Perfecto
A Perforce Company

# Test Automation Strategy For Beginners:

## How to Achieve Continuous Testing in DevOps

by Brad Johnson, Head of Product Marketing at CloudBees

# Test Automation Strategy For Beginners: How to Achieve Continuous Testing in DevOps

DevOps orchestration refers to the set of controls that manages and optimizes the flow of “stuff” through the entire software delivery process.

And within DevOps is testing – including manual and automated test plans, test cases of all types, automation scripts, test environments and their definitions, test data, test results, execution logs, and reporting output.

Testing no longer belongs at the end of the process. With the move from waterfall to agile, testing is now central to the entire software delivery lifecycle — or at least it should be.

As teams strive toward continuous testing in DevOps, the role of automation becomes all the more essential. Automation in DevOps is critical for orchestrating the what, when, and why of end-to-end test processes. Because of this, teams need a sophisticated test automation strategy that will allow them to achieve continuous testing in DevOps.

See how you can get started. Keep reading to learn how you can overcome continuous testing challenges with the help of a strong test automation strategy.

## GETTING STARTED ON CONTINUOUS TESTING IN DEVOPS

One of the challenges teams face when transitioning to continuous testing within DevOps is the oversimplification found in many models. These often underestimate the effort and time required. For new projects and small teams, you can begin with a pure agile approach, utilize the best new tools, and glory in the lack of technical and process debt.

However, most tech professionals work inside of large, established, diverse organizations. In this “real world,” we need to pick battles, work within compliance guidelines, and interact across organizations, geographies, and time zones.

Here are a few tips to consider while working towards continuous testing in DevOps.

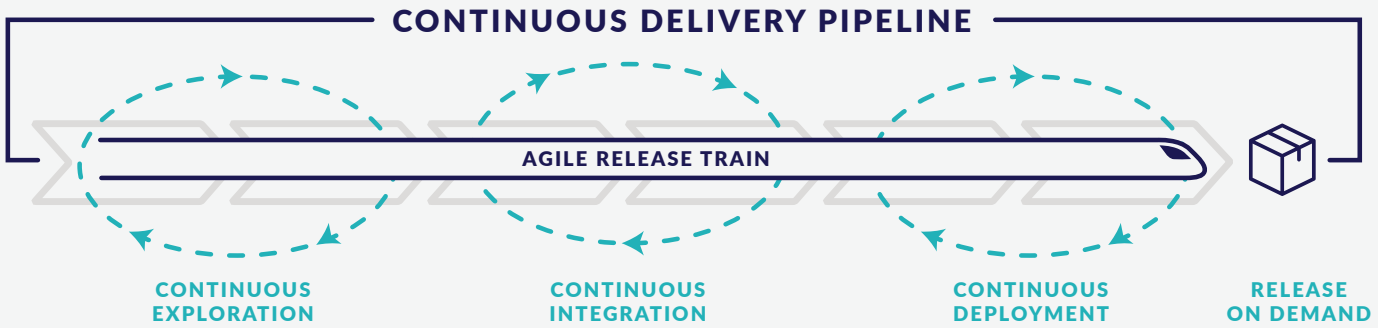
# 1 Create a Pipeline Blueprint

A big mistake many people make is thinking they are not ready to start. They let "if only" statements hold up their progress.

*"If only" our Jira system was more widely adopted.*
*"If only" the dev team would run Jenkins.*
*"If only" my testing service provider would get trained on the latest UI test tool.*

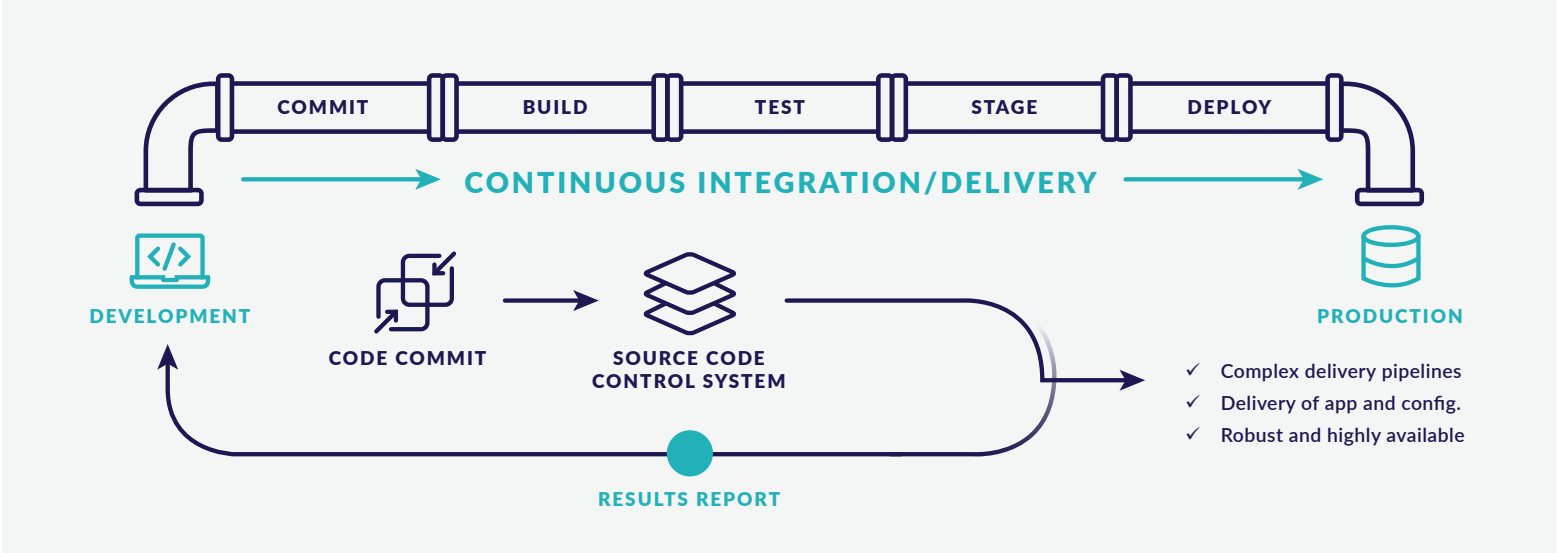
As with most big things, it's best to begin with what you have control over and start planning from there. In this instance, that's the DevOps pipeline.

Pipelines are the fundamental pathways that delivery orchestration follows. They consist of all tasks and deliverables that software delivery teams execute on to bring ideas into production via software. The "continuous" aspect of the DevOps pipeline considers the iterative and repeatable nature of software delivery. One methodology, SAFe, considers cyclical segments often owned by different teams interacting constantly along a pipeline (Figure 1.)



# Create a Pipeline Blueprint (continued)

Other pipeline views break out major stages of software delivery and emphasize technologies like source control and orchestration automation (such as Jenkins) as the engine that drives the pipeline. (Figure 2.)



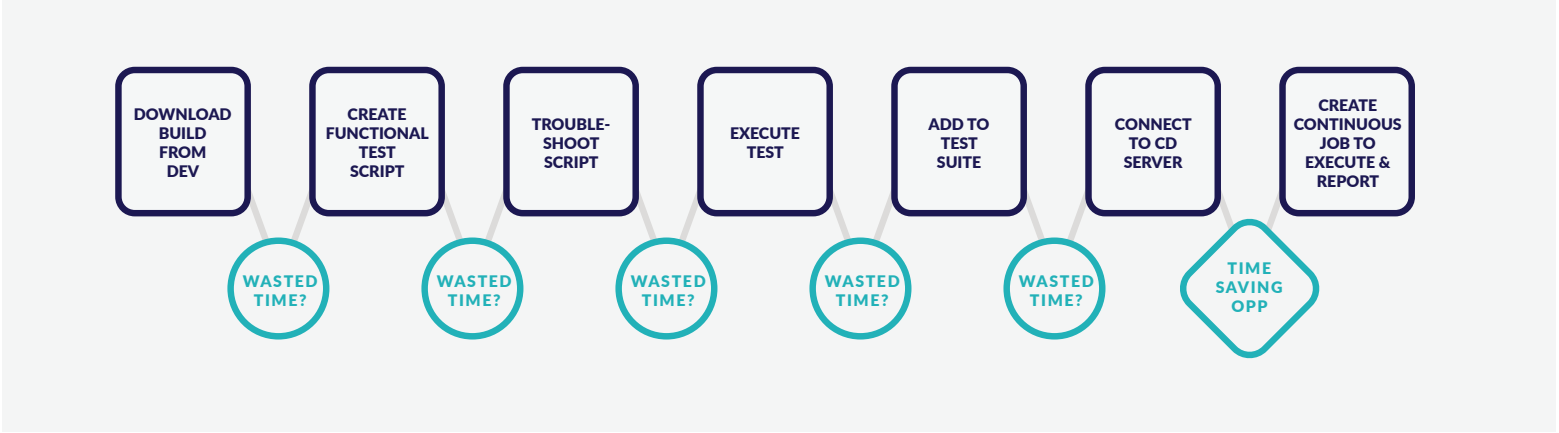

Regardless of your choice of implementation, a critical first step is to identify and document your pipeline.

If you are delivering software, then you have a pipeline. It may be partially or poorly automated. It may be broken by organizational delays. It may be blocked completely for months at a time — perhaps by testing itself. However, it is still a basic pipeline and a good way to begin is to map it out.

Start with any flowcharting technique and lay out the steps of a basic flow that you already do. Or, lay out a way that could be beneficial to implement and automate. This simple model is a quick exercise any team can do.

# Create a Pipeline Blueprint (continued)

Figure 3 is a fast and simple example of a functional test stage that most companies do today. In this diagram, the bubbles connected to steps are “wasted time.” By identifying where you know time can be cut or improvements can be made, you can plan out how to build a continuous delivery pipeline segment by segment.




Once you have even a small set of these pipeline blueprints vetted, with the areas of most time savings identified, it’s time to begin pulling in test automation strategy and other techniques to drive your DevOps orchestration pipeline, piece by piece.

# 2 Leverage the Cloud

Much can be — and has been — said about cloud computing. With instant and disposable availability of compute resources, the cloud should be the first choice for development and test teams. With a track record of more than a decade now, teams who are not leveraging the cloud are not only behind, they are at a disadvantage. Check out a few basic test use cases for the cloud:

### COMPATIBILITY TESTING

Quickly spin up multiple operating system types or versions, simultaneously, with a command. Containers make this even easier as you can deploy to different clouds. Functional tests, for instance, could be run against many operating systems. Similarly, you can perform multiple simultaneous browser tests, database types, etc.

### LOAD TESTING

Using load generators to drive simulated traffic from multiple regions and up to large scale was one of the first use cases for cloud computing. Load tests can be triggered by a build job and results can be passed in an automated fashion as part of a performance regression suite.

### USER ACCEPTANCE TESTING

AUT environments can be spun up as part of an automated process. Real users in any location and environments close to users (or far away) can be utilized to test the network impact of systems close to or far away from users.

# 3 Embrace Automation

As with the cloud, test automation, build automation, and automated orchestration and deployment have more than enough maturity to make the most skeptical testers confident.

A never-ending saga in the testing industry revolves around automation in DevOps. It doesn't matter how you feel about what is possible to automate, when you should start, how to manage test fragility, or who should create or execute tests. What does matter it that you plan to utilize a test automation strategy with your CI/CD orchestration.

If you aren't automating regression tests, you are wasting time.

Functional test automation has always been a great place to start with automation. For web, recording technology is mature and stable, so even junior-level test engineers can reliably capture basic tests to replay. Once sets of automated functional tests are proven, repeatable, and combined into regression suites, it is a simple matter of ensuring the test environment is available — perhaps in a cloud — and running the suites as part of the build process.

## ADD MOBILE TESTING

Nearly every application has a mobile component and nearly every company has a mobile app. The need to test on many different devices and the pace of change of those devices is daunting.

However, mobile testing has matured and recording or scripting tests to run on emulators or real devices can be highly automated. In turn, this automation can be triggered as a CI/ CD orchestration job, to be run "hands free."

## LOAD TEST, CONTINUOUSLY

Finally, load testing need not be the job of specialized teams. Using cloud or traditional servers, once load tests are created — at a single component test or even end-to-end — the CI/CD pipeline should include fully automated load test runs at various stages.

The earlier teams start testing for scale and performance, the fewer problems they will encounter at the "Ops" end of the DevOps cycle.

# Start Continuous Testing In Devops

Successfully transitioning to continuous testing in DevOps is dependent on automation. And a winning test automation strategy is defined as this:

- If automation is baked into the entire pipeline from start to finish.
- If test code (functional, nonfunctional) is treated as product code.

When the overall test orchestration runs at scale upon each code commit, the developers have much more time to focus on new features and to maintain high code quality that, overall, leads to faster release cycles. DevOps leadership ought to enforce test orchestration processes as a standard working flow and enable their direct reports to be successful in authoring, maintaining, and executing test automation strategy.

## START SMALL

As with every complex challenge, start small and go from there. The key to successful continuous testing in DevOps lies in building trust between the dev and test practitioners. To build trust, there needs to be a small and proven smoke or sanity test suite that runs automatically within CI (Jenkins) upon each code commit.

Once trust is built for a small test suite, the next step is to scale the number of platforms that undergo these tests to “certify” said tests and make them an integral part of the DevOps pipeline.

Next is scaling the test suite to cover more essential test scenarios and platforms. The import- ant thing in scaling up a “trustable” running suite within the CI is to continuously maintain the coding practices and test certification processes to maintain trust between team members.

## START NOW

The entire industry is implementing agile development practices and aiming to scale DevOps and shift to CI/CD. Continuous testing is, to date, the biggest bottleneck from maturing DevOps due to trust, test flakiness, lack of best practices, and unstable orchestration methodologies.

# Start Continuous Testing In Devops (continued)

Begin learning how to shift toward continuous testing, scale such testing, and make it a reality as soon as possible.

There is no other way to shift an organization to DevOps without assuring high quality iteration releases. If releasing a new piece of code takes too long, and at the end it's not covering sufficient platform and code, the entire pipeline is stuck.

And that's not really DevOps.

**GET THE PLAN:** Perfecto has a continuous testing evaluation and checklist tool called the Perfecto Continuous Testing Blueprint. Sign up at https://blueprint.perfectomobile.com/bp/blueprint.php and run through the checklist to see where your team is in the process. The Perfecto Continuous Testing blueprint is a proven tool used by many of our top customers to help navigate the waters of continuous testing. You'll find exactly where you stand in the process and understand your current maturity level, what you can gain by advancing, and exactly what steps are involved in attaining your goals.

**GET STARTED:** Start your free trial from Perfecto today.

## ABOUT PERFECTO

Perfecto is a Perforce company. We enable exceptional digital experiences and help you strengthen every interaction with a quality-first approach for web and native apps through a cloud-based test environment called the Continuous Quality Lab. The CQ Lab is comprised of real devices and real end-user conditions, giving you the truest test environment available.

More than 1,500 customers, including 50 percent of the Fortune 500 companies across banking, insurance, retail, telecommunications, and media rely on Perfecto to deliver optimal mobile app functionality and end-user experiences, ensuring their brand's reputation, establishing loyal customers, and continually attracting new users. For more information about Perfecto, visit www.perfecto.io, join our community and follow us on Twitter at @PerfectoMobile.

## ABOUT BRAD JOHNSON

Brad Johnson has spearheaded product go-to-market as well as driven technical and channel alliances for market-leading enterprise software firms and cloud-pioneering startups in Silicon Valley for over 18 years.

He speaks and writes on executive level technical topics that span software development, testing, application performance management, and now DevOps. He is currently driving high performance transformations as WW VP Growth Marketing at Gremlin.