EBOOK

ESSENTIAL GUIDE TO SHIFT YOUR TESTING LEFT

APPLAUSE

ERAN KINSBRUNER, AUTHOR

Eran Kinsbruner is an author and the chief evangelist at Perfecto, a Perforce company. He's authored two books, *The Digital Quality Handbook* and *Continuous Testing for DevOps Professionals*. Eran is also a monthly columnist at the Enterprisers Project.

Eran is a software engineering professional with nearly 20 years of experience at companies such as Sun Microsystems (Oracle), General Electric, Texas Instruments, NeuStar and Matrix. He holds various industry certifications from ISTQB, CMMI and others.

Eran is a recognized software, web and mobile testing influencer and thought leader, as well as an experienced speaker in the major software engineering conferences. He is also a researcher, blogger and a patent-holding inventor for a test exclusion automated mechanism for mobile J2ME testing.

You know all about the most common terms in software testing — functional testing, non-functional testing, integration testing and unit testing, to name a few. They're as ubiquitous as football and Mickey Mouse.

As we continue to shift toward Agile and DevOps, we need to think in terms that better fit the nature of modern software delivery. To be clear, we're not throwing out those older terms. Rather, we're discussing the concept of shifting your testing to the left — often referred to as shift-left testing. What does this mean? With a shift-left approach, you no longer perform testing at the end of your software development lifecycle, but instead move it earlier — or rather, shift it left — in your SDLC.

Why should your organization consider embracing shift-left testing? Because if it doesn't, it will struggle to:

- Rapidly release software
- Quickly identify hidden and escaped defects
- Provide timely feedback to the development team

Read on for more details on what a shift-left approach can mean for your organization.

What is shift-left testing?

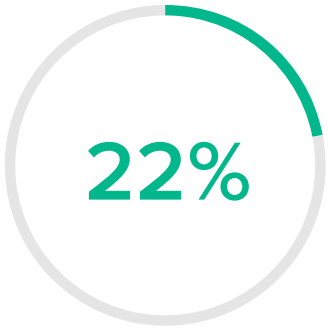
When you shift your testing to the left, developers and testers collaborate earlier in the development cycle with the goal of achieving the highest degree of relevant coverage. There are various models of shift-left testing, including:

- Incremental
- Agile/DevOps
- Model-based

Under shift-left testing, testers are well-versed on the requirements, software design, architecture, coding and functionality. This enables them to ask questions, seek clarification and provide feedback wherever possible to support the team.

Agile/DevOps is often the most useful method of shift-left, prioritizing the creation and execution of testing as close as possible to the moment code gets changed. However, only 22% of organizations are doing automated testing early in the SDLC.[1]

If your organization hasn't adopted test automation yet, you risk falling behind competitors who can test code — and thus release software — quickly. Automation is key to implementing a shift-left approach, and with automation scripts, test teams can perform testing several times a day. When your automation scripts find bugs, they are fed right back to the product development funnel, which helps

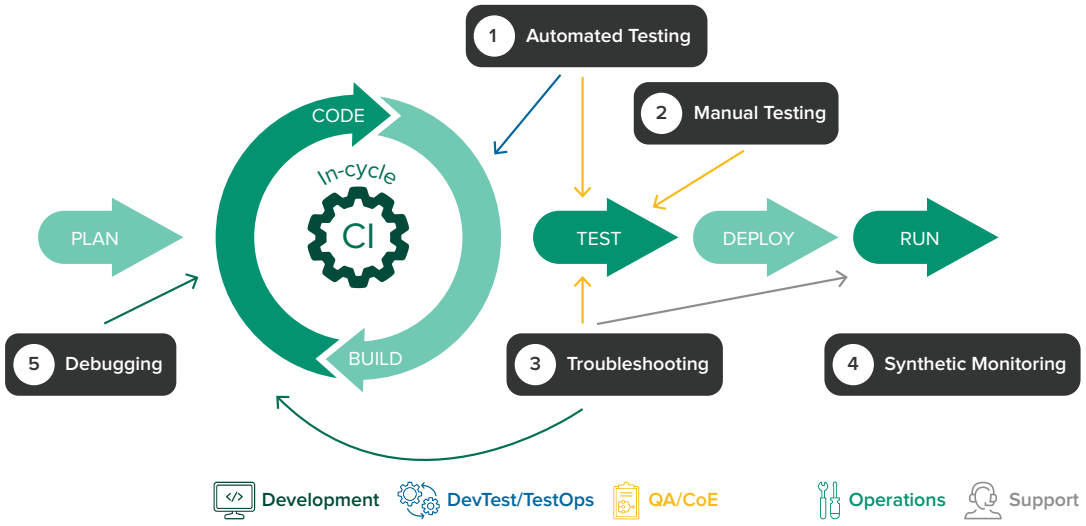


22% of organizations are doing shift-left test automation[1]

improve the code quality (as well as code coverage through the development of test cases and test suites). Using automation early in the SDLC can speed up your overall software delivery cycle, as well as mitigate business risks by quickly identifying defects earlier in the cycle.

Without shift-left testing, organizations are dependent on QA teams to fit their testing scope into a given window, only after they integrate code into the main branch. **This risks late delivery of important feedback to the developers, which typically results in more defects escaping into production.**

With a shift-left approach, testing moves earlier in the SDLC. Automated testing early in the SDLC can speed up your overall software delivery cycle.



Benefits of shift-left testing

The number one benefit of shifting your testing to the left is the ability to detect bugs much earlier in the SDLC.

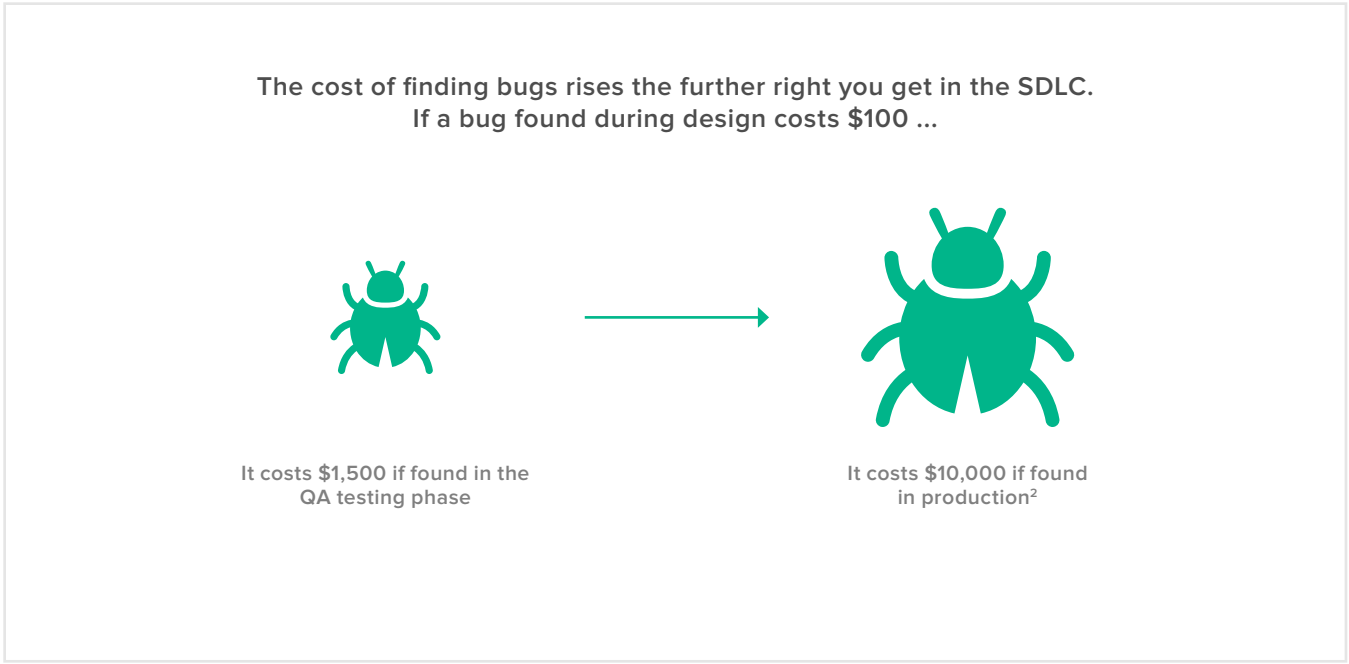
It's a well-known fact that you can reduce costs significantly by finding bugs earlier in the SDLC. For example, if a bug found during design costs only $100 to fix, it will cost $1,500 to fix if it's found in the QA testing phase, and $10,000 if it's found in production.[2] When you

shift your testing to the left, you are far more likely to discover bugs earlier in the SDLC and avoid expensive bug fixes.

When organizations embrace a shift-left approach to testing, they can quickly assess the impact of any code change. More importantly, they can obtain quality-driven insights that drive smarter decisions on future steps planned for the tested software iteration. This is helpful as code size becomes larger. When the code size increases, it can take more time (and money) to discover and fix bugs if you only test for them at the end of the SDLC. A shift-left testing strategy can reduce the overall costs of development.

In addition, **shift-left testing can become an efficient gatekeeper for DevOps/Agile teams** prior to integrating new code into main branches. This enables teams to avoid breaking a working software branch. It's also more cost-effective to involve test engineering earlier in the development process, making it a business driver too.

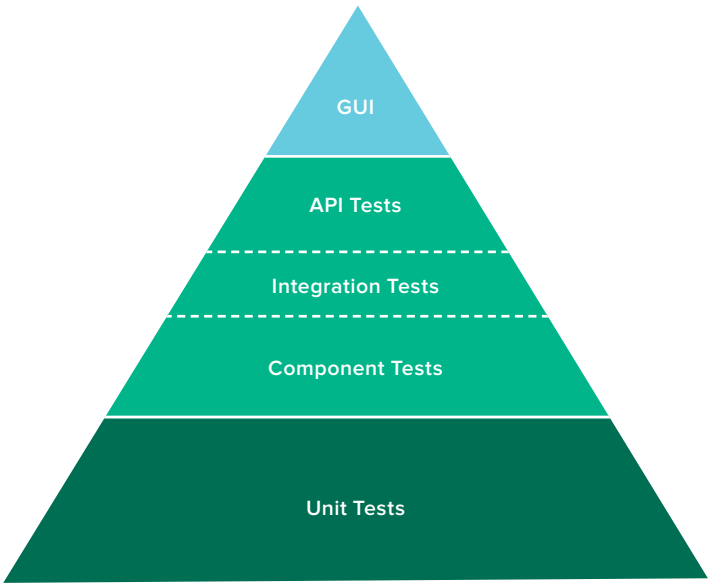
In other words, **shift-left testing enables teams to fail fast and fix fast**, as opposed to testing only at the end of code development.



Here are a few of the key benefits organizations can expect by emphasizing shift-left testing:

- Organizations see a cost reduction within the DevOps cycle through early detection of defects. The sooner you detect bugs, the cheaper it is to solve them.
- Organizations can dramatically reduce code instabilities and increase their developers' efficiency by combining coding and testing into a single activity. This helps identify bugs earlier in the process and release software more frequently.
- Enable developers to quickly test their code changes via continuous integration (CI) and test automation.
- Teams can better and more modernly structure the entire testing scope for their projects, using the well-known test automation pyramid (prioritize unit, API, acceptance, integration, UI, code analysis).
- Improve business results and generate higher customer satisfaction through a higher-quality product.
- Mature your software development cycle toward continuous testing, continuous integration and continuous delivery (CI/CD).

Only 40% of organizations are confident in their ability to achieve both speed and quality[3]



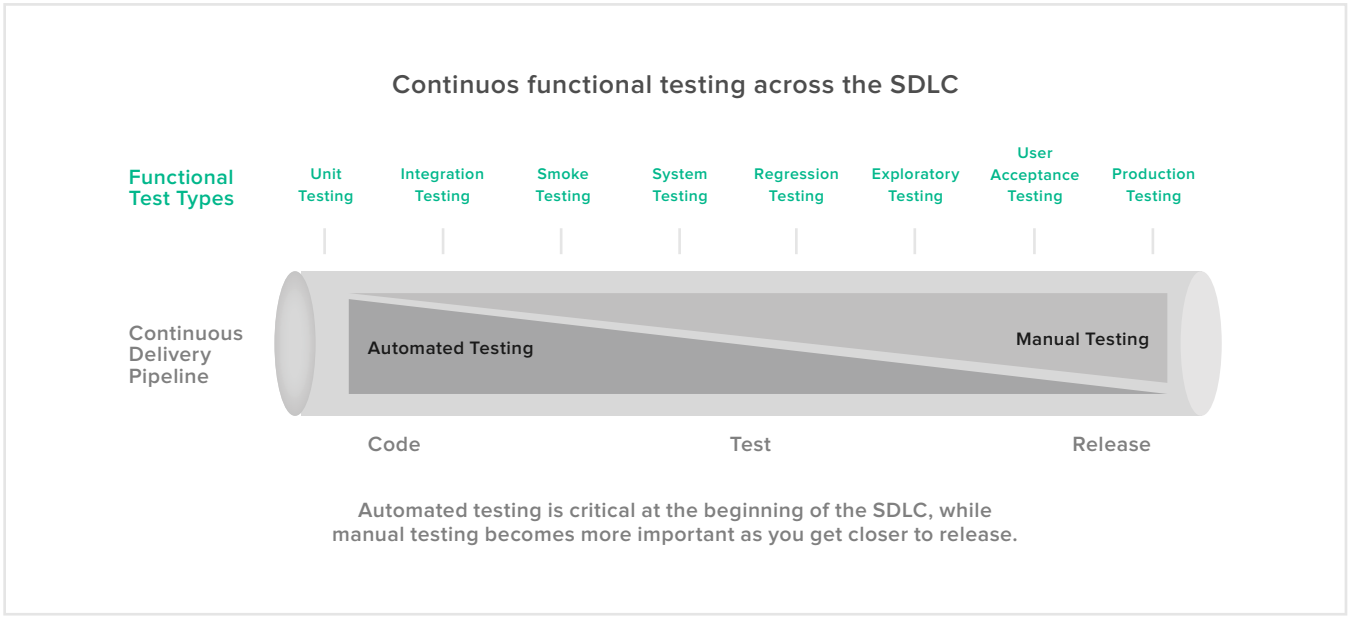
Best practices for shift-left testing

How can you actually enjoy the benefits of shift-left testing? Here are my recommended best practices to succeed in the journey toward continuous testing.

Practice #1: Process and Culture

In software development, all practitioners must align when it's time to make a change in the process. Shift-left testing involves the development, testing and product management teams. In other words, shift-left testing is the entire organization's effort and responsibility.

The parties must communicate clearly so they can understand each other's needs and include the right tests in the build cycle, efficiently debug failures and move forward iteratively. When process and organizational culture are set and embraced, that's when productivity and quality increases.



Practice #2: Technology Enablers

The term 'shift-left' includes all kinds of testing, and refers to the goal of completing more test automation cycles earlier in the build cycle. In order to properly shift your testing to the left, teams must perform the following at scale, all within a CI/CD pipeline:

- Perform various types of testing
- Conduct static code analysis using specific tools
- Maintain platform coverage at scale, ensuring platforms under test are up-to-date and in a ready-state mode for testing

More specifically, for each software build that undergoes testing, the scope of the test suite should include:

- A mix of key functional testing scenarios
- Unit testing that is specific to the changed areas between the previous and current build
- Security, performance and relevant exploratory testing that are triggered through the CI jobs and are executed in parallel on various supported platforms

Since software releases are dynamic, practitioners must apply suite auditing and maintenance between builds to ensure that the scope remains relevant and keeps up with the changed code or implemented story points.

Lastly, the platforms targeted for testing (mobile devices, web browsers, desktops, etc.) must also be at ready-state mode between and throughout the cycles. It's also best to leverage test reports to understand build quality and test automation suite effectiveness. These test reports can unveil regressions in the test code and alert you to issues in advance, enabling you to fix them before the next execution.

Practice #3: Treat Test Automation Code as Production Code

In general, test automation is a clear enabler for shift-left and Agile testing. However, to benefit from shift-left, test automation must be of high value. To deliver that value, developers and test engineers should maintain test code through source control and code reviews. In addition, keep your testing within the CI pipeline and always green-light code unless there is a real defect.

With any test automation, there is an inevitable degree of "noise" and flakiness. Teams must reduce these cases continuously, or the entire practice will fail. By following coding standards, gaining quality visibility, embracing AI/ML algorithms and other methods, you can keep that "noise" to a minimum.

How Applause Helps You Shift Your Testing Left

Many companies lack the internal QA resources to properly shift their testing left in their SDLC. For example, a company with only 1-2 dedicated in-house QA testers might not have the bandwidth to test during the early stages of development. With solely in-house resources, development teams often only have the manpower to test at the end of the SDLC, which prevents them from experiencing the quality and cost benefits of finding bugs at the beginning of the SDLC.

A crowdtesting approach, such as what Applause offers, can help teams test early and often, ensuring that bugs are found when they're less expensive to fix. Applause assembles and manages custom teams of qualified testing professionals that remain focused on your product throughout your SDLC.

These testers become integrated into your team, and can scale up or down as needed to match your company's fluctuating testing volume or business priorities. This eliminates the concern about the lack of in-house resources to test earlier in the SDLC.

Applause's fully managed service integrates seamlessly into your company's existing Agile or CI/CD environment, augmenting your internal team's ability to keep pace with development. It's not about replacing your QA team, but dedicating the resources you need to ensure you find bugs early in the SDLC and avoid the expenses of fixing later in development or even post-production.

It's time to think about shifting your testing to the left in your SDLC. The good news is, you don't need to do it alone.

Applause helped a prominent real estate listings company shift its testing to the left. The results:

1.5x lift in ability to catch defects earlier

Achieved 70% bug prevention rate[4]

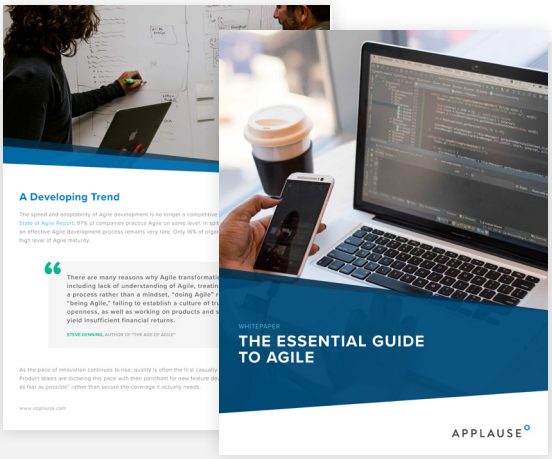
Citations:

1 Capgemini and Sogeti

2 IBM Systems Science Institute

3 CA Technologies

4 Hobson & Company



The Essential Guide to Agile

Learn how you can be Agile, not just do Agile

DOWNLOAD

About Applause

Applause is the worldwide leader in crowd-sourced digital quality testing. Software is at the heart of how all brands engage users, and digital experiences must work flawlessly everywhere. With 300,000+ testers available on-demand around the globe, Applause provides brands with a full suite of testing and feedback capabilities. This approach drastically improves testing coverage, eliminates the limitations of offshoring and traditional QA labs, and speeds time-to-market for websites, mobile apps, IoT, and in-store experiences.

Thousands of leading companies — including Ford, Fox, Google, and Dow Jones — rely on Applause as a best practice to deliver high-quality digital experiences that customers love.

Learn more at **www.applause.com**

NORTH AMERICA
100 Pennsylvania Avenue
Framingham, MA 01701
1.844.300.2777

EUROPE
Obentrautstr. 72
10963 Berlin, Germany
+49.30.57700400

ISRAEL
10 HaMenofim Street
Herzliya, Israel 4672561
+972.74.757.1300

APPLAUSE