is that organizations have to pay a lot of time for meetings.

[46:21] PARTICIPANT 27:

So, think about it this way. Daily Scrums are a fifteen minute time box every day. Sprint planning is two to four hours, sprint review is two to four hours. That's a lot of time that's not developing and not being productive. So some organizations have to have what's called an Agile coach that actually teaches and helps the management and helps the teams that are going to be in place that are going to be doing Scrum to actually understand what the expectation is and how to actually be more Agile friendly. So, I say that in the states there's not a lot of environments that are truly Agile friendly. So, if you hire a Scrum master, you're going to have to hire a Scrum master and an Agile coach.

[47:19] PARTICIPANT 27:

You have to have somebody that actually knows the methodology inside and out. You're going to have to be able to facilitate that in the corporate environment, that's not used to it. And that could be very challenging. Talk about a lot of disgruntled people, C-level people just don't care. All they care about is the end results. But to get there, like management or like directors, they're very into the weeds, they're granular and they're into people's lives. And they don't like a lot of means, it's like why we waste all this time. Does that make sense?

[48:00] RESEARCHER:

Yeah, of course. I've been in environments like that where we do have a meeting about a meeting.

[48:10] PARTICIPANT 27:

That doesn't happen too often, there's meeting and after meeting. I've been in project management environments where that's happened. But in an Agile environment, if we keep to the time box, which is really important for the different ceremonies and we keep to the task at hand and not joke around or talk about somebody or talk about politics. When you talk about what you have to talk about in these meetings, it's actually pretty productive in regard to getting our backlog all refined. So, I mean two, four hours, is a lot of time. But to me, I think it's an important time to be able to actually empower the team. Have them involved in the planning aspects of the project and help them to know that they're making a difference and making a part of building this piece of software for this customer. I think that's a difference. I think that they take a personal responsibility to the software. Let's say it's Apple or something, like hey, I built this page. You know it's given you some pride in your work. And I think that it reinforces and reconstitutes the aspects, having pride in your work and taking ownership in your work, and I think it just facilitates success for that.

[49:57] RESEARCHER:

I'd like to follow up on something very important you mentioned. The importance of Agile or Scrum ceremonies in making the process work. Can you elaborate a little bit further on that?

[50:12] PARTICIPANT 27:

Certain Scrum ceremonies are time boxed, right? When I say time-boxed, it means that there is a certain time limit for each of those ceremonies. So, like, daily Scrum is not a time to actually talk about all the problems that I'm having with this particular developer or this particular product or particular product owner. Daily Scrum is essentially what am I going to accomplish today, what I want to accomplish tomorrow and what I have planned to accomplish by tomorrow. And essentially its regards to the stories so that developers taken ownership over stories. Let's say it's a story block A, B and C. That's great. You know, taking over those story points and they're taking ownership over it. But then the next day, if I go, what are you going to accomplish today? It's the same developer. What do you have to accomplish tomorrow? And they say, well, I'm still doing A, B and C. Then my eyebrows are going to be raised. Didn't you do that yesterday?

[51:17] PARTICIPANT 27:

So, during the sprint ceremonies as one of the ceremonies that's really important is the daily Scrum. It's the fact that they have to have the transparency, the have their ownership over their particular story areas. And the idea is that you want to have each day be a different story block, right? They should be saying, hey, I'm done with A and B and I need to finish C. That's acceptable to me on the same day. But if they say I'm still working on A, B and C altogether.

Then what have you been doing for the past six, eight hours at work? What have you been working on? So, the ceremony brings out the deficient workers. Also, it builds intuition and actually builds a lot of ownership with each developers to do the job. So like me as a developer, when I was doing story points and I was like, well, I've dealt with story A, B and C, and I'm working on D and I kind of just picked up D and I said, I'm doing D now.

[52:39] PARTICIPANT 27:

That shows initiative and that shows a drive and that shows the people that are in charge that I'm a go-getter. I get things done. And, you know, the work's getting done. And I'm taking ownership and I'm taking responsibility for my particular portion of the project. So, I think it actually builds that in inherently, which in turn, when you have somebody that has what I like to call tribal knowledge about the product. So, here's the deal is that anybody could be hired to do pretty much anything, right? But if you have somebody that has a lot of experience with a particular system or a particular way of doing things or a particular process, or the ability to say hi, what I call the ability to say hi to a system. That is invaluable. That's what they call tribal knowledge that's invaluable to be able to replace that person. Does that make sense?

[53:40] RESEARCHER:

Yeah it does.

[53:41] PARTICIPANT 27:

You don't want to replace that person. You want that person to be happy. You want that person to be OK. And part of the Sprint retrospective, which is one of the ceremonies, is one of the questions that I do ask and say, hey, are you okay? Are you happy? Are you pleased with the work that he'd been given? Are you feeling like you're getting everything you want to get out of doing this work? Some people like to be challenged and some people just like to get by. So, during a Sprint retrospective, I have those conversations with the individual developers or the development team. They might come back and say, hey, you know what? We're going to be good. We feel like we're going to be able to get things done quicker than what we thought. That's a great thing for a Scrum master to hear. That doesn't necessarily mean it always happens.

[54:35] PARTICIPANT 27:

So, that's what the Sprint retrospective is for. And then Sprint planning is essentially before the sprint starts, you take some time, time box between two and four hours. Depending how big the product backlog is, which is generated by the product owner, essentially the product owner writes a bunch of stories and then we take the stories, we look at each of the stories and then during that two to four hours, we actually say, how long do you think it would take to be able to actually do this? And somebody says, well, it's going to take a couple of days to do it. Somebody else says it will take about half a day to do it. Well, that's a big difference, right? The difference in story points. And so, then we have a conversation about that. We say to the developers say, well, why do you say it's this? Why do you say it's that? You do compare contrast. And then all of a sudden, they come to a meeting somewhere in between, then say it's going to be about this. And so, the people that spoke up are the ones that usually take on those particular story points. And then they ask if there's any questions about story.

[55:39] PARTICIPANT 27:

If there's a question about story, most of the time during the sprint planning the product owner is there present. So, questions will go directly to the product owner from the developers. Which is nice. And the product owner can answer any kind of high level questions about that particular story. And what that story is trying to accomplish. The developer says, yes, I understand. This is the story point, this is what it's going to cost and then add it to the sprint. So, you refine the story a little bit in the sprint planning to actually mimic what the product owner intended. The product owner might have written it in a way that was abstract, and then the developer and the product owner can actually rewrite the story point or story, or item to actually be more defined in detail. And then with each story item, there is a definition of done right. So, the definition of done says this is done if A, B and C is accomplished. So, when that story point is being worked on, you make sure A, B and C is accomplished. And that's the definition of done for that particular story. So that all happens in the Sprint planning and the product backlog.

[56:59] PARTICIPANT 27:

And then on the Sprint planning, we start building out our sprint, say, OK, we have so many story points available during the day. We have so many stories with points added to them. And then we just do simple math and say now add this story, this story, and this story. Then the product owner comes back and says, well, I don't think we should have this story yet. We probably should add this story. And that's what Scrum Master and the team kind of works on together says, well, I don't think we should add that story until we add this story, because that story is going to be dependent on that story. Then there's a meeting of the minds. The product owner doesn't just get to dictate what stories should be done first, and then the team takes ownership of the product from the sprint planning standpoint, adds the definition of done, adds the story points. And then we start to sprint, and we start to sprint. Then we have the daily Scrum time box for fifteen minutes. We talk about story points. What each developer has accomplished today? What the developer plans on accomplishing by the end the day for the next daily Scrum. And then we keep track of that, and we compare the work product to the definition of done.

[58:17] PARTICIPANT 27:

So, if the work product is equal to the definition of done, then it actually is considered done. And each story, I don't know if I made this clear or not, each story could have its own definition of done. So not every story has the same definition of done. Individual stories can have different definitions of done. So, it could be compared to each particular story. And that could be the actual definition of done. And then at the end of the sprint, which is typically two to four weeks, I like two week sprints because you can't get too much trouble in two weeks. I mean, in two weeks, you get some work done. And then when you get to demo, you really can't screw up a lot of things. In four weeks, I think you can. But in two weeks, I like two week sprints we do the demo with the product owner and the stakeholder. The product owner or stakeholder are looking at the demo of the actual piece of software we're working on, they are able to actually play with it and touch it, be able to actually work with it. And they're going to give a lot of feedback on possibly user experience, possibly on the look and feel, possibly on the functionality, possibly on the data, where the data is going. How's the data being mined. All that good stuff.

[59:44] PARTICIPANT 27:

Those get converted into stories and actually get added back to the product backlog. So, the whole process happens again. So that's your Sprint review. The product owner and the stakeholder says, hey, I really like what you did here, or I don't like what you did here. I really prefer to see this, this, and this. The whole team's hearing this at the same time. And then we get to sprint retrospective, it's more or less a meeting of the minds with the developers. Kind of have a little of Kumbaya and say is everything going OK? Are you happy? Are you happy with the work? How are things going? And then also the Sprint retrospective is also the chance to say, why is developer XYZ having problems building this particular functionality. This shouldn't be difficult. And then you have those kind of discussions as well.

[1:00:44] PARTICIPANT 27:

And then it becomes a meeting of the minds that developer XYZ is a good developer, but he's just not very good at this particular portion of the project. Then you yank that particular developer out of that particular portion of the project and you put somebody else in place, but not remove him from the team, you just remove him from that particular portion of the project, you give him something else to do. So that's what the Sprint retrospective does. That's how it's supposed to work. And then the process starts all over again. Another two weeks before that, we have sprint planning and we have the daily Scrum and then we have the sprint review. And then we have a sprint retrospective.

[1:01:28] RESEARCHER:

Thank you very much. That was very, very detailed and very knowledgeable. The last question, it's a little bit provocative, but the purpose is not to upset you, but just to get you to share with us your opinion. What do you think of this statement: Agile produces poor quality software?

[1:01:57] PARTICIPANT 27:

I don't believe that statement's true. I believe that I've proven over and over again compared to the Waterfall methodology. So, Scrum is just an add on, if you will, to Agile. Agile has a lot of different frameworks. There's Kanban, there's...

1:02:19 RESEARCHER:

There's XP...

[1:02:21] PARTICIPANT 27:

There's all kinds of different types of...

[1:02:26] RESEARCHER:

Implementations.

[1:02:27] PARTICIPANT 27:

Yeah. I think Agile can produce poor software if you don't implement the processes or implement the framework appropriately. But I think Scrum is actually is a foolproof way to actually be able to develop a really nice piece of software. I think it gets rid of a lot. What Scrum introduces is a lot of checks and balances, which I really like. I think Scrum allows you to have a lot of changes in the software on the fly. I think Scrum allows you to have local teams working on pieces in silos from each other and still being productive. So, I don't agree with that statement at all. [redacted].

[1:03:30] RESEARCHER:

Not at all. It's not a true statement. Just to get people to give us their deep beliefs about Agile. But people who think this way, like you said, have experienced a poor Agile implementation. So that's the answer we were looking for and you're spot on. You stated that really clearly, which is really great. Thank you. That brings us to the end of the interview. Thank you very much for your detailed answer. And it was very insightful. I really enjoyed it. Thank you very much. Do you have any questions for me?

[1:04:27] PARTICIPANT 27:

Maybe I get to [redacted] and we can hook up or something.

[1:04:26] RESEARCHER:

Thank you very much.

[1:04:29] PARTICIPANT 27:

Take it easy. Bye.