REPORT ON SOFTWARE QUALITY

The State of Software Quality Assurance

by
CHRIS DURAND
CTO, Bridge360

Software development practices have evolved substantially over the last few decades. Until recently, quality assurance practices have been advancing more slowly, relying primarily on an ever-growing army of testers to execute test scripts manually. Out-of-control testing costs, increasing software complexity and many companies adding additional platforms (mobile), are greatly responsible for the lost focus on quality standards in recent years. It's a trend that should not be tolerated by companies, and certainly will not be tolerated by consumers.

There is good news, however—there is a resurgence of focus on quality assurance standards and testing in that project budgets are rising (as much as 50%-75%[1]) to support user satisfaction with software product deliveries. Companies more and more are investing in their development strategies to benefit from newly developed best practices and to assuage the negative risk that would damage their reputation and brand. There are four key areas of software development practices which companies need adopt in order for their software projects to achieve optimal results.

[1]Bernie Gauf, Innovative Defense Technologies LLC, Quest North America 2014 keynote address "Software Testing in a Reduced Budget Climate," April 9, 2014.

1

TEST AUTOMATION ADVANCEMENTS

Test automation has been around for many years, albeit the focus was only at the enterprise level in Fortune 500 companies such as IBM, HP or Texas Instruments. Formerly, programmers would write software code without concern for testing it. Test automation professionals would rely on expensive GUI (Graphical User Interface) automation tools to provide impressive results once set up and maintained. Unfortunately, these tests remained sensitive to updates and changes, and this led to brittle, expensive-to-maintain tests.

Test automation is no longer the sole responsibility of the quality assurance team.

Those days are gone.

Test automation is no longer the sole responsibility of the quality assurance team. Companies have realized that efficient, cost-effective test automation relies on architecting software for testability. Software must be built in a more modular manner

- ***Software must have modules containing clearly defined APIs (Application Programming Interface) that can be quickly and easily tested.***
- ***API testing, which is simple to implement using less expensive tools, results in more comprehensive tests that can and should be integrated into the automated build processes.***

This trend will only accelerate as companies seek to reign in their product costs for software testing while accelerating the delivery of higher quality than before. Quality Assurance (QA) professionals must have test automation skills and be proficient in basic scripting knowledge or risk being left behind. Development teams must start architecting applications for testability today and integrate it wherever possible to high-value areas of existing applications.

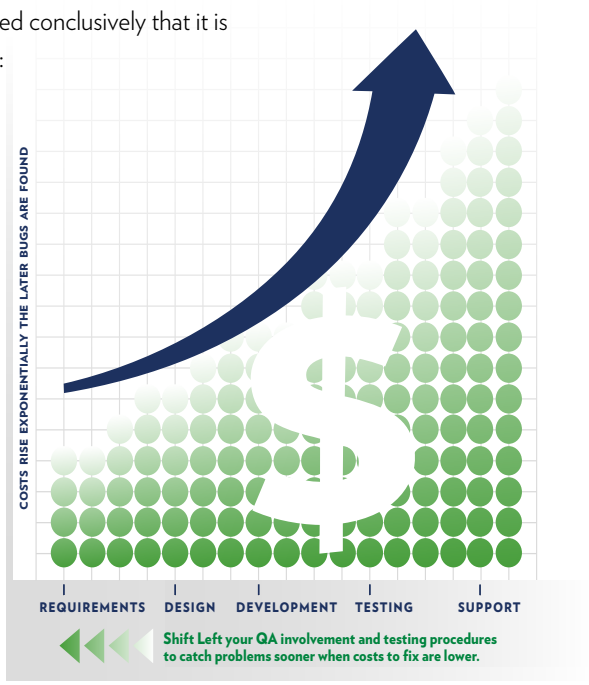
2

"SHIFT LEFT"

The traditional split between "programmers" and "testers" is rapidly blurring. At one time, business analysts wrote requirements, developers coded them, and testers validated that the software met documented specifications. Decades of experience in software development have demonstrated conclusively that it is difficult to get complete requirements at the start of projects, which only leads to:

- ***Multiple change requests***
- ***Rework***
- ***Ever-increasing costs***

Companies also risk implementing functionality that made sense a year earlier when the requirements were written, but is now obsolete or unnecessary, thereby increasing costs.



Companies have discovered the value of getting testers (and programmers) more involved in the up-front requirements and planning phases of projects instead of bringing them in once the architecture has been settled and the code has been mostly written. Collaboration and involvement earlier in the process allows QA professionals to bring unique perspectives to requirements gathering. QA is trained to identify unclear and ambiguous requirements and can be constant advocates for writing clear, testable requirements and ensuring the application is architected for testability. The QA industry refers to this earlier involvement as "shifting left" (imagine a project Gantt chart where the QA tasks now start farther to the left than before).

Requirements defects are among the most expensive to fix late in the project, therefore, companies looking to reduce software development costs have QA shifting left to catch these issues earlier. Such investments at the front end of a product's lifecycle help mitigate the potential for exorbitant costs in later stages of the development, and often produce greater returns by resulting in superior product quality. Agile development practices advocate the entire team shifting left, but companies can still reap the benefits of shifting left regardless of the software development methodology in use.

3

COLLABORATIVE AND RISK-BASED TESTING

In the past, QA often focused on black box testing, where the QA team tested an application via its external interfaces with little regard to how the software worked internally. Part of the allure of this approach is that without knowledge of the internal working of the software, the QA effort will be more comprehensive. The QA team will not be tempted to neglect testing certain areas of the software and thus miss issues because they thought they understood how something worked by talking to the developer. For companies that can afford it, relying strictly on black box testing is a viable option. But the overall amount of testing required can be prohibitively expensive and resource-intensive.

...few companies have the budgets or scheduling capability to test as much as they might desire.

Not surprisingly, testing less often can lead to substantial cost savings. Companies today are using more collaborative approaches between programmers and testers to assess the quality of their applications while eliminating duplicate or low value testing. With well-written, thorough unit tests for particular functions, testers can validate that unit testing is sufficient, and they can avoid recreating functional tests that duplicate unit test functionality. In situations where user acceptance testing merely re-executes the same test cases that were previously executed, an opportunity for reduced costs is presented.

By leveraging risk-based testing approaches, companies can allocate scarce testing resources to areas of greatest importance. Rarely does a spelling error on a form cause as much business impact as a bug involving financial calculations, so testing resources should be directed appropriately.

These testing techniques are proven methods for driving greater efficiencies. Despite the trepidation often associated with less testing, few companies have the budgets or scheduling capability to test as much as they might desire. Collaborative and risk-based testing methods help companies invest scarce resources more efficiently. Implementing such methods may require a major cultural change within an organization, but substantial benefits are often realized. This approach is becoming more commonplace, because companies and QA teams are being asked to do more with less.

4

METRICS

William Edwards Deming, an acclaimed statistician, professor, author, lecturer and recipient of the 1987 National Medal of Technology, expressed his value of metrics by saying, "you can't manage what you can't measure."

"Without metrics you are just another person with a different opinion."

Most companies collect metrics, but few are maximizing the benefits. To do so, they must learn how to interpret them properly and leverage them as a catalyst for improvement. For development and testing teams to benefit from the metrics they collect, it is vitally important that they gather the right metrics and understand how to interpret and use them.

Savvy companies are tracking metrics (or, Key Performances Indicators [KPIs]) that answer specific questions tied to business outcomes. They do not track metrics merely for the sake of tracking metrics. An excellent model for identifying useful metrics is the Goal-Question-Metric (or GQM) model.

Companies can start by identifying specific Goals for given applications (e.g. reducing the cost of software development). They can then decide what Questions they need to ask to determine if goals are being met (e.g. does the application have lots of technical debt?). Finally, they can collect Metrics to answer those questions (e.g. what is the defect recurrence rate?) and translate the results into actions (invest in refactoring the application so it becomes easier to maintain).

Good metrics can be translated into business results, which eliminates finger-pointing and guesswork by providing an objective window into what is really happening in the application.

Software development is a complex process, and companies that use metrics effectively are better able to leverage their investments in software testing. At this year's QUEST North America Conference and Expo, speaker Michael Mah, managing partner at QSM Associates Inc., emphasized the importance of metrics this way, "without metrics you are just another person with a different opinion."

RECOMMENDATIONS

There are many ways to make an impact on your organization's QA practices. To get started, here are several places to quickly affect such change:

- Make automated testing a requirement. Teams will need time to get there, but the benefits are clear. Testability is just as important as security, performance, and scalability for modern applications.
- Add 'quality' to the developer job descriptions and goals. This ensures everyone on the development team knows they are responsible for quality, not just your test team. Failures in production are a team failure, not the failure of an individual in development or QA. Missing a delivery deadline because QA is not complete is a team problem, not a QA problem.
- Eliminate unnecessary or duplicated work. Get development and test working more closely together to eliminate overlaps in testing. Test less, but don't shirk on quality.
- Use metrics. Find metrics that can be measured and will be useful. A good one to start with is tracking the root cause for bugs (poor requirements, developer mistake, deployment problem, etc.).

SUMMARY

QA teams and organizations have made huge strides over the last few years in their quest to improve software and business outcomes. No longer do QA teams serve as outsiders to the software development process, merely reporting back their findings. Strong QA teams are working side-by-side with other stakeholders to determine, as a team...

- ***How to efficiently automate applications***
- ***How to define solid requirements, and***
- ***How to divide the testing effort efficiently among team members***

They are also measuring their results along the way and making constant improvements based on these metrics. Often such approaches require substantial organizational and cultural changes to achieve, but many QA teams have wisely concluded that the results are worth the effort involved.

ABOUT BRIDGE360:

Bridge360 is a software consulting company that is leading the mission to bring quality back to software, implementing quality solutions for clients throughout the entire software development lifecycle. This includes program management, architecture, design and development, testing and support, and maintenance. By specializing in solving complex problems at every phase of the project, Bridge360 removes roadblocks to bring clients' software and applications to any market with the highest level of quality.