Erik van Veenendaal

Requirements Engineering, also for Testers

Column

Testers use requirements as the basis of test cases, review them for testability, and often participate in general requirement reviews or inspections. Unfortunately, many testers have little knowledge of or skill in requirements engineering. What level of quality and detail is realistic to expect in requirements documents? What does testability really mean? How can testers help improve requirements? Testers should be able to answer these questions and more, possessing skills in requirements engineering.

We complain about requirements "Can't test this, not clear, not unambiguous" but have no clue and are unable to answer questions in return such as "What do you think is a good testable requirement?".

However:

- we are one of the main stakeholders, risk analysis, test designs are based upon requirements
- we are involved in requirements reviews, what level of quality is reasonable?
- test designs may even be used as requirements
- sometimes (in agile) we identify and specify requirements
- we have a major interest in requirements and are heavily involved!

Agile

The IT-world has changed and most companies practice some kind of agile development, at least in part of their projects. In agile the tester is even more involved in requirement than ever before and contributes to documenting requirements and its acceptance criteria. User story is one of the primary development artifact for agile project teams. In agile methodologies requirements are prepared in the form of user stories which describe small functional units that can be designed, developed, tested and demonstrated in a single iteration. These user stories include a description of the functionality to be implemented, any non-functional criteria, and also include acceptance criteria that must be met for the user story to be considered complete. Testers have heavily involved in documenting user stories and its acceptance criteria

Broaden your skill set

There are trends in software testing that the (traditional) tester needs to be aware of and respond to. Knowledge and skills will be a challenge in the very near future for many testers. It is just not good enough anymore to understand testing and hold an ISTQB certificate. We will not anymore work in our safe independent test team. We will work more

closely together with business representatives and developers helping each other when needed and as a team trying to build a quality product. It is expected from testers to have domain knowledge, requirements engineering skills, development scripting skills, and strong soft skills, e.g., on communication and negotiation (figure 1).

Test knowledge ▪ test principles ▪ techniques ▪ tools, etc.	**IT knowledge** ▪ software development ▪ requirements ▪ configuration management
Domain knowledge ▪ business process ▪ user characteristics	**Soft skills** ▪ communication ▪ critical mindset ▪ presentation and reporting

Figure 1. Testing skills and knowledge

Now understanding that as a tester one needs knowledge and skills in requirements, there are many options. Some testers take in a course in Requirements Engineering based on the IREB certification scheme, other course being available as well of course, some practice apprenticing, etc. Whatever it takes to get the job done.

Five success factors

Based on many years of experience in Requirements Engineering, I would like to point you to five critical success factors that I would recommended the tester to start digging into:

1. Requirements attributes

Requirements are much more than "just" the sentence, consider documenting its rationale, priority, requirements type, related use case etc. Requirement Attributes are properties of a requirement. They capture important additional information about a requirement. Usually the requirements attribute evolve into a card (e.g., user story card) being used in a project or organization (see figure 2). Don't go overboard, define a practical set of attributes that all have added value.

Requirement #: *Requirement Type:*
Event/Use Case:

Description:

Rationale:

Source:
Fit Criteria:

Priority:
Supporting Material:

Figure 2. Example requirements card

2. Requirements acceptance criteria

Acceptance criteria (also called fit criteria) complete the definition of the requirement. We have to be able to tell whether a solution completely satisfies, or fits, a requirement, they will make requirements measurable. It is often much easier to add concrete acceptance criteria than to write a 100 % unambiguous requirements. Acceptance criteria in some way detail the requirement.

3. Requirements rules

The discussion on "what are good requirements?" is endless. Of course it depends on the context but most important is needs decisions. A concrete and usable requirements rule set should be defined that leads to "good enough" requirements your context. Discuss and define rules on issue such as iidentification, annotation, changes, consistency, language, brief, unambiguous, rationale, quantify and compound.

4. Requirements templates

Instead of re-inventing the wheel over and over again, use templates when defining both functional and non-functional requirements. They provide consistency and contribute largely to a higher level of unambiguousness. It is even more efficient, so why not tomorrow? For stories typically the following format is applied "As a <role>, I want <goal/desire> so that <benefit>". Other common templates include:

The <stakeholder> shall be able to <capability> (e.g., The order clerk shall be able to raise an invoice)

The <product> shall be able to <action> <entity> (e.g., The launcher shall be able to launch missiles)

The <product> shall <function> <object> every <performance> <unit> (e.g., The coffee machine shall produce a hot drink every 10 seconds

5. Requirements reviews

Reviews are by far the most effective and efficient quality assurance measure to find defects. However, this is only true is applied well. Balance practical vs. theory is one that is very true here. Understand the difference between a walkthrough and inspection, these are different processes, with different stakeholders and different objectives. Start with your objectives and define a review process that matches these objective.

I have been running a tutorial called "Requirements Engineering for Testers" for a few years now, maybe I will see you there ... ■

> about the author

Erik van Veenendaal (*www.erikvanveenendaal.nl*) is a leading international consultant and trainer, and a widely recognized expert in the area of software testing and quality management. He is the founder of Improve Quality Services BV (*www.improveqs.nl*). He holds the EuroSTAR record, winning the best tutorial award three times! In 2007 he received the European Testing Excellence Award for his contribution to the testing profession over the years. He has been working as a test manager and consultant in various domains for more than 20 years. He has written numerous papers and a number of books, including "Practical Risk-Based Testing: The PRISMA Approach" and "ISTQB Foundations of Software Testing". He one of the core developers of the TMap testing methodology and a participant in working parties of the International Requirements Engineering Board (IREB). Erik is also a former part-time senior lecturer at the Eindhoven University of Technology, vice-president of the International Software Testing Qualifications Board (2005–2009) and currently board member of the TMMi Foundation. You can follow Erik on twitter via *@ErikvVeenendaal*.