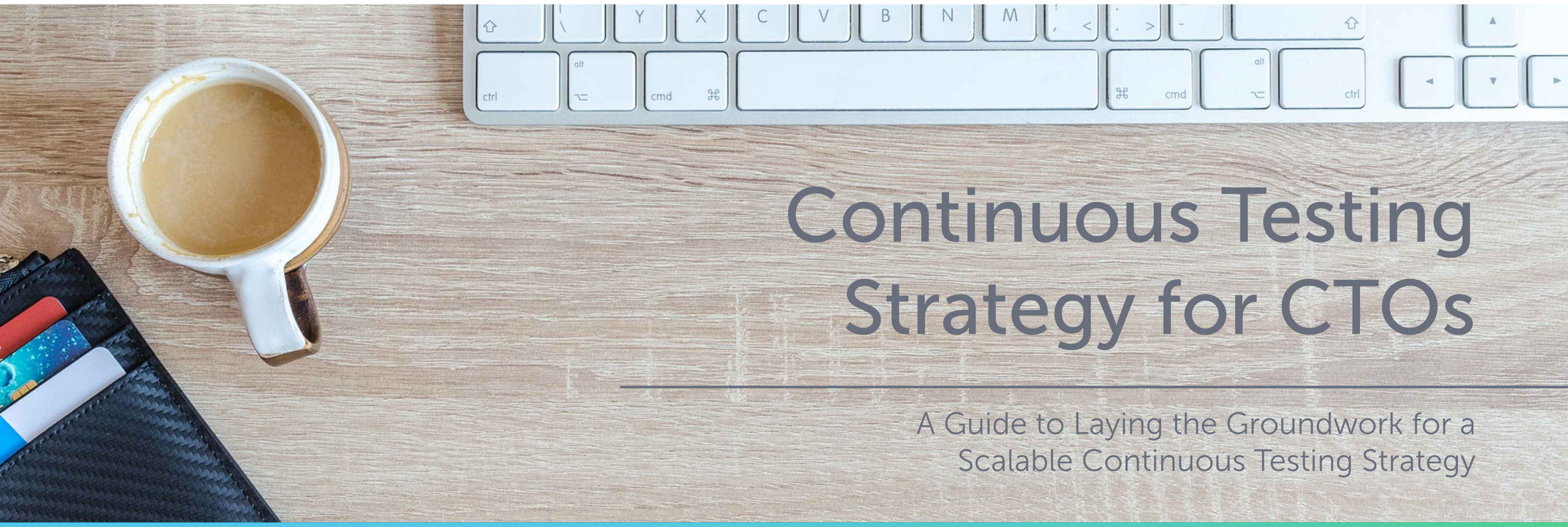
rainforest



# Continuous Testing Strategy for CTOs

A Guide to Laying the Groundwork for a Scalable Continuous Testing Strategy

## In This Guide:

Getting to continuous deployment requires fast, efficient QA testing to keep product quality up without sacrificing speed. But traditional QA methodologies fall short when applied to faster development cycles, and many organizations often risk either overspending on QA to keep quality high, or losing profit or users when they deploy bugs into production.

Continuous testing helps bridge the gap between developing quickly and maintaining high quality products. Many organizations are adopting continuous testing techniques to stay competitive in this fast-moving software market.

A recent survey revealed that 50% of senior-level IT executives stated that their teams are currently utilizing continuous testing to reduce time between development and operation, while a further 29% plan to implement it in the future.[1] To successfully implement continuous testing, CTOs must take a strategic approach to building a testing infrastructure and toolset that empowers their team to move fast.

## Creating a Continuous Testing Strategy

In this guide, we'll explore what continuous testing is, why companies should adopt it, and some of the best practices for ramping up continuous testing strategies. We'll also dig into a continuous testing case study with Zenefits, and learn more about how Zenefits augmented their testing automation processes with the Rainforest platform to successfully do continuous testing.

[1] HP World Quality Report 2015-2016

rainforest

## The Challenges of QA Testing Today

Anyone who ships software knows that shipping broken products is a pain. Customers expect consistently high quality products and have limited patience for bugs and broken software.

The conventional solutions for maintaining product quality have their respective strengths and weaknesses, but none are perfectly suited to continuous testing on their own. As a result, many teams struggle to implement a QA strategy that fits their development process goals.

60% of IT leaders say that they have not yet found a testing strategy well-suited to agile and CD methodologies.

HP World Quality Report, 2014-2015

## The Pros and Cons of Existing Testing Methods

**Automated Testing:** Automation can be extremely fast and effective. But it's expensive to get off the ground because you need human engineers to start writing the test for you. Once tests are written, automation requires dedicated resources to manage any test brittleness or flakiness.

**Manual Testing:** Running tests by hand is a common bottleneck for many companies. It's simple to implement and easy to do. However, it's also slow and can be resource-intensive, especially for larger teams. Because manual testing relies on humans, it's difficult to integrate into a continuous integration pipeline

## Who Owns QA at Your Organization?

55% of QA teams currently spend little to no time on more strategic exploratory testing, and instead spend much of their time running regression and functional verification tests.

The Future of Testing: Where Do Testers Spend Their Time?

**In-house QA:** Whether they're using testing automation tools or not, in-house QA teams are costly to hire, manage, and maintain. The benefit of building an in-house QA team is gaining access to a skilled set of testers with valuable insight into improving products through testing. But in-house QA teams are often overextended, and spend much of their time running low-level, repetitive functional tests. Additionally, even on the longest development cycles, in-house QA is never fast enough. In most cases, a product is "ready" before it goes through in-house QA, making QA a major blocker to getting a product to customers.

**Developer-Owned QA:** We're also seeing that many teams opt not to have a dedicated QA team at all, especially for smaller teams with fast-moving deployment cycles. Developer-owned testing can be a highly efficient way for small teams to allocate resources, as it allows them to avoid hiring a dedicated QA team.

But these developers can end up being highly overextended, and have to make sacrifices on either feature checking or feature development over time. While making developers responsible for quality can keep your team lean, without the proper resources it can waste valuable developer time and focus better spent building new features.

The average savings from outsourcing IT activities is only abut 15% over in-house IT costs for the sam projects.

The Business Value of IT Outsourcing Benchmark Report

**Outsourced QA:** Outsourcing is often seen as a more cost-effective, scalable solution than hiring an in-house QA team. But because outsourced teams are often remote, language barriers and differences in time zone can become a major roadblock to fast development. Additionally, while the cost of outsourcing often seems lower than in-house QA on the surface, the time and resources required to manage an outsourced team can be significant, and over half of companies have reported that their outsourcer has not performed to their expectations.

rainforest

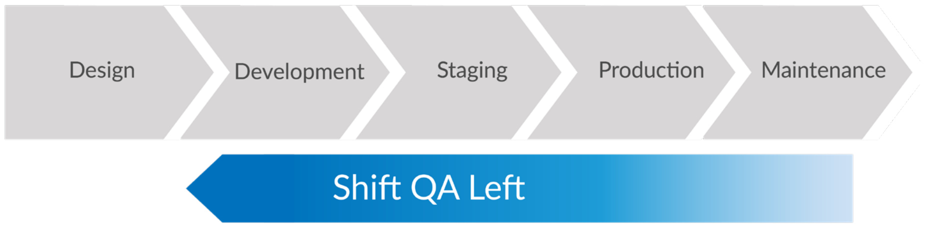
## Integrating Continuous Testing Throughout the Development Process

Continuous testing is the principle of testing your software as often as possible, from early on in development and continuing through release. Generally, continuous testing involves everything from unit tests, automated testing suites, and manual tests run by humans. Implementing a continuous testing workflow that fits into every stage of your development processes helps your team save time and resources.

### When Does Continuous Testing Happen?

Continuous testing starts from the beginning of development and continues through production. The aim of continuous testing is to ship your code faster and give your customer high quality releases.

We advocate "shifting left," or moving testing earlier in the development process. The more you can "shift left" with continuous testing, the earlier you'll find bugs and the faster you'll be able to get your product to market. In this section, we'll outline the kinds of tests your QA strategy should include in each phase of development.



**QA Focus in Development:**

- Usability testing
- Functional testing
- Regression testing
- GUI testing
- Cross-browser testing

### Continuous Testing in Development

Testing should start when the code is still in a developer's hands, before it has been merged into the main code base. Unit testing is the primary focus of continuous testing in development. Some companies are opting for testdriven development (TDD) strategies that put test writing ahead of writing code. Either way, unit tests and some functional testing should start as early as possible to catch and resolve bugs quickly.

rainforest

### QA Focus in Staging:

- Exploratory testing
- End-to-end testing
- Usability testing
- Functional testing
- Regression testing
- GUI testing
- Cross-browser testing
- Database testing

## Continuous Testing in Staging

In staging, testing should take on a more holistic, customer-oriented focus. Your QA team or developers should focus on pushing your application to its limits to find new bugs, then write tests that catch them later. Staging is where exploratory and end-to-end testing are usually introduced, in addition to the functional and regression tests that started in development. Usability, integration and acceptance testing take place in staging as well, to make sure that the product meets standards of design and usability for your customers.

### QA Focus in Production:

- Sanity tests
- Smoke tests
- Regression testing
- Automated testing

## Continuous Testing in Production

By the time a feature reaches production, its test cases should start to become relatively stable. At this point, automation becomes a more viable option, because automated tests are less likely to be brittle when the product is not changing frequently.

Smoke and sanity tests are another important part of testing in production. These "safety net" tests are often run manually, to ensure that the product or feature is functioning correctly. An engineer or QA person will usually check a release just after it's done and run through a couple of core test flows. It is important that these smoke and sanity tests emulate the actual user experience, so they are not often automated. Because Rainforest leverages human testers to run tests, it can be an effective tool for executing smoke and sanity tests at scale.

rainforest

## Zenefits: A Case Study in Continuous Testing

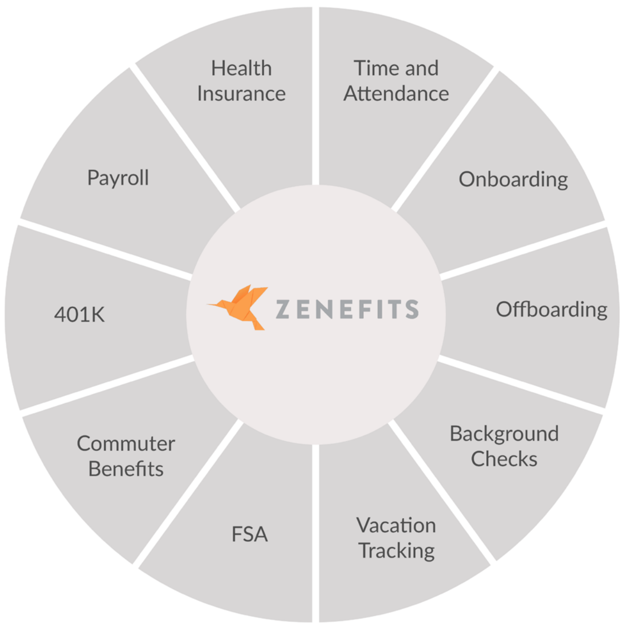
To see how continuous testing actually looks in action, let's take a look at Zenefits.5 Zenefits has been able to scale their customer base to over 10,000 without building an in-house QA team by leveraging a combination of testing automation and Rainforest. Laks Srini, the founder and CTO of Zenefits, cites maintaining quality while scaling rapidly as a major pain point for his team:

> "While we were growing very fast and we were deploying multiple times and adding more and more to our products, at the end of the day we're also enterprise software. The software cannot really break. If people don't get paid on time, or people don't get health insurance, that's really, really bad..."

## The "Hub-and-Spoke" Organizational Model

The bigger the company becomes, the more systems people have. Of the challenges of moving quickly at scale, Laks says, "The problem is that it's pretty hard to build a distributed database when you own all the nodes and the network, but we don't own all the nodes. These are health insurance providers and payroll companies and benefits providers. Sometimes, there's no network."

To combat fragmentation and communication breakdowns, Zenefits has adopted what Laks describes as a "hub-and-spoke" model, with the core Zenefits product as the hub, and the different services rolled into their offerings as the spokes. Each "spoke" has a dedicated development team. "It lets people make decisions in local maxima and move really fast in terms of their product itself," says Laks. "They can make decisions, they can do stuff independently. There's no centralized bottleneck."



### QA Focus in Development:

- Usability testing
- Functional testing
- Regression testing
- GUI testing
- Cross-browser testing

## Roadblocks to Continuous Integration at Scale

While the hub-and-spoke model has helped keep Zenefits' teams productive, one of the major challenges to running a large, segmented team is integration. As code from each spoke is completed, it passes through a CI server before being integrated into the "hub." But as Zenefits' team grew and deployment increased, it started taking longer and longer for every engineer to get feedback whenever they committed code, even with extensive automated test suites in place. Developers were sometimes waiting for hours for code to pass integration tests. "It's almost like you haven't even got continuous testing because the productivity is so low."

## Developing a Scalable Strategy for Continuous Testing

In order to keep Zenefits' continuous deployment strategy working efficiently, Laks and his team have established centralized policies and testing practices. Developers cannot make a pull request without running a test. Everyone uses a centralized pipeline for CircleCI, that goes through the staging environment. Additionally, Zenefits has a centralized programmable infrastructure system with standardized containers with the monitoring, logging, application framework and firewall baked in.

Zenefits does not have a QA team, and each of its over 200 engineers is expected to write their own tests. However, they do have an infrastructure team that maintains their testing framework and CI infrastructure to keep tests running faster. By combining this strategic, developer-owned approach to test writing with a comprehensive suite of smoke and sanity tests in Rainforest, Laks has been able to implement continuous testing effectively and keep deployment fast at Zenefits.

## Leveraging Automation and Human Testing

In the last year, Zenefits grew their engineering team from 35 engineers to over 200, across 3 global locations. With so many people writing code, it was important for Zenefits to implement a plan to ship code frequently to keep risk low. To ensure that there was a human safety net beyond their extensive automated testing suites, Zenefits has leveraged Rainforest for smoke and sanity testing in production.

By automating as much of their QA as possible, Zenefits can prevent QA from slowing down their deployment pace, even when they deploy multiple times per day. Where automation doesn't fit their testing needs well, such as for new features and highly-visual features, they use Rainforest. "For every automated test, we probably have a Rainforest equivalent as well," says Laks. Because their webapp is highly visual, Zenefits relies on tests to make sure that the interface is functionally correctly. These UX tests are challenging to execute effectively with automation. With Rainforest, Zenefits can doublecheck their automated tests results and ensure that the platform is usable for their end users.

## Zenefits Saved $2MM Annually with Rainforest QA

Typically a company with one hundred engineers, like Zenefits, would employ a software QA team of up to twenty people. With the average QA salary ranging between $100K and $125K, Zenefits saves upwards of $2MM a year by using Rainforest to supplement their testing automation activities. Laks clarified the impact of Rainforest on Zenefits' growth: "In terms of ROI, Rainforest has been great. We don't have anyone, other than the engineers themselves, that specifically does human QA and testing. And that's thanks to Rainforest. To be able to have peace of mind when we deploy knowing any human can actually log in, get around the website and everything works, is awesome." [2]

[2] Rainforest QA Interview with Laks Srini, Continous Testing for CTOs Webinar

## Best Practices for Integrating Continuous Testing into Your Development Workflow

If you want to integrate continuous testing into your team's continuous integration flow, identifying and understanding your current testing process is the key. Here are four best practices for getting started on the path to integrating continuous testing into your development flow.

### Identify Gaps in Your Current Testing Process

Identifying weak points in your current testing strategy is key to implementing continuous testing. Audit your existing testing strategy and identify where your team is experiencing stress and bottlenecks in releases. Your team's pain points will often be indicators of QA weak points. Another strategy is to look at any historical bugs and outages, and looking for trends can help identify where to build out your QA infrastructure. For example, if a feature commonly breaks on Internet Explorer, then cross-browser testing might be a weakness in your existing testing process.

### Integrate QA into Development

If you think of QA as its own siloed stage of development, then you can never truly do continuous testing. Continuous testing requires a tight feedback loop between development and QA teams. Some companies have made significant organizational shifts to overcome this siloing mentality. For example, the QA engineers at productivity software company Atlassian often work directly with developers to improve unit tests and implement more effective testing strategies while code is still in development.[3]

### Leverage Continuous Integration Services for Regression Testing

Continuous integration, or CI, gives you the advantage of a consistent build and a consistent run of your tests. Running regression test suites via your CI server, in conjunction with an effective branching strategy, helps reduce the number of regressions in each deployment by catching them early on. This approach allows your team to confirm that the new code will not break any existing features.

[3] Atlassian Blog, "Atlassian QA Makes Development Faster"

## Use Test Automation for Stable Features

Automating tests can be hugely beneficial when implementing continuous testing. As your team grows, you'll likely find that testing automation tools like Selenium improve your team's ability to test quickly, at scale. But be careful of going all-in with automation, especially for features and products that are in flux. Automated tests tend to be brittle or flaky, requiring near-constant management and updating to stay functional. If your team is spending time writing and managing automated tests for features that aren't stable, they're not testing efficiently.

Rainforest can function as a stop gap for features that require more testing than your team can do manually, but that aren't ready for full-blown automation. Because Rainforest tests are written for and executed by human testers, your test cases will be less brittle, less flaky, and easier to maintain than with testing automation for this stage of development.

## About Rainforest QA

Rainforest is changing the way QA is done in an era of continuous delivery. Our on-demand QA solution improves the customer experience by enabling development teams to discover significantly more problems before code hits production.

Hundreds of companies including Adobe, Oracle and Solarwinds use Rainforest to automate their QA testing process and easily integrate it with their development workflow via a simple API. Headquartered in San Francisco, Rainforest is a 2012 Y Combinator graduate funded by Bessemer Venture Partners and SVB Capital, among others.

[6] IBM, The Future of Testing: Where Do Testers Spend Their Time?