rainforest

90 Days to Better QA

The Strategic Guide to Kickstarting Your QA Process

Software defects cost the US economy $59.5 billion every year. Nearly half of those costs could have been prevented by better testing. [1]

Software defects cost organizations $1.1 trillion in business valuation last year.[2] *QA testing budgets have fallen for two years in a row.* [3]

Software quality is inextricably tied to business success today. It's not good enough to conceive of innovative features or deliver them quickly. Organizations must also be able to execute them well. Unfortunately, many organizations spend so much time on engineering with ingenuity and speed that they leave the principles of quality assurance in the dust.

In many cases businesses can get by in the short term in this manner, and may even achieve a high degree of success due to the "first-mover" advantage. But as the customer base scales up, the product grows more complex and the market expects more maturity from the offerings, the cracks in this veneer eventually start to show.

These cracks typically manifest themselves in a plethora of higher-order business problems. These could be customer complaints, site outages, high instances of abandoned orders— the list goes on and on. At their root these problems usually stem from one or more major categories of common quality woes:

- ☐ Inadequate testing coverage
- ☐ Poor performance
- ☐ Test processes impeding engineering velocity

Some organizations may only face troublesome issues in one category. In the most dysfunctional cases, an organization may suffer chronic issues in all three. Many of these issues can seem daunting to address at first glance— particularly when business problems are layered on multiple technical and process issues.

But with proper investigation, solid planning and steady execution, it's possible for engineering leaders to make big gains in as little as a few months. Sure, you won't chip away at everything in one fiscal quarter, but by targeting one or two major problems you can make a difference.

In this guide, we'll explore what it takes to put your QA process on track to meet your quality goals in just 90 days.

[1] http://www.ashireporter.org/HomeInspection/Articles/Software-Errors-Cost-U-S-Economy-59-5-Billion-Annually/740
[2] http://servicevirtualization.com/report-software-failures-cost-1-1-trillion-2016/
[3] https://www.capgemini.com/service/world-quality-report-2017-18/

rainforest

Creating a Roadmap to Better QA

Section One

"You get one shot at it and if the quality of what we're delivering doesn't create and keep those individuals happy then we've lost them." [4]

-Aaron Haehn, Director of QA and release Sprint

HOW MUCH CAN YOU ACCOMPLISH IN 90 DAYS?

The key to change is striking a balance between the most impactful short-term fixes that can address some of the most obvious business pain points in a short window, while keeping the long game in mind. A 90 day period will afford time for both planning and taking action. You'll pull the trigger on focused, short-term fixes now, while allotting time for meaningful planning and early execution of medium-term and long-term fixes.

A CHECKLIST: REALISTIC GOALS FOR A 90-DAY PLAN

- ☐ Evaluation of the current state of QA— what is good, bad or lacking?
- ☐ Establishment of business and technical goals for short-, medium- and long-term
- ☐ Identification of metrics that will measure progress toward goals
- ☐ Identification of a team of stakeholders to jumpstart this round of improvement
- ☐ Implementation of one to two new processes

By the end of the 90-day period, an organization should be able to show measurable improvements in at least one or two key areas. At the same time, they should keep in mind that day 90 is a springboard for future incremental gains. In this section, we'll delve further into an aggressive 90-day timeline for improving QA results.

Days 1-30: Assess & Monitor

The first 30 days is all about taking stock of the situation. Engineering leaders should use this time to ask as many questions as possible across key stakeholders—line-of-business application owners, front-line customer service representatives, QA staff, developers, and whoever else has a stake in the quality equation.

ASSESS THE PROBLEM

The goal is first and foremost to diagnose the biggest business problems related to QA. For example, customer complaints could be rolling in rapidly about a particular software

[1] https://www.stickyminds.com/interview/sprint-s-journey-testing-organization-quality-advocate-interview-aaron-haehn

product or mobile app to the point where it is affecting customer satisfaction numbers. Alternatively, outages and performance issues on a company's retail site might be getting to the point that cart abandonment has gone through the roof and revenue is slipping.

Once the biggest business pain points have been identified, it is time to turn the lens inward and start asking "why?" repeatedly. Cart abandonment might be caused by outages, but why are those outages happening? If it is a code quality problem, why is that an issue?

Ultimately you're digging deep to diagnose the engineering root cause to the business problem— be it technical or process-oriented, it likely comes down to some permutation of a quality issue. A truly fruitful assessment is one that tries to identify very specific QA problems, because the more generalized an identified problem is, the harder it is to make a definitive improvement to fix it. The more things are broken down, the easier it is to start to plan and measure progress.

Clearly, the crucial part of this entire assessment stage is asking the right questions. But equally important is how you find the answers. Listening to employee analysis is important, but leaders should also be asking people to use data whenever possible to back up those assertions. The more data is used to substantiate a diagnosis, the easier it will be to gain buy-in both from the top down and the bottom up as you plan and carry out your improvements.

INSTITUTE BETTER MONITORING AND METRICS

Data is crucial to backing up assessment assertions. If you find you don't have enough data to test your hypotheses about what's going wrong with tests and QA processes, now is the time to start amending that. Before you establish an improvement plan or try to execute on it, make sure the right mechanisms are in place to measure both the current deficiencies and what future success will look like.

This is essential for not only ensuring you're maximizing your efforts, but for proving the ROI of the initiative to the executive suite once you hit day 90.

Days 31-60: Plan & Execute

Once you've identified the metrics that will matter most to this improvement cycle and you've ensured that you've got mechanisms in place to reliably collect them, it's time to think about goal setting. The key here is to create achievable goals you'd like to target, both immediately and farther out in the future.

MAKE A PLAN

"Setting the right expectations for goals is so important. Otherwise, you put a process together that might be fine and which makes improvements beneficial to the business, but the goals are not achievable," says Derek Choy, CIO of Rainforest QA. "At the end of the day, if you miss the target, the team is discouraged because they worked on a whole bunch of things and still feel like they failed. From a business perspective they also feel like you failed, even if your improvements were a huge accomplishment— all because there was an unrealistic expectation at the outset."

“Collecting information is not enough. If nobody talks about it, you might as well throw it out.”

- Daria Mehra
Director of Quality Engineering, Quid

At this point, it’s time to plan short-term and long-term actions that can be taken to achieve those goals. As you establish the goals and the plans for how you’re going to achieve them, it is crucial to communicate expectations both up and down the food chain so that executives know what to expect from your progress and the people involved in the work know how their workflow will need to start changing. As you communicate upward, this is the time to use data and metrics to make justifiable budget asks or requests for reallocation of resources to achieve goals. Ideally, all stakeholders will have been involved enough in that initial assessment that their suggestions are fueling these plans, which will drive their engagement and buy-in.

START EXECUTING

The goals are made, the plans for improvements are in place and now it’s time to start executing short term fixes. This could be making one or two process or workflow changes. It could be temporarily (or permanently) reallocating someone from one team to another to either supplement manpower in a certain area or bring a new type of thinking to an old issue that has previously been a pain point. It could also mean bringing in a new technology to help streamline work in the biggest bottleneck you’ve identified through your assessment. This early execution is all about thinking small and fast. You’re looking to make one meaningful and measurable change that can both help the business now and also act as a proof-of-concept for more change moving forward

Days 61-90: Evaluate and Refine

The last month of your 90-day cycle should be dedicated to continued execution and to observation. Start looking at the initial execution and keep track of the metrics that you identified early on.

EVALUATE INITIAL EXECUTION

If necessary, refine the metrics to add context about the changes that are (or are not) happening. And, finally, don’t be afraid to share your initial successes. Look for ways to present the data clearly and transparently across the organization so that everyone knows how things are going.

REFINE A LONG-TERM PLAN

The observations and progress on the metrics will be what informs your future action. By day 90 you should be able to prove whether you met your realistic goals and to set new goals for the future. You should be able to use the data from the improvement cycle to inform suggestions for longer-term fixes and you should be able to break down those long-term fixes into other small objectives that can be accomplished in the next 90 days. As you refine the long-term plan, the data gathered will also allow the team to adjust budget and expectations for change in the future.

rainforest

Troubleshooting Common Issues

Section Two

“Software moves fast. Engineering is a really rapidly evolving field and the best skill that any developer or tester or dev ops or anybody involved in producing software can have is the ability to evolve and adapt, along with how quickly we’re changing.” [5]

-Melissa Benua, senior technical lead, mParticle

IDENTIFYING YOUR QA PAIN POINTS

So far we’ve mostly explored the management mechanisms you’ll need to put in place to carry out your 90-day QA improvement plan. Now it’s time to turn the lens toward the kinds of changes that realistically can be made during this timeframe. The types of change you’ll need to make are obviously unique to your organization and highly dependent on your assessment during the first weeks of the initiative. But these are a few examples of the types of problems that organizations tend to run into when looking for quick wins in quality.

As you look through these, be mindful of the fact that you can’t solve all of these types of issues at once. You may not even completely fix one of them in 90 days, but this timeframe should be sufficient to get started on one or two large issues and begin to show measurable improvements.

As we’ve already mentioned, the short-term initiative is also great for proof-of-concepts. Data collected from changes made to one product group, one process or one technology can suggest the most important changes that need to be made across the business moving forward.

COMMON QUALITY ISSUES

Bugs Clustered on Specific Features	*“A significant percentage of our customer complaints are about a single product.”*
QA Slows Down Development	*“Our team is consistently missing deployment deadlines, or we must sacrifice test coverage to meet deadlines.”*
Too Many Bugs on Legacy Features	*“Our team seems to need to rewrite regression tests every time the developers touch older software.”*
Users Find Bugs Before Your Team Does	*“Every time we launch a new product or feature, customers complain that our software or website seems half-baked.”*

[5] https://www.stickyminds.com/interview/how-adapt-new-age-testing-and-development-interview-melissa-benua?page=0%2C3

Pain Point: Bugs Clustered Around Features

KEY INDICATORS

You have a significant number of test cases, but issues still appear regularly. You have one or two “problem features” where bugs seem to crop up most frequently.

SOLUTION

It might seem obvious, but if an organization starts hunting around in its metrics and finds that the majority of issues revolve around a specific application, product feature or code base, then the greatest gains are going to be made if a 90-day improvement initiative focuses in on that specific software.

It doesn’t mean abandoning everything else in the interim, but instead repurposing a few extra resources and muscle to make a difference in the targeted technology.

For example, if the majority of customer complaints focus on one aspect of a product then a short improvement initiative would be well spent solely in improving quality around that aspect. It might mean borrowing product managers or some developers from other projects and reassigning them for a month or two to build up test coverage or automation around that feature.

REAL-WORLD FIX: BUILD DATA-DRIVEN COVERAGE

A big believer in data-driven quality improvement, Quentin Thomas, senior QA Automation Engineer for BleacherReport, says the best changes are made based on facts rather than feelings. In one instance, he managed to take a trove of defect data from his organization to prove that the majority of issues were coming from an old code base that was only being lightly maintained. Rather than suggesting the organization pour more resources into coverage for that code-base, the data showed that maybe it needed to be abandoned altogether.

“The data gave us the ammo as QA to say “Hey, why don’t we just consider phasing it out, because it is causing us a lot of issues and that’s better than trying to spend all this time to test and analyze this stuff,” he says. “Sometimes getting rid of a service no one is maintaining is going to do a better job of improving quality than anything QA can do.”

Pain Point: QA is Holding Up Velocity

KEY INDICATORS

Every release becomes a struggle between shipping on time and completing your QA process. Testing often doesn't begin until late in the development process.

SOLUTION

The average level of automation for all QA test activities is just 16%. [7]

If your assessment shows that QA is holding up development velocity, moving from manual to automated testing should be an overarching goal. But that is a long-term goal that takes tons of incremental improvement to achieve.

As your team works toward it, it's imperative to look for stopgap measures that improve velocity without having to wait for a year-long push to build out an automated test framework. Additionally, even already highly automated shops can find that velocity is impeded if the automated tests are extremely brittle. In that case, improving thereliability of testing and instituting stopgaps is crucial to keep the gears moving.

REAL-WORLD FIX: IMPLEMENT FUNCTIONAL TESTING EARLIER

Enterprise infrastructure management firm SolarWinds leans heavily on its real-time SaaS operations analytics solution to help clients understand what's going on with their IT data. The development team averages 20-25 pushes per day to keep Librato continuously improved. but the highly visual platform was challenging the limits of their existing automation capabilities.

"As we scaled up, we found that we had a lot of challenges with our existing automated CI tools, both in terms of how long it took to run those tests, and the overall stability of those test suites," says Matt Sanders, Director of Engineering for SolarWinds.

As a result, the organization still needed to depend on manual testing that didn't impede velocity but also kept risks minimized. They turned to Rainforest to fill the gap for on-demand testing.

"For a long time it was common that when we made changes to our visualization layer, we would roll it out only for our team, wait 2 or 3 days, and then turn it on for everyone else," Sanders says. "We tend not to do that anymore because at this point we feel like the feedback loop is fast enough that if we break something we're going to find out pretty fast. Now that we're using Rainforest as a safety net, it's more acceptable to move fast."

[7] 2017 World Quality Report, Capgemini

Pain Point: Too Many Bugs on Legacy Features

KEY INDICATORS

Your regression testing suite is large and constantly growing. Every new release seems to create new issues that center around older features. If you have automated tests, many of them are broken or unreliable.

SOLUTION

If organizations are struggling to identify problems in updates to older features due to reliability problems in regression testing, then this is where the leader may want to bite the bullet during the improvement cycle.

If there's no automated regression test suite in place, you may want to assess whether or not some of your manually executed tests would be more efficient if automated (without sacrificing accuracy). If the team already has automated regression testing in place, but that automation is flaky, it might be time to rethink the approach.

"Are you constantly getting false positives or false negatives? Are tests just breaking and needing to be rewritten any time the software is touched?" says Mehra. "If so, you don't have automation, you have technical debt."

REAL-WORLD FIX: PARTNER WITH CUSTOMER SUPPORT

According to Mehra, when she arrived at Quid her team's regression suite was running on someone's laptop and not in a continuous integration environment. It couldn't be maintained because it was written in a language no one in the company was willing to support. She needed to rebuild and decided to try implementing Rainforest QA in order to allow non-coders in the company who understood quality problems to help create the regression suite.

"I could draw on the experience of customer support folks and our commercial team, who do demos of the product and know it intimately to write tests," she says. "We helped them with test design and having a solution like Rainforest allowed us to completely sidestep the coding problem and have work flow directly from user to test."

Pain Point: Users Find Bugs Before Your Team Does

KEY INDICATORS

You're executing a number of tests in pre-production, but your customers always seem to find issues after release.

SOLUTION

If the regression suite has been developed and the biggest problems tend to be manifesting themselves in newer features, then it may be that your exploratory testing strategy needs improvement.

As you assess your problems, ask yourself who's finding the most bugs and when are they usually identified? "If you don't have an answer to that, then the answers are 'your end users' and 'after you release the software,'" Mehra says.

If your engineering organization already has a decent-sized QA team, Mehra believes the most impactful investment is is to focus on exploratory testing rather than having just them retesting old flows over and over again. But there's also no reason why a smaller team or even a large-team spread thinly can't find external resources to shore up exploratory testing. Whether doing it internally or outsourcing, for the sake of 90-day improvement, working on exploratory testing can help to indicate where longer-term QA process improvements can be made.

REAL-WORLD FIX: ROBUST EXPLORATORY TESTING

Thomas with BleacherReport has fueled reductions in defects quarter over quarter by leveraging Rainforest to incorporate fast, effective exploratory testing into their development process.

"The information gathered during exploratory testing is combined with all the defects that our team found internally and issues reported by users. We compile all that information into QA reports for each quarter," he says. "Then we can see where we need to focus our QA muscle. We are doing the same level of testing internally as before, but now it's much smarter and more focused."

Other Common QA Scenarios

While the previous pages outlined four key QA problems that many teams face, they don't account for every issue that QA teams face. Here are just a few more common scenarios that your team may experience, with some quick tips for resolving them.

DIFFICULTY IDENTIFYING SOURCE OF PERFORMANCE ISSUES

If your team is having difficulty identifying the source of outages or performance issues, then it may be necessary to dedicate improvement efforts to instituting tools that offer greater visibility into all aspects of the application stack. These are tools like PagerDuty, Datadog, and Bugsnag.

The important thing is to take the results that these monitoring tools start to surface and create processes that get stakeholders into a room talking about them. "Put diagrams up, discuss what you see on the dashboards, and get people thinking," Mehra says. "One way to do it is attach it to a periodic retrospective that the team does. Maybe you do it every two weeks or every month, allocate ten months on the agenda."

TEST ENVIRONMENT ISN'T RICH ENOUGH TO FIND BUGS

Sometimes what looks like a test coverage problem often comes down to issues with test environments or test data, because the organization is unable to execute a specific scenario in its test environment because it doesn't have the right data or the environment doesn't support that function. Even with added coverage, if the environment doesn't support executing that test case then the bugs are still going to sneak through.

This is the type of issue that can be found, diagnosed and improved through the targeted implementation of service virtualization and test data management tools.

TEST COUNT IS HIGH, BUT SO IS BUG COUNT

One mistake that engineering organizations often make is to conflate number of tests with the robustness of test coverage. But it is very possible to have a high test count and still let through an unacceptable number of bugs that impact performance or customer engagement.

In this scenario, the quality of test cases may be an issue. Or you could be facing some dependency-related problems. For example, an organization could need to spend an improvement cycle simply instituting a process where code review is never conducted in isolation. If there's a dependency between mobile and web, then both teams should have to look at that in code review to ensure that dependency issues don't slip through the cracks.

Thinking Beyond 90 Days

Section Three

GETTING CREATIVE WITH PEOPLE

"There's a strong connection between the quality of an application and the business value that application provides. As software works its way into the heart of every organization, every IT leader needs to internalize this reality. [6]

-David Chappell, technology analyst David Chappell and Associates

No matter which area you choose to target for your 90-day initiative, one thing to keep in mind is that the pull of the status quo is strong if you keep people in the same roles they've always filled. Leaders can jump start change by using the shortened window of improvement as an excuse to make some dramatic human resource reallocations that would never work politically if it was positioned as a lasting change.

For example, say test coverage is not hitting the right spots because the quality of test cases has reached a plateau. This might be a call to bring in other people from other teams who might not traditionally write test cases, such as product managers. If senior management knows that a key product manager is needed, but only for a couple of months, they're much more likely to give the green light. Similarly, if test environment problems are hindering quality, then perhaps the answer is bringing in architects from another team to help design a better provisioning process.

"We've got to be a little bit more creative from a human resource perspective and make sure that effort is focused on having the right resource rather than dictating changes to those normally responsible for something," says Choy. " Maybe I'll borrow resources from other teams to help me build out coverage for a few weeks. Or, maybe I will have my developers actually stop development for one sprint and have them focus on building databases, test coverage, automation, whatever, so I can see a significant improvement in the next month or two on a targeted problem."

It's not just a matter of getting creative with new technologies or processes, but also bringing in the outside people who can affect change at a rapid clip.

Similarly, engineering and quality leaders are more likely to magnify the results of initiatives like these if they're creative with the alliances they make outside of the traditional tech geek circle. For example, Mehra says one of the best things that QA and software engineering leaders can do is build relationships within the customer success organization.

"They've been told where the bugs are by irate users so they will talk to you at length about the fragile areas of an app and will solve half the problem for you, because you don't need to go looking for bugs they've already found," she says.

[6] http://www.davidchappell.com/writing/white_papers/The_Business_Value_of_Software_Quality-v1.0-Chappell.pdf

FINAL THOUGHTS: MAKING IMPROVEMENTS STICK

As you start your 90-day journey, it's important to keep in mind that the goal is making these improvements stick. It's very easy for an organization to slip back into bad habits if they're comfortable enough and if new processes are difficult to maintain.

It may make sense as you're drawing up objectives for your next 90-day initiative to bake in a maintenance plan for changes made in this previous improvement cycle.

"Draw up a plan of how the organization will maintain the change. Want some inspiration? Look at the service plan for your car," explains executive communication and leadership consultant Jonathan Halls. "It identifies things to check every ten thousand miles or so. Look at your change project and ask what needs to be checked and then write it down with clear instructions that are observable and measurable. Appoint someone to make sure these are checked and report back to management on the change results." [8]

But before you even start your initiative you should be thinking about the sustainability of changes. This means trying to maintain four success factors over the next three months:

- ❒ Executive buy-in
- ❒ Team buy-in
- ❒ Proof of measurable improvement
- ❒ Making process changes as painless as possible

First of all, you can't do this without executive buy-in. Senior leaders need to be engaged and convinced of the benefits. Similarly, you'll need buy-in from your team. On both ends of the spectrum this kind of buy-in is best gained through data-driven proof. Metrics will be your best friend when making the business case for change and they'll be your best friend for sustainability when they offer evidence of improvement. But cold hard facts may not always be enough--leaders who make things stick also know how to frame the metrics so that they get team carrying out changed process to understand what's in it for them.

"If you want your team to accept change and make it stick, you need to frame it in a way that will tap into people's emotions," explains transformational leadership consultant Bill Hogg. "This is why leaders have to make the concept of change real for people — it cannot exist as a conceptual theory." [9]

In this case, it might mean explaining to a team how an improvement of performance metrics shown by the 90-day push will mean fewer 3 am wake-up calls in the long-term due to outage emergencies.

Ultimately, it's important to keep organizational pain in mind when suggesting new processes. Wherever possible, these changes should be working with the normal flow of development and shouldn't contribute to technical debt in the long run. Otherwise you risk losing gains as quickly as you made them.

Key Takeaways

- ❒ Don't spread your focus too thinly: try to break down problems as specifically as possible and work incrementally
- ❒ Do gain buy-in from the team and executives
- ❒ Don't forget to look for quick fixes as well as long-term improvements
- ❒ Do expose data for optimal use
- ❒ Don't get too hung up on mindless automation
- ❒ Do communicate: manage expectations of executives and the team continuously

[8] http://jonathanhalls.com/making-organizational-change-stick/
[9] https://www.billhogg.ca/2017/08/how-transformational-leaders-make-organizational-change-stick/

About Rainforest QA

Rainforest QA helps agile and continuous delivery engineering teams move faster with the industry's only AI-powered crowdtesting platform. Our platform leverages 60,000 qualified testers to deliver on-demand, comprehensive and machine learning-verified regression test results. Rainforest customers spend less time and money testing so they can ship better applications faster.

For more information on Rainforest, visit https://www.rainforestqa.com.

Learn More

Special thanks to Daria Mehra, director of quality engineering at Quid and author of Tester's Digest, Quentin Thomas, senior QA Automation Engineer for Bleacher Report, Russell Smith, co-founder and CTO of Rainforest QA and Derek Choy, CIO of Rainforest QA, for valuable input in developing this guide.