TELLTALE SIGNS THAT YOUR SOFTWARE NEEDS TO IMPROVE QUALITY

WHITEPAPER

Software quality is one of the main priorities of every software organization but many of them do not produce quality software. Producing quality software requires a lot of discipline and the development organization needs to work with customers closely and align their specifications to the user needs.

There are a number of telltale phrases that are a definite give away as to the quality of the software such as 'that's a feature, not a bug', 'we've lost the source code', 'we're too busy to document that', 'it must be a hardware problem', 'it works in my system', 'no one will ever notice', and many other funnier ones. These phrases are done to death in reality and this can be handled by re-aligning the quality culture of the organization.

Over and above this, there are a lot of real-world challenges that every software organization deals with. Our intention in this paper is to cover these challenges and offer practical solutions towards handling those. We have kept the structure of the paper in Question and Answer format, and our answers are based on experiences gained from working on multiple projects related to quality.

There was an instance, when we developed software for our customer that met their specifications and perception expectations. We were very proud of this release initially. However, it failed the outcome expectation of our customers, which was greater adoption by their user community. They didn't blame us for this though we felt that we should have somehow done a better job of it and met the outcome expectation of our customer as well.

You are absolutely right in thinking so, as your customer's success is what makes your job easier and will let your business grow. As I see it, essentially this is a requirements problem, where you cannot take what a user is requesting at face value. Users would typically tell you that the sky is falling and they need to immediately 'Get-A-Fix'. This may result in you building applications that will never be used in practice. In order to avoid this, your requirements gathering should include talking to multiple users and documenting work flows to understand the pain points. This will help you in actually figuring out the problem for which they need a solution as opposed to the problem that is perceived by the users.

We as an organization keep having conversations on speed vs. quality as the customers typically insist on faster time-to-market. How do we go about it?

This is a very generic question and my answer would be - it depends. Speed with quality is what everyone prefers, but practically this is sometimes hard to achieve. The general preference should be towards quality, as rushing to meet deadlines without a quality focus will mean more time in the long run, as you fix bugs and have to correct wrong assumptions. There's a saying in software engineering 'fast, cheap and good', pick two of these attributes.

I can actually use this pitch of 'pick two of these attributes' with my customers as well. Having said that, how much of quality do you think is good for me? What percentage of reliability should I strive for?

Software development will be more expensive if you under perform on quality guidance and the same can be true when you over perform on quality. For instance, every extra '9' in 99.99 reliability is going to be expensive in terms of design, architecture as well as testing. Typically, the reliability and cost graph becomes exponential at this level. For all you know, the customer might be happy with 99% reliability or for that matter 95% reliability and would have a likely price that is agreeable. Hence, it becomes important to understand the customer's perception and expectation on quality and stick to that without over promising on quality.

Our last release to the customer had a number of quality issues. We have the next release in 3 weeks' time. How do we ensure a reliable release here?

3 weeks is too short a time for you to make any substantial changes to your process of ensuring better quality. Hence talk to the client and see if they would accept a revised schedule of problem fixes first, until a stable platform is established. Have your timelines extended for newer functionality. Introducing new features on an unstable platform may be setting you up for more problems.

Here is a related question to my last one. Our last release has been a cause of a problem. We introduced newer features and some of the older features aren't working properly anymore. We are in a fix to set this right. Regression testing is a problem here. QA needs to do a better job of ensuring that new additions, be it resolutions or new features do not cause any issues to existing functionality. The best option you have will be regressing the software back to stable build and start adding new features one after the other and determine if those additions are causing the software to be unstable.

We did a software upgrade recently for the customer and the customer is complaining about quality. QA and developers don't see eye-to-eye on this. Developers do not have much understanding of the customer's business while the business analysts are new to the product. How do we address this problem and we are short on time?

This is an interesting set of inter-related questions. This problem is more common than one would imagine because we don't typically have the right team doing the right job. One needs to hire developers who can understand the business side of things, in addition to their familiarity with tools and technologies. Also, a BA need not understand your product; instead the BA should understand the customer's problem and break them down into smaller components that are easily understood by the development organization.

Looks like, here your developers do not understand the requirements and are developing based on their assumptions, while QA is trying to validate the original business requirements. It precisely sounds like that your developers do not have workable, testable software requirements.
Developers should know the product and the business analysts should know the customer's needs. They both ought to work closely, instead of just translating the requirements to paper and left to the developers to develop them as we are talking about short duration of time.

Quality has to be built in the product and it cannot be tested in the product. Hence, developers pointing fingers at QA for quality doesn't really make sense. QA's job is not to add quality, but to verify the work the developers have done against specifications. Developers need to be responsible for the quality.

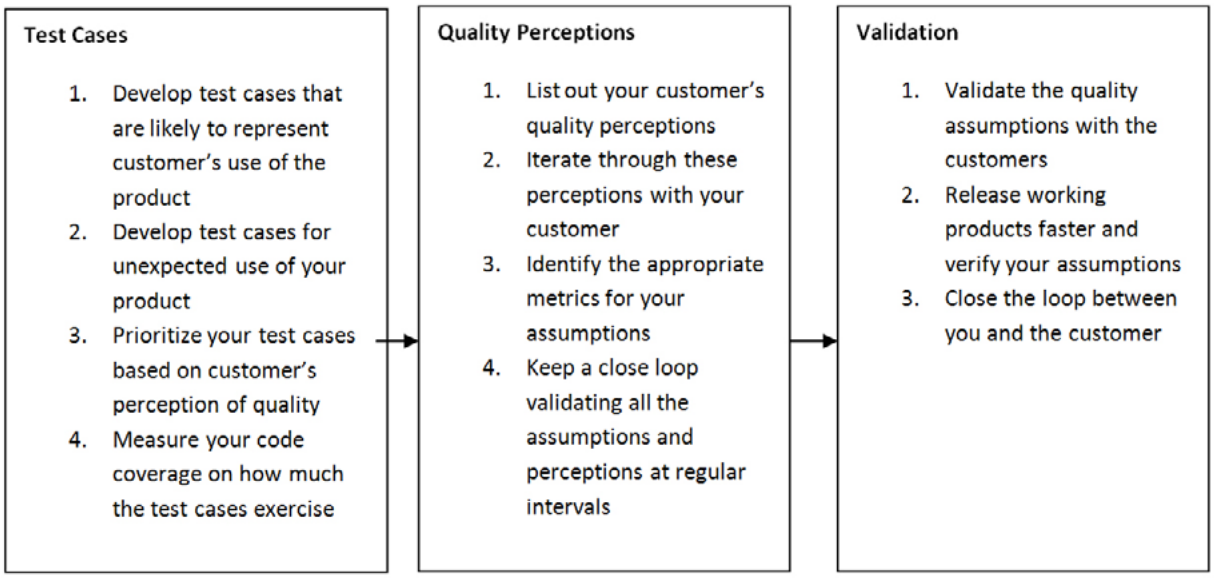
One has to move away from 'did you finish your functionality on time' mentality. If this is all you are interested in, then your developers will focus only on functionality for that is all you care about.

Few months back, we lost a project on pricing to one of our competition. When we asked our account manager on why we weren't flexible on the pricing, he mentioned that he understood our costs well and it didn't make sense to provide sub-standard software or to do it at a loss. This I guess is a valuable input for the whole organization and a fantastic perspective on quality. I would like your views on this.

I'd love to work with such account managers who have the ability to understand the sanctity of costs in development. More often than not, account managers and business analysts agree to stiff budgets and tight deadlines. This would invariably mean cutting corners and meeting the deadlines. The resulting product will certainly have errors in the code, workflow or in the business logic. This would mean failed projects and probably more dollars down the drain in getting it fixed.

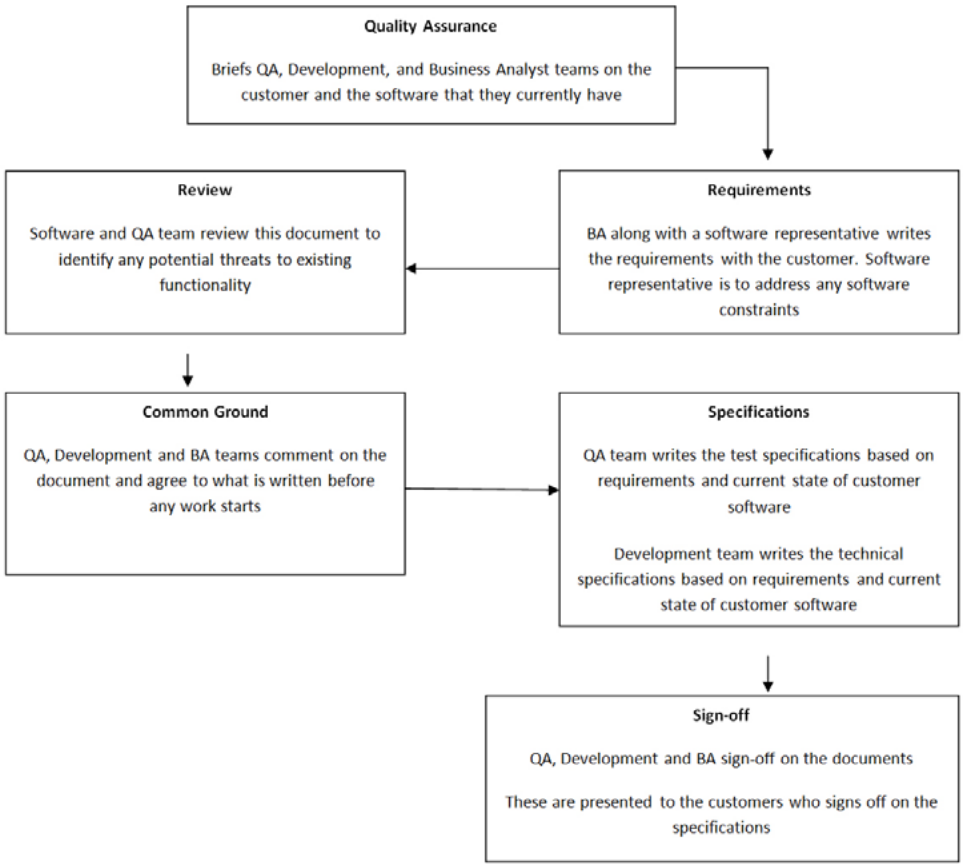
I would like to know as to how I can measure the quality level of my product before I release it to the market?

Quality is in the eyes of the customer. The best way to measure is to have your customers test and use the product the way they would on a daily basis. This is easier said than done, unless they are willing to participate during the beta. However, the best you can do should be to do your test cases, understand the quality perceptions of your customers and validate them before releasing it to the market.



Very often we keep getting change requests from customers. We prioritize them and keep building them, and at times, we even cater to the loudest customer's feature requests early and the process followed is pretty much ad hoc in nature. How do I structure my software delivery upgrade considering that it satisfies my customer's quality needs?

I will list an approach that you can use for your needs. This can be summed up as change management process as well and this would help in delivering reliable and high quality solutions



CONCLUSION.

Software quality is not the responsibility of the QA organization alone; instead it is the responsibility of the entire organization. Quality needs to be looked at from the perspective of end users by understanding their quality perceptions and expectations and matching them in the software. This is what would improve and increase the adoption rate of the software, providing tangible benefits to the user community.

Quality is a science and there needs to be collaborative effort among all the stakeholders towards making it a reality with clearly defined metrics. Essentially the organization delivering the software and the customers using the software should be on the same page in terms of expectations and benefits that they are looking at deriving from the software.

ABOUT ZADO

Zado is a provider of test automation solutions with specific focus on web, mobile and cloud applications. Our framework-driven approach to test automation ensures reliability and performance of your applications in diverse environments and complexities.

Our Center of Excellence works towards ensuring the success of every test automation initiative of our customers, irrespective of the stage that they are in – startup, transitional or mature. We have successfully helped startup, ecommerce and Independent Software Vendors with their automation needs. Our goal is to ensure quality of your software using test automation optimally.

We are open to doing POCs and Pilots that prove our credibility. We also have an innovative engagement model, Enhance – Optimize – Transfer (EOT), where we implement automation testing and transition it to your local teams. Our points of intervention after that, will be only towards enhancing the automation framework.

Zado automation frameworks help manual testers write their own test scripts without the necessary automation expertise. This qualifies manual testers into automation testers, providing better economies of scale and faster ROI of your automation efforts.

ZADO
Infotech Solutions

Zado Infotech Solutions India Pvt Ltd
85/7 IIIrd Floor Orchid Plaza
Razaak Garden Road , Arumbakkam, Chennai - 106.

contact@zado-tech.com
www.zado-tech.com