[00:00:00] **Anna Domascan** Hello, everyone. My name is Anna Domascan, and welcome to my talk for accounting for requirements engineering in test automation. Briefly, around the agenda, I'm going to talk a little bit about who I am and what I do, as well as why I consider this topic to be important.

[00:00:20] We'll talk about what are requirements and why do they need to be engineered, as well as what is test automation and how the requirements are relevant for it? And lastly, we'll talk a bit about how you can improve the process of requirements engineering within the test automation. As mentioned, my name is Anna Domascan. I work as a technical consultant of Keysight Technologies, helping and guiding our customers. Introducing test automation with Eggplant Suite. Previously, after working as a trainer for a project management business analysis, as well as an application engineer catering to a variety of customers introducing automated document management systems within their organizations.

[00:01:05] I've talked and worked quite a bit around requirements and project management. There was a variety of things that comes related to software and software engineering. And among others, requirements engineering tends to be one of the tasks that might not always be treated with the importance that it deserves. Nonetheless, I would consider to be one of the most crucial phases for any software system to be defined and designed. Let's look briefly at what are requirements? In product development and process optimizations requirements.

[00:01:40] A requirement is a singular documented physical or functional need that a particular design product or process aims to satisfy. Commonly, its use in formal sense and engineering design, including examples of system engineering, software engineering, or enterprise engineering. Would be one of the definitions that you will most likely find when you search for the term requirements. Requirements themselves can be obvious or hidden.

[00:02:04] They can be known or unknown, expected or unexpected from the customer's point of view or also from any stakeholders point of view. Each and every one of us being a stakeholder to a project can have certain requirements that we know or that we don't.

Further down, you can specify and separate requirements into different categories. One of them being the overseeing one, you can consider requirements to become one for addressing business or coming from the business or others that are addressing the solution that is being developed. Further, you can write the business requirements into general and technical requirements.

[00:02:39] General requirements being business constrained, certain business policies, legal requirements, branding, culture or language requirements. Anything that comes directly from the business in the way the business is conducted, depending on the industry that you're in. There are also technical requirements, which are sometimes set a certain frame in terms of what is already available within the organization or also defined needs offer would be to be introduced into the organization after the project itself has been completed. Technical requirements relate to hardware and software is available also to interfaces, for example, within the system or to other systems, as well as network connectivity. Requirements concerning the solution address to functional or can be further specified into functional and nonfunctional requirements.

[00:03:27] Functional requirements are anything about what the system can do, such as data entry, data maintenance, any procedures that it needs to have, and retrieval requirements on how data fetch back from the system, as well as nonfunctional regarding performance and security, as well as legal, backup, recovery, archiving, maintainability, availability, visibility, capacity, and business continuity at the end of the day. Anything that sets the frame apart from how is information stored and processed within the system?

[00:04:06] What else does it need to fulfill in order to be considered a valuable and a valid system for a particular business? Regarding that, requirements engineering is one of the processes of establishing the services that the customer requires from the system and the constraints under which needs to be developed. Keeping those four categories in mind, you started the process of identifying what is specifically needed, ensuring that you're building the right application and you're building it the right way are among the most crucial parts of what you would have or what you need to consider within your software engineering process or within any engineering process at the end. And one more point to keep in mind.

[00:04:51] Over 80 percent of faults are introduced during requirements, analysis, or even before you actually start building the solution. You might already face some challenges or create errors that would later on result in the system either not being fit for the business or otherwise, not working to the level that is expected based off quality. But overall, you can divide the requirements engineering process into four phases. You have the requirements eliciatation or the discovery of all of the requirements, and this is a process of collecting data and information from all the possible stakeholders with all the relevant stakeholders to the system through interviews for workshops, through observation, focus groups or prototyping, for example, just identifying what do our stakeholders really want.

[00:05:41] What does that look like? Once you've collected all of that data and information, you can start dividing that further and performing an analysis on those requirements. You analyze them to be unambiguous, clear, and atomic. There is a variety of techniques that can be used in order to subdivide them and classify the requirements that you have. But among others or among the most relevant. I would consider also the smart analysis.

[00:06:07] You specify that your requirements are specific, measurable, achievable, relevant in timeframe. You establish not only the clarity, but also the automization of those specific requirements. Next step is to validate that what you've collected, analyze, and structure is actually reflecting truly what the business needs and what all of the stakeholders can agree upon. And later on, once that document is validated and approved by all of the relevant stakeholders to requirements management. So during the project lifecycle or the time where would within which the product is being developed, any of the arising changes can be accounted for.

[00:07:01] You can also always identify where the requirement is coming from. Why is it relevant to the system and how can it be addressed? And who needs to? Who's responsible for it as well? Managing changing requirements during the requirements engineering process and during the system development phases and making sure that whatever changes might be due to technical difficulties, it might be due to changing legal requirements.

[00:07:27] It might be also just different business environment that arises around you, that all of those changes and their effect on what system you're developing can still be reflected within the requirements that are used for the system development. Coming to test automation. To briefly mentioned, test automation is about using purpose built software tools to control and set up test preconditions. It can also be tools to execute specific tests. And also comparing actual outcomes for the predicted outcomes. Similar as to manual testing, where you have a certain step and you have an expected outcome in the same way within the test automation, you still have your current actual outcomes compared to your predicted or expected outcomes.

[00:08:21] Test automation itself is about running or helping you run many test cases consistently, repeatedly and in a variety of environments and systems under test. Test automation in summary test automation is about using purpose built software tools to control and set up test preconditions. It's about executing tests using those tools or potentially a different set of tools and comparing actual outcomes with predicted outcomes. Similar to manual tests where you have a test precondition and an expected outcome, you still have outcomes of your test automation which need to be compared with your expectations or your predicted outcomes. Test automation is there to help you run many tests in a consistent, repeated way on a variety of environments or systems on test.

[00:09:17] Why do we need test automation to begin with? Among others, the objectives of test automation are considering improving your test efficiency, providing wider function coverage within those tests, reducing the total test cost, performing tests that manual testers cannot. But also shortening the test execution period and increasing your test frequency or otherwise reducing the time required for the testing cycles or both at the same time. Those are, among others, objectives that are listed by the International Software Testing Qualifications Board regarding the test automation and considering the current I.T. industry and the situation that is in. With more and more acceleration of your software development lifecycle.

[00:10:10] So when emerged in the past 20, 30 years from Classical Waterfall Project, where you build everything as a phase and then only or closer to the end to come to testing and the actual release of the software and everything is planned what you have and you expect the project and process to run over months and years. Over the last years, with spread up the process or accelerated the process to be more and more agile. We don't want to necessarily plan everything we have. We want to plan and design for change to a certain extent.

[00:10:50] We want to be able to address emerging and changing requirements in the same way that it looks like within the Agile Manifesto, which also leads us to much, much quicker release cycles because we have shorter iterations and in which we're working and we using our variety of backlogs and requirements list, as well as in the last couple of years, the spread and the introduction of Devops, and as a way of working within different departments of the I.T. in different parts of the I.T. industry, is accelerating that even further and further and further, because you want to have an automation which you can continuously develop and continuously integrate the software that you're building into the productive environments of your customers and make those systems available to your users as soon as possible.

[00:11:44] Right. You have less and less time, more and more parse to test. Therefore, test automation becomes one of these solutions that can help you address those challenges, so it gains more and more important. It still comes with its own challenges. On the one side, you have a certain investment to set up your test automation solution. It's not only about purchasing licenses for your software, it's also about providing the infrastructure for it. You require certain additional technologies which are introduced within your organization in order to do so, as well as additional specific knowledge within the development and test automation. In order to be able to not only build the software, but also deploy that and run your test on it.

[00:12:29] Mainly and also from my perspective, it requires a certain maturity of the testing process. If previously tests were pretty much undefined and unstructured, bringing test automation into it might cause further chaos within the organization because still. Beforehand, nobody knew what the testing or just tested something. And now they still don't know it, but now they're doing that much quicker. That is not necessarily what you want test automation to be performing for you. And that would potentially send you on loops and iterations of tests that have not been completed as you would expect them to.

[00:13:08] Test automation being a major overhand to the team, to the organization without actually providing any return on your investment. Any potential benefits of what we were searching for by achieving the objectives that I've listed earlier, test automation also requires to have a set of understanding of which tests can be automated and which should not. That's still also just because we introduced test automation into an organization doesn't mean that we do not need manual testers.

[00:13:42] There might still be scenarios where your test automation will take you a couple of weeks to actually set up, whereas it takes a regular manual tester about five minutes to perform the actual test. The question is where is the trade of? Where do I get the benefit from putting that into my automation plan? And mostly one of the biggest challenges, test automation needs a control process for the development and implementation, you still need to be quite sure and have a defined line.

[00:14:14] How is test automation defined? How is it built? And how is it implemented at the end of the day? How do requirements have an impact on our test automation? One of the key points to consider, and definitely one that becomes the most critical one around a test automation is still a software development process you developing software in order to test other software with it. Therefore, it follows a certain sort of development lifecycle with it as well. Apart from being an emerging needs, being something that is already introduced in something that might be either deprecated or reworked later on, it still follows the same phases of analysis, design, development, testing and release. Just not about releasing to the end user, but releasing to the organization so that software can be tested with the software.

[00:15:08] Therefore, it should also be planned as a software project. You still need to define those phases and set up your test automation in an ordered way, so you need a clear definition also of the expected outcomes. It's not only about the outcomes of the particular tests, but also how much automation whether you need that. When do you need them? And both software, software requirements and the testing requirements should be defined and considered when introducing test automation. From my perspective and my experience within the industry, I would address the following or identify the following five points of strategies to improve the situation and elaborate on them a little bit further as well.

[00:15:58] One of the key points or one of the strategies to address this problem with is to set up test automation as a project. Define a project manager and find somebody who would be catering to those needs of eliciting requirements, analyzing them, documenting them, and validating them, as well as managing those requirements which are out there specifically for your test automation. Check the software development and testing requirements. Look for matches, look for them addressing each other and working with each other interchangeably, either the software requirements or the the overhead or the overlaying project for which you're building up to test automation.

[00:16:42] Is that clearly defined? Are those requirements actually testable? Yeah. Also within that clarify any ambiguities that come up ideally at the end of or even before you start developing the actual automation, you should not have any question marks each and every one of the steps within the way and each and every one of the parts of the software that you're building up needs to be clarified. You should also be able to address within each and every one of the steps that that your test automation is covering. You should have a clear definition of how test automation is supposed to address your system under test. Create a plan for your automation development in terms of what are the phases. What are the timeline? What is going to be released? Where do we start?

[00:17:33] Where do we plan to end this? How we splitting that, especially if you have your test automation as a standalone organizational unit, for example, within a competence center of your organization? How does that correlate to the software development releases, for example? Which pieces are addressed first, which should be addressed later? And this brings me also to my last point around synchronizing your test automation effort with the software development phases. So ideally, what you want to achieve is to have your test automation set up and ready so that the next release of your software can be already automatically tested. Right.

[00:18:13] Those phases need to be synchronized among each other. If you know when a certain phase within your software development is beginning, it should be a good time also to start introducing parts of the automation. It might sometimes be easier for existing systems, because you have already something that you can build upon, and if certain functionality is not changing, but new ones are added to it. You can evolve your test automation just to adhere to the new functionalities rather than reworking the old one. But also, if there is already an existing piece of software, you can work towards automating the tests for that existing part and then keep on adding new ones. Sometimes it might be also a bit easier with newer software that's being built.

[00:19:01] You might not have the full range of functionality yet, but this is still being released in certain phases and pieces. Overall, you don't have to work off in the sense of a certain. That we're automating. Twenty five preexisting functionality, if you just get those 5 released, you need to automate only for those. Both of them can be challenging in their own way in terms of visibility of systems under test, in terms of the amount of functionality in terms of the clarity of defined requirements.

[00:19:33] Nonetheless, check with the development what are their phases, what their project look like, what is their release cycle look like in order to ensure that your test automation is there on the right date, at the right time in order to run and assure that the software product itself is fulfilling all of the needs and all of the requirements that were raised by the business.

[00:19:59] I hope this helps you and you'll find new thoughts and new ideas of how you can work and improve your test automation within your organization and also help you build better software overall. If you have any questions, please raise them to us and thank you very much for your attention.