

View Point



Maximizing QA Performance

An Inexpensive Approach Towards Improving QA Organization

Abstract

If you find that your QA team has hit a ceiling in testing performance, chances are that you may be overlooking some environmental factors which are limiting your team's ability to perform better. This document highlights some of these factors which may help maximize the performance of your QA teams.

An important element of any IT strategy is to ensure deployment of defect free systems. Among other benefits, this helps minimize significantly the total cost of ownership (TCO) of applications. However, organizations quickly discover that despite their best intentions and efforts, their QA team has hit a ceiling from a defect leakage standpoint. It seems as if an invisible hurdle is preventing the QA team from achieving its true potential - deploying defect free systems.

“So what are we not seeing in this picture?”

While tool based automation or customized test automation is definitely beneficial in lowering defects, it is an option which requires significant investments. Hence organizations, with their budgets stretched thin, are constantly on the lookout for inexpensive options. Maximizing the performance of existing QA teams is often the ‘low hanging fruit’. In an attempt to squeeze the maximum out of the QA teams, organizations often overlook a few extraneous factors that play havoc with the team’s testing effectiveness. Without an eye for these factors, they can go unnoticed and continue to bleed the organization.

Based on our extensive and varied experience in test engagements, we have identified a few factors which we believe can help QA teams unlock their true potential. Of course before we start we would like to quote an old saying, “To improve the condition we need to first start measuring it”. In line with this quote, we recommend organizations begin by collecting metrics that indicate the current health of their QA programs. Most companies miss this basic, but critical point in improving processes. Once we have measured the current state of the QA programs, we are all set to further analyze the shortcomings and maximize the QA performance of teams -

How many releases does the application go through in a year?

This provides a good indication of whether sufficient time is being allocated for every QA iteration. Across engagements we have seen clients plan from as many as 12 releases in a year to as few as 3 in a year. It is essential for IT teams to plan the number of releases in combination with business teams, such that only releases which are necessitated due to business needs are included. Allowing the QA team to ‘huddle and regroup’, before the next iteration, is extremely important and the time needed for QA iterations must not to be viewed solely as test script execution time. Root cause analysis, test script updates based on requirements in the beginning, and UAT result analysis at the end of the test cycle are crucial steps in the QA process too and demand equal attention. Through these steps the QA team is able to refine its processes and prevent a recurrence of defects. Frequent releases compromise the effectiveness of these crucial steps and curtail the feedback loop, limiting the improvement in defect leakage. Hence, every application should have an optimum number of releases in a year and this should not be exceeded.

The best way to avoid having more than the optimal number of releases in a year, is to educate the business team on the pros and cons of increasing or decreasing the release frequency. It is absolutely essential that the business and QA team agree upon a plan keeping in consideration the resources available to the QA team.

How many ‘code drops’ are allowed per release?

Often frequent changes to the requirements result in increased instances of ‘code drops’ within a release. At times the frequency of code drops are not even planned making it almost impossible for QA teams to effectively test before a fresh code drop occurs. The solution lies in defining requirements freeze dates, post which no new change requests should be accommodated.

Is there a large variation in the size of every release?

The frequency of a release by itself will not set off any alarms and it needs to be considered in conjunction with the size of the release. The size of the release directly impacts the number of enhancements to be made to test scripts, some of them permanent, which need to be diligently incorporated into the QA plan. Further, releases which modify the base or common functionalities of a system need to be viewed as large releases even if their enhancement time is not large. This is because such releases have sweeping implications on the system and improper assessment of the size of the release from a QA standpoint and can lead to heightened levels of defect leakage. Further, variations in size of the release leads to variations in the size of the QA team which makes effective knowledge management a huge challenge. Hence we strongly recommend that business and IT teams plan uniform sized releases as much as possible. This can be done by prioritizing and limiting the change requests (CRs) which will help in reducing defect leakage rates and better management of the QA team.

How process oriented is the development team and how early is the QA team engaged in the SDLC?

While the defect detection during the testing phase is the onus of the QA team, the team in charge of the development also plays a significant role in the performance of the QA team. Clear sign-offs during the development process and waterfall execution of lifecycles help provide the QA team unambiguous points of entry into the development process. Further even though the role of QA team starts much later in SDLC, the involvement of QA teams in the requirement phase can make them far more effective. The distinct advantage gained by having a better view of the system enhancements to be built, allows the QA team to plan their tasks more effectively

Are you providing the QA team with an interference free environment?

QA teams need environments; viz. independent servers, database instance, test data, etc. that demand investments on a continuous basis from the organization. However organizations are faced with the twin dilemma of reducing costs and simultaneously ensuring that sufficient investments in IT infrastructure have been made. In the bargain QA teams are not always provided with an independent environment for testing. Dedicated test environments ensure that the QA execution plans are not upset and testing proceeds without interference from external factors.

Have you staffed the QA team with the right onsite-offshore leverage?

In a highly globalized world where offshore teams are a given, the onsite-offshore ratio for a QA team determines the efficiency of the QA processes. Maintaining a low onsite presence creates bottlenecks. Consequently the reviews of the enhancement documents (or change requests), test script vetting with the UAT team, etc. suffer. At the same time, a big onsite team may not always help realise the cost benefits of an offshore model either. Hence it is recommended that QA teams begin with a 25-30% onsite staff and gradually target aggressive onsite ratios as low as 15-20%.

How well do you manage knowledge erosion?

This is especially important for a large and complex application. QA teams often struggle because of knowledge erosion associated with planned or unplanned attrition in team. In teams where staffing size fluctuates either due to varying size of releases or due to attrition, it is important to retain a core group of experts as the base team. An extended team, built around this base group, can then be engaged or released depending on the demands of the project. Of course, while such a model would help manage knowledge erosion to a large extent, it would need to be bolstered further by strong Knowledge Management (KM) practices

Finally, while the above mentioned factors can help improve QA performance, it is important to track the QA program using objective metrics. Some of the metrics which we believe need to be tracked closely are:

- Defect Leakage: This measures the capability of the QA team in detecting defects/bugs. It is important to track this number accurately, assigning specific reasons for each defect. All issues reported during QA may not necessarily be defects. Categorizing the defects as “Valid Defects”, “Not a Defect”, “Duplicate”, “Changes in Specifications” and “Recommendations” help identify the true performance of the QA team. The unsaid part, of course, is that there needs to be a robust defect tracking system already in place for formal recording, tracking and subsequent analysis.
- Release frequency: This indicates the number of releases that an application is subjected to in a normal year.
- Code drops per release: Refers to the number of builds during a release.
- Schedule/Effort Adherence: Measures how well the QA team is performing vis-à-vis the planned milestones and bandwidth of resources.
- ‘Not-a-bug’ percentage: It indicates the number of defects logged by the QA team that are not defects, as a percentage of the total reported defects. This metric is very important since it is an indication of the unproductive effort expended by the development team in investigating a non-defect. It is also an indication of the need to update the documentation / improve test scripts

The environmental factors discussed in this document, while hidden from the immediate field of view of the team, greatly influence the outcome of the QA efforts. Hence they have to be objectively assessed and tracked to understand the influence they exercise on QA performance. By learning from them and adopting appropriate measures, QA teams can be helped to realize their true performance potential.

About the Authors

Shishank Gupta
(shishankg@infosys.com)

is a Delivery Manager at Infosys Limited responsible for providing independent validation services to Infosys clients from the Retail segment. Of his 14 years of IT experience, the last 10 years have been in Independent Validation. He holds a patent for his estimation model for testing projects and has also authored multiple papers on test optimization.

Rajeev Wahi
(Rajeev_wahi@infosys.com)

is an Engagement Manager with Retail, CPG and Logistics unit at Infosys. He has over 15 years of business and IT experience. Rajeev has successfully managed client relationships in South East Asia, Europe and USA.

Sreenivas Madugula
(Sreenivas_Madugula@infosys.com)

is a Project Manager at Infosys Ltd. He has over 14 years of IT experience across various technologies like embedded systems, Microsoft Web and Java Enterprise Solutions. His area of specialization is implementation of e-commerce applications.

Contributions by

Sreejith Kolore
Business Manager - Retail, CPG and Logistics
Infosys ltd.

Nandakumar Lakshmanan
Test Manager
Infosys Ltd.



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.