RBCS

TIME TESTED.
TESTING IMPROVED.
www.RBCS-US.com

Project Retrospective

An excerpt from The Expert Test Manager: Guide to the ISTQB Expert Level Certification *book by Rex Black, Jim Rommens and Leo Van Der Aalst due to be published by Rocky Nook. All material is provisional and may be subject to change.*

The aim of a test retrospective – regardless the development approach such as waterfall when working traditionally or scrum when working in an agile way - is to learn from experience gained during the completed testing phase(s) and to document the lessons learned and improvement suggestions for future projects or sprints. As a test manager, you should be able to organize and lead evaluation sessions. A possible way of organizing these sessions includes:

- Identify and gather evaluation topics
- Require each team member to participate
- Ensure mutual trust (open exchange of ideas, no personal 'attacks')
- Collect all issues and group them
- Form pairs and give them three minutes to think up as many actions as possible per group
- Rotate the pairs after three minutes, having them move on to the next group
- Run through all actions
- Provide each participant with three votes and ask them to select their favorite actions
- Identify the most popular actions and determine the action owners
- Set a maximum time limit, for example, of 60 minutes for an agile retrospective (could be a few hours in a traditional environment)

These 60 agile retrospective minutes apply to the effective amount of time needed. I often have combined such an evaluation session with a lunch, because it was the end of a project and the people had done their best to make a success out of it. Personally, I think it is important to pay some attention to this achievement and thank the people who have done such an excellent job.

You could extend the evaluation session with a more personal feedback when useful (maybe for resolving any interpersonal issues):

- Tips & tops[1] for the team and for each other
  - Stick notes onto the board (at least one tip + one top) for the team
  - Offer feedback to the team member via a note (at least one tip + one top) per team member
  - Remember: feedback is always constructive!

Another structure for outlining an evaluation in an agile environment (retrospective) is given by Derby and Larsen[2]:

- Set the stage
- Gather the data
- Generate insights
- Detect what to do
- Close the retrospective

Often these retrospective sessions are held at the end of a project or sprint, but it is a smart thing to organize these on a regular basis during the project or sprint as well. This is the best way to continuously improve what has been learned. During the retrospective sessions, you might take the following feedback rules into account:

When you are the sender of the feedback:

- The feedback should be wanted by the receiver and of good quality so the receiver finds it useful and would like to have more of it.
- The feedback atmosphere should be one of mutual trust.
- Describe briefly, without judging or interpreting, what you observed and start with something positive first.
- Remember it is your – subjective - feedback, so use “I” instead of “one” and “you”, and make sure your feedback is as specific as possible (don’t make it universal).
- Provide concrete and realistic improvement suggestions (don’t say “you might think of another approach”, but come up with a possible other approach yourself).
- Don’t wait to give feedback until the organized retrospective session, but provide it during the project or sprint whenever you think it is useful or necessary.

1 Tips and tops is a feedback method where the tips are recommendations and the tops represent what you did well.

2 Esther Derby and Diana Larsen (with co-author Ken Schwaber) explain in their book “Agile Retrospectives (Making Good Teams Great)” how to use tools and which tricks and tips you could use to organize a retrospective in general and how to organize a retrospective specifically for your team and organization.

When you are the receiver of the feedback:

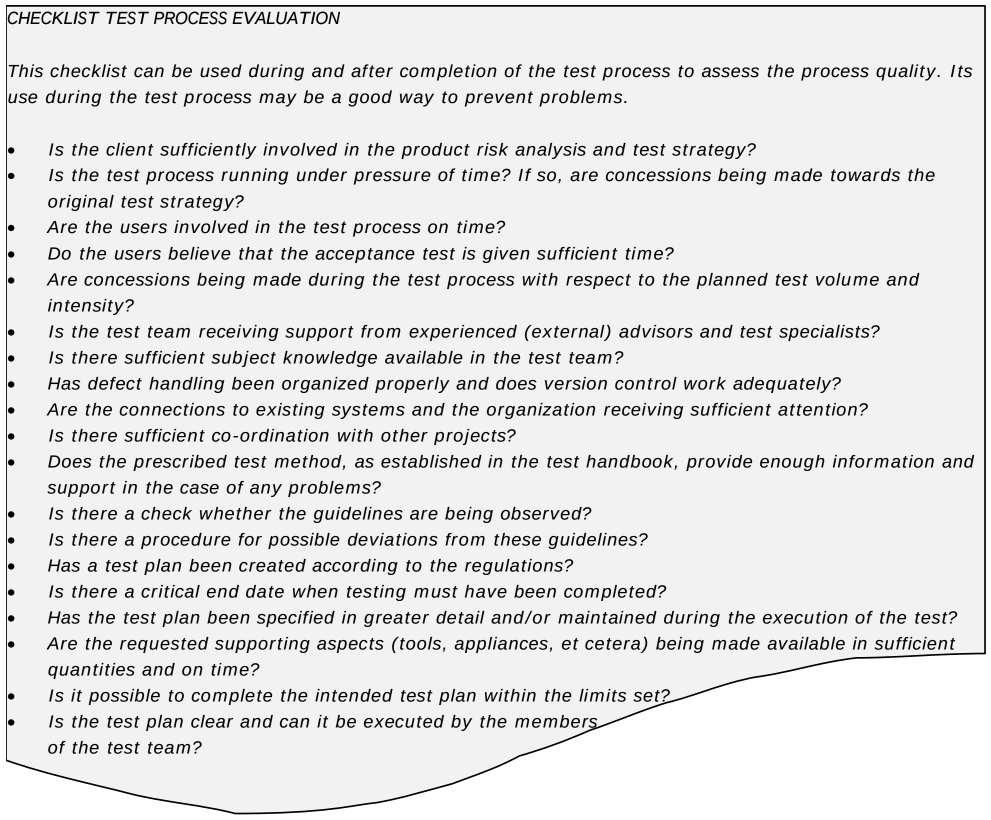
- People have thought about it, so value the feedback, listen to it, learn from it and thank the feedback provider for giving the feedback.
- Accept the feedback as it is. You could ask questions for a better understanding of the feedback, but don't start defending yourself.

As the organizer of the evaluation session, you could think of possible topics yourself or ask the intended participants of the evaluation session which topics they want to discuss. Often, evaluation topics such as the following are addressed:

- Test process
- Test result (maybe also root cause analysis)
- Stakeholder involvement
- Test infrastructure
- Results from implemented improvement suggestions from previous evaluation sessions

Of course, each project or sprint requires its own list with appropriate topics/questions. If you need some inspiration you might take a look at the following sample "evaluation checklist" (Figure 1) on tmap.net[3].

[3] In Figure 1 a part of an example evaluation checklist is shown. For the complete list refer to http://tmap.net/downloads. If you're interested in checklists for other topics like internet testing or mobile apps testing, these can be found on the same web page.



CHECKLIST TEST PROCESS EVALUATION

This checklist can be used during and after completion of the test process to assess the process quality. Its use during the test process may be a good way to prevent problems.

- *Is the client sufficiently involved in the product risk analysis and test strategy?*
- *Is the test process running under pressure of time? If so, are concessions being made towards the original test strategy?*
- *Are the users involved in the test process on time?*
- *Do the users believe that the acceptance test is given sufficient time?*
- *Are concessions being made during the test process with respect to the planned test volume and intensity?*
- *Is the test team receiving support from experienced (external) advisors and test specialists?*
- *Is there sufficient subject knowledge available in the test team?*
- *Has defect handling been organized properly and does version control work adequately?*
- *Are the connections to existing systems and the organization receiving sufficient attention?*
- *Is there sufficient co-ordination with other projects?*
- *Does the prescribed test method, as established in the test handbook, provide enough information and support in the case of any problems?*
- *Is there a check whether the guidelines are being observed?*
- *Is there a procedure for possible deviations from these guidelines?*
- *Has a test plan been created according to the regulations?*
- *Is there a critical end date when testing must have been completed?*
- *Has the test plan been specified in greater detail and/or maintained during the execution of the test?*
- *Are the requested supporting aspects (tools, appliances, et cetera) being made available in sufficient quantities and on time?*
- *Is it possible to complete the intended test plan within the limits set?*
- *Is the test plan clear and can it be executed by the members of the test team?*

Figure 1 Part of an evaluation checklist.

When you as test manager want to focus merely on test process improvement you could choose from the various test improvement approaches described in the ISTQB Advanced Test Manager and Expert Improving the Test Process syllabi, such as CTP, IDEAL, TMMi, and TPI NEXT.[4]

[4] In the book Test Maturity Model integration (TMMi) - Guidelines for Test Process Improvement written by van Veenendaal and Wells, you'll find an improvement model largely based on the CMMi model. Or, you could use the approach as described in the book TPI NEXT - Business Driven Test Process Improvement written by van Ewijk and others. In this approach, key areas, maturity levels, checkpoints together with clusters and enablers are concepts which are used when identifying improvement suggestions. Both TMMi and TPI NEXT have developed an approach aimed towards use in agile environments. The book Critical Testing Processes by Black provides a non-prescriptive (i.e., business-oriented rather than maturity-model-oriented) framework for test process improvement as well.

Organizing and executing the retrospective is one thing, but delivering outcomes that result in actual change is another thing. Successful retrospectives involve the following:

- Keep track of history
  Use a library of data (e.g., a Wiki), reports, and minutes from past retrospective meetings, which could serve as readily available information (for instance, what were the improvement suggestions and what were the results after implementing these?) as a resource for future retrospectives.
- Eliminate the cause of the problems
  Identify the source of the problems encountered and suggest solutions that likely will eliminate the causes of the problems in the future.
- Make improvement suggestions visible
  A pitfall I often encounter in practice is that a retrospective is executed as planned and improvement suggestions are identified, but they don't get implemented. Besides the obvious prerequisite "management commitment" as described in the next bullet, it could help to visualize the identified improvement suggestions. In scrum environments I see improvement suggestions translated into user stories which are not only visible to everyone but also can be taken into account when executing planning poker.
- Get management commitment
  Implementing improvement suggestions often consumes time and resources, so make sure you've got management commitment and support to implement these suggestions