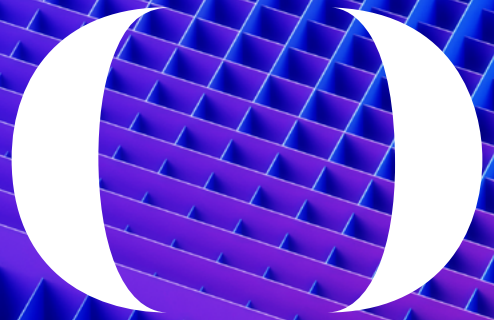
opentext™



ADM Market Insight:
Improve Application Quality in Production: Shift-Right Testing with OpenText

Key Takeaways

As companies continue with their digital transformation initiatives, performance engineers are expected to release software faster and meet enhanced performance requirements while optimizing the customer experience. As a result, there's been a recent push for them start testing earlier in the software development life cycle (SDLC). Often referred to as shift-left testing, pre-production testing determines whether the code is returning expected results, allowing the program to work properly.

Shift-right testing is a critical, complementary element to shift-left testing. Shift-right testing continuously tests software while it is in the post-production environment. Also known as "testing in production," this approach tests applications in production. It detects issues that might not have been apparent during development, such as application latency or availability. Identifying these issues early enables your organization to deliver a seamless user experience that customers expect.

Let's take a deeper look at shift-right testing and its benefits.

Combining Shift-Left and Shift-Right Testing for Continuous Quality

Implementing both shift-left and shift-right testing into the SDLC process allows software development teams to build quality applications from the beginning and get continuous user feedback. The combination of the two also delivers the following benefits:

1. Reduce costs in development and testing.
2. Detect and address bugs early, ensuring better code and product quality.
3. Deploy time, manpower, and resources more efficiently.

Shift-left testing starts earlier in the development process, focusing on problem prevention rather than detection. Think about unit testing, manual testing, and integration testing. In contrast, shift-right testing development teams test scenarios and applications towards the end of the SDLC. Tests in production examine functionality, performance, failure tolerance, and user experience (UX).

Shift-right testing identifies unexpected scenarios not examined in the development environment. By monitoring, observing, and analyzing log data and "testing in production" shift-right testing helps ensure the correct behavior, performance, and availability of an application.

The Case for Shift-Right Testing

Shift-right testing happens late in the application lifecycle, pre- or even post-deployment (that is, in production). By testing in production, you're ensuring that an application will work under all circumstances and environments. Planning to roll out a new business application for your global workforce? Want to ensure customers have access to your ecommerce site during a sale? Shift-right testing covers it all.

No matter how well you test up front, elements of the application can break in production. Knowing when and why is crucial for guaranteeing software quality. Shift-right testing enables a continuous feedback loop from users and analyzes use cases that teams can't anticipate such as crashes, failures, and slow performance.

Examples of shift-right testing include A/B testing and canary testing. In A/B testing, you redirect a portion of your site's users to a new site, testing whether users stay on your page longer and if they look and click where you expect. Canary testing redirects a small portion of your users to an update of the existing website to confirm everything works as it should, especially integrations. If all works well, you can then roll out the update to all users.

Testing whether all APIs and microservices are available and work appropriately is paramount, especially with many APIs and microservices available. Shift-right testing helps accomplish this best practice.

Delivering a quality user experience is critical whether the user is an employee or customer. By rolling out apps that work upon release, you build customer confidence, which can positively impact revenue.

Anticipate Potential Issues

Experienced developers know that even when everything works well in development after testing, something could still be unavailable in production.

Picture this: a client wants a file upload option on their website so that users can upload invoices and receipts for a manager's approval. Developers write the solution and test it using five different browsers and versions. All work well, so they deploy to production.

A month later, when the manager wants to approve all files from the month before, in comes the bug report: the manager only sees "[object Object]".

It turns out some users are still using an old version of Internet Explorer, which does not support the JavaScript API used for uploading files. Instead of saving the bytes to a file, the application has saved the bytes of the text "[object Object]."

If they had tested in production and identified defects earlier, the team could have collaborated with performance testers, identified defects sooner, and facilitated a better user experience.

Instead, the client has lost a month's worth of files.

Adopt End-to-End Performance Engineering Solutions

Shift-right testing sounds great, but how do you start? First, you need a tool specifically for testing in production. The **OpenText LoadRunner family** is a great way to get up and running fast. With multiple LoadRunner solutions to choose from, you'll be able to implement extensive, flexible test scenarios and assess their impact on every application component.

LoadRunner's family of tools delivers end-to-end performance engineering for shift-left and shift-right testing. Its intuitive and easy-to-use interface makes it simple and versatile for performance testers of all levels. The LoadRunner family delivers both shift-left and shift-right capabilities, with tools engineering quality and optimizing performance throughout the DevOps pipeline.

Real-world performance engineering

LoadRunner solutions support all types of web, mobile, and packaged applications without heavy customization. For performance testing, you can apply workloads to any application with flexible hardware usage. In production, you have a single view of end-user response time, infrastructure-level, and network breakdown.

It's easy for DevOps to integrate continuous performance testing in the CI/CD process. LoadRunner also makes it easy for teams to collaborate, so it's an ideal tool for your DevOps and TestOps environments.

Sharing and collaboration

When testing, you need to have quick access to enterprise engineering capabilities and facilitate asset sharing and collaboration. LoadRunner Solutions allow global teams to share a common infrastructure and can execute multiple performance tests concurrently and continuously with all relevant assets being shared to increase collaboration. With cross-project reporting and individual project drill-downs, it's easy to see your entire application landscape. Ultimately, you'll be able to analyze end-to-end performance, including topology, infrastructure-level, and advanced insights.

Performance engineers gain 24×7 access to all testing operations, including uploading test scripts, scheduling load tests, creating load test scenarios, running multiple load tests, monitoring test executions and analyzing results. Everyone on the team can view load testing data, progress, and run information in real time leveraging this collaborative infrastructure. Using LoadRunner Enterprise, you can concurrently execute and monitor multiple tests from any location or schedule them to start unattended. Relevant testing assets such as test scripts, load test configurations, test data and analyzed results are stored in LoadRunner Enterprise for easy access, sharing, and reuse.

Performance engineering for both cloud and on-premises solutions

LoadRunner solutions also offer a cloud-based solution for extreme scale and flexibility. With LoadRunner Cloud, you can easily plan, run, and scale performance tests. Rapidly execute cloud performance tests, easily scale more than five million virtual users without the management overhead, and run multiple tests simultaneously with no test concurrency limits.

Bridge Gaps with Functional Testing

Gaps in testing coverage happen when people aren't aligned. Shift-right testing empowers subject-matter experts and QA engineers to close these gaps. Without scripting knowledge, they can build and execute tests.

The OpenText Business Process Testing solution within our UFT family speeds up the journey to shift-right. Its simple design makes tests easier to read and maintain. Now you can replace programming with an intuitive, scriptless interface that builds manual and automated test assets.

Using keyword-driven testing and data driving, business analysts can mirror business processes by interacting with the application under test. Once tests are captured, users can execute them as manual tests. Or quality engineers can even integrate them into automated tests.

Other benefits of using Business Process Testing allow you to:

+ Generate test plan documentation automatically.
+ Enable test versioning and baselining.
+ Define pass or fail conditions via component criteria for logical requirements coverage.
+ Centralize test maintenance so that application changes automatically spread through automated test assets.
+ Support input parameters and data driving during execution.
+ Provide a framework for building user acceptance testing (UAT).

Adding Shift-Right Testing to Your App Production

Shift-right testing complements your shift-left testing practices, improving your overall testing strategy. Shift-right testing allows you to find bugs and other issues in production so that you can fix them before they become a real problem for customers.

The OpenText LoadRunner platform is the ideal way to shift your testing to the right. OpenText LoadRunner solutions simulate load, device versions, and other real-world conditions—without the cost of purchasing physical testing devices.

Learn more about the LoadRunner family and request a free trial or a demo of the solution that best fits your testing needs.

Learn More about LoadRunner Family

opentext™