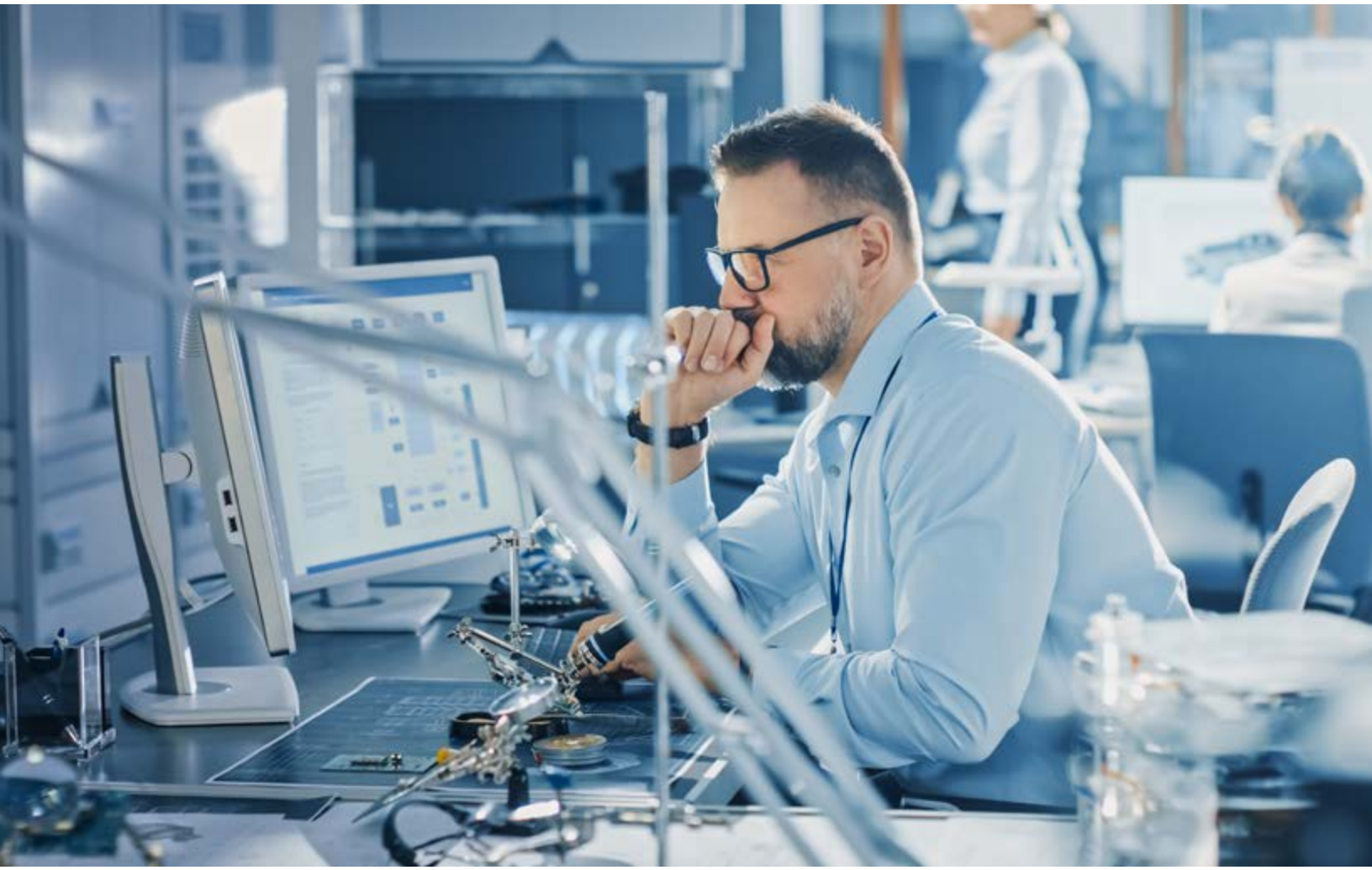
PROD PERFECT

WHITE PAPER



How to Ship Products Fast and Fixed

Dan Widing, Founder & CEO

TABLE OF CONTENTS

INTRODUCTION

"Without continual growth and success, such words as improvement, achievement, and success have no meaning."
—Benjamin Franklin

Modern software development has changed dramatically from even 5 years ago: things move fast. New features are developed continuously, and the ease of updating web applications means that software is constantly in motion. This creates fierce competition to constantly improve an application for your customer base.

THE PAST: FAST AND BROKEN

Facebook famously launched a "move fast and break things" approach to its development. Many followed. But, even Facebook eventually learned that "move fast and break things" doesn't work. It turns out users don't like broken applications, but they want something that works. This adds even more pressure to dev teams to maximize both speed and quality/functionality when shipping code.

THE FUTURE: FAST AND FIXED

Rather than a "fast and broken" mindset where a team fixes software in production, after users see bugs, a winning software team will have a new standard: fast and fixed. Get new features out quickly, and make sure they work. Continuous, quality delivery is the name of the game.

However, this standard is extremely difficult to reach. Most teams feel constrained to pick just one focus: speed, quality, or price. But, if you want to win, you need to do it all. And this requires a highly mature organization with mature processes.

If you want to ship fast and fixed products—if you want high quality continuous delivery—you need three process elements to complement your team's talent and rigor:

- **An efficient continuous integration (CI) pipeline**
- **A diligent and thoughtful code review process (with every build)**
- **A rigorous testing methodology (with every build)**

"Rather than a "fast and broken" mindset where a team fixes software in production, after users see bugs, a winning software team will have a new standard: fast and fixed."

THE COMMON TESTING PREDICAMENT

Adopting a CI pipeline is becoming standard, and standardized. Code review is a subject for another time and requires discipline and work, but can be achieved through sufficient effort. But what constitutes sufficient testing much less rigorous testing is its own set of challenges that are largely unsolved in the industry: most teams are picking between having extensive and slow testing suites, or minimal and fast ones. The former hurts deployment speed. The latter hurts quality. Most teams don't believe they can have both.

Part of the problem is that teams frequently misuse the testing tools they have available. Unit tests are best applied to enable refactoring, validate low-level intent, and manage edge cases. Having a large number of unit tests doesn't necessarily imply that you will catch bugs because inherently they're not testing what users are doing in the way users are doing them. They have to do this to be able to be run quickly. To validate that the high-level features are working the way intend them to, you need to exercise all the levels of the application in the way users would exercise them. This means exercising a full server (or model thereof) and the part of the application that lives only in the browser. Exercising the browser is particularly important as more teams move more behavior into single page applications. These browser level tests are therefore slower, harder to write, and more difficult to write because of this complexity.

THE PATH FORWARD

The path out of this choice is to let unit tests be unit tests and then shift the focus of browser testing from needing to be "extensive" to being "accurate."

Fast and accurate browser testing can be described as satisfying four requirements:

- **Using a test runner that runs quickly and stably (we suggest Cypress, TestCafe, or Capybara).**
- **Covering the important user stories in your application.**
- **Minimizing unnecessary and overlapping tests (and eliminating obsolete ones).**
- **Running as many tests as possible in parallel.**

"Let unit tests be unit tests and then shift the focus of browser testing from needing to be "extensive" to being "accurate."

It goes without saying that browser testing needs to be automated, rather than manual, to ship fast and fixed code. Building it from scratch is an option, but one that often fails: talent is rare, resourcing is expensive, and test suite stability tends to degrade. There are a number of external services that will help a team run browser testing that meets the above criteria, including the following:

Crowdtesting

Crowdtesting is a process by which the software team provides test cases to the vendor, and the vendor provides a large bank of people to manually perform the scenarios. It has a few advantages: it's easier to set up than your own automation suite, it requires less ongoing maintenance than a home-built suite, and manual testers can sometimes catch bugs that automated tests would miss. However, this approach has several drawbacks.

Because customers pay for each test run, more software shipped correlates directly to more money spent. While manual testers can sometimes catch bugs that automated tests would miss, they will also frequently report false positives or miss other critical bugs, due to the inexactness of a manual review by an untrained/unfamiliar resource. In addition, while the only real

maintenance is updating test instructions, it still means that a resource has to be assigned to the task, continually updating and changing the test cases to prevent the test from becoming stale and outdated.

Machine Learning (ML)-Enabled Record-and-Play

With Machine Learning (ML)-Enabled Record-and-Play, a third-party application adds an additional layer to your own, allowing you to build tests recording you using your software. These tests are intended to be functional through many small changes of your software, by building "models" of the test event, rather than using conventional testing hooks. This reduces test

maintenance costs. Because the tests are truly automated (rather than crowdsourced), you wouldn't have to pay for the cost of running each test.

However, since it is your team developing the tests with the external application, the gap between your team's understanding of the application and actual user behavior remains. Additionally, the tests have to be redone every time there's an appreciable change to the product, requiring substantial attention and input from your team. Lastly, since tests all run through the interface, if you decide to leave the service, you take no assets with you—you're back at square one.

Autodetection/Autogeneration

ProdPerfect offers the final type, Autodetection/Autogeneration. Autodetection tooling analyzes user traffic to determine test cases that represent common flows of user behavior, and then Autogeneration automatically produces repeatable test scripts based on those test cases. The process requires no input from you and minimal (for ProdPerfect) human input to finish test validations. Autodetection and Autogeneration work together continually to update and maintain your test suite through each build of your product, allowing for accurate and realistic testing with minimal time and effort. The tests are parallelized, so they run quickly, and the results are automatically and instantly sent to your developers through CI. Also, you get a copy of the test code with each build, allowing you to run it on demand and continue using it if you leave the service.

When using Autodetection/Autogeneration services, your team will still need to test brand new features that do not affect any previous functionality, as they will not have yet been detected from user traffic.

Ensuring all releases in continuous development are fast and fixed isn't easy, but it's absolutely necessary. By removing some of the burden of end-to-end browser testing for each release, your team can focus on doing what they do best: developing the best product they can quickly and efficiently.

"By removing some of the burden of end-to-end browser testing for each release, your team can focus on doing what they do best: developing the best product they can quickly and efficiently"

ABOUT PRODPERFECT

Unleashing the power of machine learning to solve the hardest, most important, and previously unsolved problems in end-to-end (E2E) QA testing, ProdPerfect is the only autonomous E2E regression testing solution on the market that continuously identifies, creates, maintains, and evolves E2E test suites via data-driven, machine-led analysis of anonymous live user traffic. It is a fully-managed testing solution that addresses insufficient test coverage which causes critical and costly bugs in production; removes the burden that consumes massive engineering resources; and eliminates long test suite runtimes that slow deployments and decrease developer velocity.

SCHEDULE A PRODUCT INTRODUCTION TODAY

PROD PERFECT