SHAF
.COM

Getting the best out of testing for management

By Shafiq Ahmed

Introduction

Over the life of my testing career, I have come across management, including C level executives, VPs, etc. who misunderstand the test role (or the test function). The misunderstanding tends to lead to poor decision making at the project level and the test function treated as an unimportant activity, compared with development, can hamper the project with unplanned delays and poor quality.

The test function should be seen as an independent auditing function throughout a project's lifecycle, from beginning to end.

From my professional experience, where a test function was involved early at the start of a project, the projects tended to run smoothly with issues captured early and project replanned early to avoid nasty surprises towards the end.

The following points are not exhaustive or complete, that need to be considered by management to instil a culture that enables an organisation's test function to succeed. I use the term management loosely to include anyone with authority in an organisation to make changes to the testing and software development functions. A note of caution: implementation of a test policy for the organisation needs to be developed by professional test engineers with input from other disciplines.

Involve testers at the beginning of projects

a. Allows test engineers to ask questions about timescales so that they can give a realistic estimate as to the level of testing can be delivered.

b. Testers would be acting as auditors/reviewers from design, implementation through to delivery by reducing cumulative mistakes at each stage.

c. Testers can identify the skills needed to test the project before the project will start.

d. Allows the project manager to ask the test engineers how certain development methodologies can impact the testing and pick the right development methodology.

e. Including the test engineer at the beginning of the project allows them to absorb the information slowly rather than having to interrupt developers and other team members midway of the project when everyone is busy. Not allowing testers to start at the beginning of the project can lead to stress and mistakes will happen.

f. Earlier the problems are found, the cheaper it is to fix them and allow enough time to re-plan if needed.

g. Even a small project requires a test person at the start of the project because one mistake can cause the project to run over budget.

Don’t tell the testers on how to do the testing

a. You will annoy the test engineer (yes, you will!).

b. You could be applying the wrong test strategy.

c. You will be stifling the creative thought process of a test engineer.

d. Test engineers have different test approaches that need to be respected.

e. There is no point in having a test expert and then telling him/her how to do their job.

Don’t silo the test function

a. Encourage developers, testers, UX, analysts, hardware engineers, etc. to collaborate freely to enable them to deliver a quality project.

b. Testers can ask developers questions on how to find faults in the software using developers knowledge of the internal workings of the software.

c. Allow UX/design people to interact with testers so testers can ask questions on how the software should behave under certain conditions.

d. Siloing the test function from other disciplines will lead to miscommunication and mistakes.

Don’t use test engineers for politics

a. It is quite easy to use the testers to use them as a stick against developers.

b. Don’t blame testers for holding back a project because the defect count is too high.

There is no one size fits all test strategy

a. Every project is different in size and complexity. An experienced test engineer can define the correct strategy given the circumstances.

b. Software evolves over time when features are added which may require a test approach rethink.

Allow testers to voice their concerns

a. Testers spot things that other engineers might not be able to.

b. Testers are passionate about delivering a quality project and need the freedom to say what is wrong without being seen as negative.

Don’t force testers to decide whether to raise an issue or not during testing

a. Encourage test engineers to raise issues and then triage them later to avoid the tester from spending a lot of time making judgement calls during testing. At the triaging meeting, commercial or project decisions should be made on issues raised by the test engineer.

b. Encourage test engineers to raise issues even if they are unsure and allow them to be wrong.

There is more to testing than test cases, expected results and detailed steps

a. Test engineers can use different approaches that make use of charters, checklists, ad-hoc testing, mind maps, etc.

b. Test engineers can find usability and performance issues by using different combination of test techniques.

c. Testing is an art as well as science. There are elements of system 1 thinking that can not be precisely written down similar to driving a car from A to B. After some practice, driving a car becomes natural and you don't spend a lot of time thinking about braking, pressing the clutch, changing gears, etc. Having to describe your journey in detailed steps will be time consuming. Some project managers make the mistake by getting testers to write every single detail for tests which can take a lot of time to write down.

d. Mechanising the test process using pre-scripted tests can lead to problems such as false confidence in the results and issues not being found. Having all test cases pass does not mean the product is fit for purpose. Pre-scripted testing should be one part of an overall test strategy.

Automation is not the silver bullet

a. Automation will not find usability issues.

b. Automated tests tend to be self-contained and have setup and teardown before and after running the test and not representative of real-world use.

c. You will not get the “feel” for the quality of the software under test.

d. Don't use automation if the payback is going to be low number of test runs versus the time to develop automated tests.

e. You cannot reliably test GUI interfaces.

Use automation where it makes sense as part of the overall test strategy

a. Use automation for repetition of certain tests.

b. Use automation to semi-automate tests to make manual test execution quicker.

c. Use automation for checking purposes to ensure functionality is present and working.

d. Use automation to test API calls.

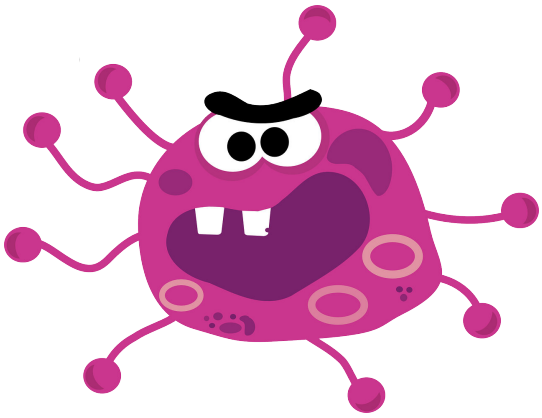
e. Use automation to create test data and create tools to make testing easier or improve test coverage.

Don’t make testers compromise on integrity

a. Testers have to give an honest appraisal of the software at a given point in time.

b. It is important to set a good example, otherwise, you will end up with testers hiding their mistakes from you!

Don’t punish testers if a bug slips through



a. When there are lots of errors in the software/hardware, it is very easy for testers to get focused on a cluster of defects and run out of time to do testing in other areas.

b. Punishing test engineers for mistakes will make it less likely they will own up to mistakes.

c. If mistakes are made, establish what went wrong and learn from them.

d. There will always be bugs in the system that are never found during testing and may crop up after the project is delivered. This is understandable as there is a finite amount of time versus quality that needs to be delivered.

Don’t leave the test strategy role to a non-tester

a. Non-testers may be experienced in their role, but will not have the same experience as a test engineer in the way they would apply a test strategy.

b. Non-testers are most likely to miss things as they are too focused on the implementation or other aspects of their expertise.

c. Non-testers don’t have the necessary test experience that is built up over the years.

d. Non-testers will not have up to date testing methods and techniques.

Don’t have meetings discussing test strategy without a professional test person present

a. Yes, this does happen!

b. Just don’t do this!

c. This is a recipe for miscommunication and can be easily avoided.

Get testers involved when negotiating outsourcing of projects

a. Questions can be asked as to how much testing has been done by that particular vendor.

b. Potential risk areas can be identified when integrating various software/system components.

c. Identify any additional testing that needs to be done to validate the vendor's claims.

Scan QR code or goto
www.shaf.com/contact to contact me

www.shaf.com

SHAF
.COM