InformationWeek CONNECTING THE BUSINESS TECHNOLOGY COMMUNITY

AUGUST 2020

Why Performance Testing is Crucial Today

Implementing the right kind of performance testing is as much an art as it is a science. Why? Today's applications are different. They are built with containers and call on microservices. They're intended for desktop, tablet and smartphone users. Performance testing must take all this into account, absorbing some functional testing tasks and keeping the user experience in mind. To control costs and speed development, performance testing must be done earlier in the development process, and it must be automated and continuous.

informa tech

INSIDE:

NEXT

Microservices Applications Spur Shift-Left Performance Testing

Getting performance testing right, while avoiding over testing, will give your organization a significant competitive edge.

By Stan Gibson, Technology Journalist

Everyone remembers their first reading experience. For many young people today, it's Reading Wonders, an online textbook and reading guide developed by McGraw Hill. Far more than an eBook, Reading Wonders is an internet-based distributed application, built with Docker containers, Kubernetes orchestration and running on Amazon Web Services (AWS) Elastic Container Service (ECS). Creating such a complex application and assuring that it works flawlessly is no simple task. Rigorous performance testing throughout the development cycle is required.

"There are more things that can break down. Everything is a microservice. Performance issues are more likely because you are dealing with multiple entities," said Vivek Koul, engineering manager for performance and reliability at McGraw Hill. The advent of microservices-based applications has led Koul to schedule performance testing earlier in the development process – so-called "shift-left" testing. "When developers make code changes, they want to test immediately. Using shift-left methodology makes it easy to address performance issues early in the process," Koul explained.

Performance testing comes in six flavors: load, endurance, volume, scalability, spike and stress. It was formerly a late-stage exercise in the development process, often intended to verify that an application could handle large numbers of simultaneous users. That approach was fine in its day, but it won't do when it means taking apart a complex, container-based application late in the game and putting it back together so that it works.

"The way you test applications that are built with containers and Kubernetes is totally different from the way you test traditional monolithic applications," said Jim Rapoza, research di-

PREVIOUS NEXT

rector at Aberdeen. “A lot of applications are ephemeral – they don’t exist until somebody tries to use them. You could start with only three lines of code, but when a user interacts with them, a hundred functions might spring into action,” he said.

“Performance testing was focused on the back end – the application server, and how fast were database queries,” said Joachim Herschmann, senior research director at Gartner. “Today, we’re in a highly distributed microservices world. Applications have sophisticated web and mobile front ends. The back ends consist of many components and services that might come from Google or third parties,” Herschmann said.

Performance testing comes in six flavors: load, endurance, volume, scalability, spike and stress.

Shift-left has re-established the importance of testing in the development process, according to Nancy Gohring, senior analyst at 451 Research. “People dropped performance testing because it took too much time. That’s not the best idea. The idea [behind shift-left is] to insert load testing throughout the development process,” she said.

GETTY IMAGES

This sea change in application performance testing is due to the need to think in terms of the user’s experience above all else. Because users employ different browsers on a wide variety of desktop, mobile and smart phone devices, developers must prove from the beginning of the development process that applications work equally well on all of them.

“We handle over 1,000 unique browser and native combinations for testing purposes,” said Claude Jones, senior director of software engineering at Walmart Labs, the research and development wing of the giant retailer. Jones is responsible for the technology underpinning Walmart customers’ omni-channel experience. His teams conduct performance testing on software components as well as the end-to-end customer experience across Walmart’s apps, whether desktop, mobile, web, iOS or Android.

“IT used to think things were running fine, but they weren’t, from the end user’s perspective,” said Herschmann. “Now, JavaScript-heavy web front ends handle a lot of the processing. Just looking at the back end for performance is not enough,” he added.

Small bits of latency can prove fatal to an application, according to Rapoza. “It used to be if something took three seconds to load, people would abandon. It’s less time than that now, and it’s more acute for mobile applications,” he

said. “Even if you are a small store, users will compare you to Amazon, Facebook and Apple. They expect any other application they use to work as well.”

The losses caused by small amounts of application latency have been well documented. The BBC, for example, found each additional second it took its site to load resulted in the loss of an additional 10% of users. Pinterest, meanwhile, discovered that reducing perceived wait time by 40% yielded a 15% increase in search-engine traffic and sign-ups.

GETTY IMAGES

Shift-left testing is entering the development scene at a time of upheaval. It’s part of the continuous testing, DevOps, and DevSecOps trends that have been ongoing for the better part of the last decade. And it frequently includes characteristics of functional testing, formerly a separate realm. “Functional and performance testing have sort of merged together. What good is a functional test if you don’t know how it will work in production, and vice-versa?” said Zach McCormick, engineering manager at Braze, a provider of software-as-a-service (SaaS) marketing automation services.

As developers shift left, they are discovering what works – and what doesn’t – in the complex blend of human inventiveness, powerful tools, interpersonal dynamics and business management that is application development.

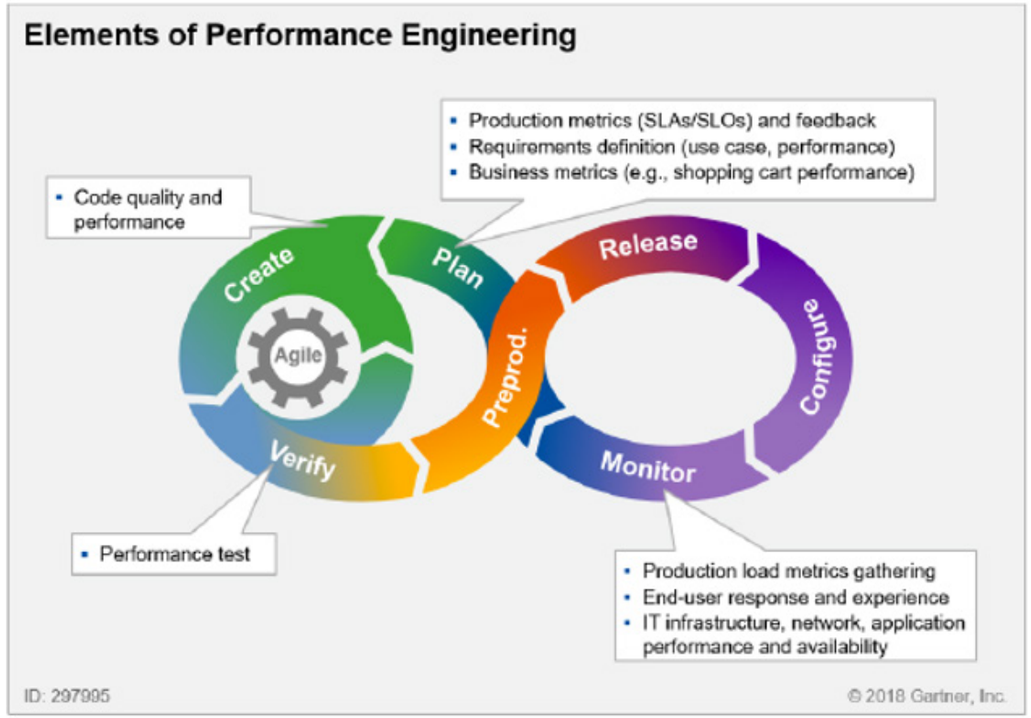
People and Processes

“Shifting left is a mindset. It requires more accountability on the developers’ side to take ownership of what they are testing – in addition to allowing product [managers] and the business to factor into testing as part of the software development life cycle (SDLC),” said Walmart Lab’s Jones. Involving multiple participants requires clear rules for the different players to follow. “Be sure to set clear best practices, guidelines, and standards to help your organization

PREVIOUS NEXT

adjust," he advised.

Similarly, McGraw Hill's Koul stressed the need for teamwork. "You need to make sure that performance engineers are geared to match the ways your development is happening. They can't be a bottleneck – collaboration is key across teams," said Koul. New skills and procedures are also needed. "There is a learning curve when you move from monolithic applications to microservices," he said, explaining McGraw Hill has settled on a four-stage process. First, shift-left testing requires isolated microservices to be tested early in the development process; second, microservices are integrated with one another and tested; third, end-to-end testing is performed, including the user's browser interface; fourth, chaos testing is performed, in which applications are confronted with random events. "Chaos testing is very important for microservices, because it generates scenarios that are anomalies," explained Koul.



Source: Gartner (June 2018)

At Braze, a team dedicated to microservice infrastructure focuses on the interdependencies within the new breed of applications. "The concept of graceful failure is a first-class citizen in the design on those microservices. If one of them breaks, what happens?" said McCormick. The merging of functional and performance testing – and the mingling of development and testing staff – have resulted in some unexpected, and welcome, results. "At cross-team brainstorming meetings we have come up with strategies to test at scale – performance and functional – and for improving our internal quality assurance (QA) process. The tools to test at scale are also useful for debugging in development and production. There are lots of wins you get from functionally correct performance testing," said McCormick.

Tools and Techniques

According to Jones, his biggest challenge is code instrumentation – coming to grips with the vast array of metrics to measure – whether testing the front-end, back-end, web or native platforms. Each might require separate tools. "The proliferation of data and tools makes it difficult to align on a performance testing strategy," said Jones. The answer,

GETTY IMAGES

he said, is to standardize on a common set of tools for capturing, measuring, and analyzing test results, finding the right balance between open source and custom-built tools, he asserted.

In handling a myriad of possible user configurations, Walmart Labs processes millions of records through its data pipeline and has a tightly integrated custom testing platform built with open source code that ties into its continuous integration/continuous development (CI/CD) pipeline, according to Jones.

Braze is also finding a custom approach to be advantageous. “Homegrown tools are the star actor today,” said McCormick. While the DIY approach carries a cost in the time and effort, tools that are specifically tailored to developers’ distinct needs generally yield the best and most economical results in the long run, he said.

It’s important to test high volumes of user requests, both correct and incorrect, McCormick said. “We had one customer in particular who was sending a high volume of invalid requests and we had an extreme performance regression for invalid requests. They were sending them in the wrong format. That’s the difference between a single very fast call to mem-cache, and 2,000 to 3,000 calls to mem-cache. So, don’t just test the happy path, but test your error cases, too,” McCormick advised.

Cloud-based tools can be a good choice for shift-left testing, according to Diego Lo Giudice, vice-president and principal analyst at Forrester Research. “Since more applications are native to the cloud, it makes sense to test them in the cloud. You can’t create a simulated environment on-premise.” He said the use of cloud-based testing tools that are themselves container-based and that can be launched and scaled up and down quickly saves money.

Shift-left practitioners are reporting a variety of benefits, not the least of which is speed.

Shift-Left Payback

Shift-left practitioners are reporting a variety of benefits, not the least of which is speed – a critical success factor in an era when business opportunities appear and disappear quickly. “This gives you faster feedback. Development velocity gets a big boost,” said Koul. There are also benefits in terms of culture, he added. “People become more aware of performance. They think of performance as a more important part of the SDLC.” Handing testing to developers permits them to do small-scale testing early on, and address issues quickly, he said. “If you have a lot of feature

GETTY IMAGES

development happening in your SDLC pipeline, performed by multiple scrum teams, then you have to shift left."

Because Braze enables its customers to send out marketing emails in large quantities, small additions to an application can cause costs to multiply quickly. Shift-left testing reveals which modifications might be worthwhile. "We want to understand when we release a new feature what it might cost us," said McCormick. Conversely, shift-left testing helps McCormick understand the cost savings that result from an algorithm that reduces the amount of data that needs to be transmitted or stored. In one case, Braze reduced the usage of RAM by "20%-30%," yielding savings of over $1 million on API servers, according to McCormick.

The benefits of shift-left free up engineering resources to focus on feature development.

At Walmart Labs, shift-left performance testing has increased developer efficiency, saved time and increased the scale of testing. According to Jones, Walmart Labs logs over 625,000 automated developer hours a day, the equivalent of 75,000 engineers doing manual testing. "By

encouraging the prioritization of testing efforts earlier in the development cycle, coupled with automation, the benefits of shift-left free up engineering resources to focus on feature development," he said. As a result, issues are identified earlier, reducing the amount of time to put code into production by 40% - 45%, and improving the overall customer experience by delivering a higher quality product, Jones explained. But results alone aren't enough, Jones continued. It's necessary to measure and communicate tangible benefits. "Make sure you have outlined clear thresholds and goals and are able to provide insight on how your company is trending over time. This will help build confidence in the overall approach," he said.

Shift-left testing is demonstrating its worth in enabling developers to build highly functional applications quickly and cost-effectively, offering better customer experience and faster response times.

Bottom Line

Today's applications must be developed quickly, they must deal with diverse data sources and large volumes of online traffic, and they must provide a top-notch user experience on a variety of clients.

While retailers such as Walmart and automated marketing providers such as Braze deal in user experience across millions and even billions of customer interactions, great performance is no less critical for schoolchildren who are embarked on the journey of learning — increasingly a 100% online experience through applications such as Reading Wonders. The stakes are high. Small amounts of latency can cost millions in lost revenue — or can play havoc with students' ability to access, understand, and retain information.

Shift-left testing is demonstrating its worth in enabling developers to build highly functional applications quickly and cost-effectively, offering better customer experience and faster response times. Not to be overlooked, however, is the simple, fundamental goal of all testing: "The biggest benefit is confidence," said McCormick.

Stan Gibson is an award-winning editor, writer and speaker, with 37 years' experience covering information technology. He is currently principal at Stan Gibson Communications.

GETTY IMAGES

Quality, Speed Not Mutually Exclusive with DevTestOps Approach

For experienced IT teams who are steeped in DevOps practices, shifting to DevTestOps requires a change in mindset.

Nancy Kastl
Executive Director of the Testing Services Practice, SPR Consulting

DevOps is a tech industry game-changer. It speeds up the time between developing software code (Dev) and making the new code operational (Ops), saving time, money and resources along the way. But the DevOps process often overlooks a crucial step — testing — because normal testing requires time organizations don't have.

Insufficient testing in the development process can produce buggy software that needs revisions and updates post-production. More often than not, these corrections require more time than it would take to test the code during the development process. Organizations currently waste about $300 billion a year fixing bad code — a common outcome in development work that lacks adequate testing.

To address this issue, organizations need to adopt a "DevTestOps" mindset. This approach incorporates testing within the DevOps process to discover and correct code bugs before the software is deployed. With DevTestOps, your organization can retain the benefits of DevOps (e.g., speed and lower costs), while improving the quality of the code and the integrity of the development process.

A different approach for better results

The inclusion of testing in your DevOps process may involve an initial set-up investment. But verifying the code's functionality, integrations, data integrity, security and performance in parallel with development activities identifies bugs early in the process, preventing project delays.

For Agile projects, it's simply no longer acceptable for testing to lag a sprint behind development activities. Instead, organizations need to adopt a new, all-encompassing approach to development that integrates testing throughout the entire workflow. This includes the creation and maintenance of test environments that encourage development teams to frequently deploy builds, so software testers can test early and often. But continuous integration and code deployment can be cumbersome and time-consuming without the right tools. That's where many organizations turn to automation, according to the latest World Quality Report:

- 19% of survey participants use automated tools to help decide what tests are really needed
- 11% said they integrate all test cases in

the CI/CD pipeline

- 6% of respondents automate all tests

Your automated tests must be suitable for the speed of your DevOps process. It's important to ensure your tests are modular and streamlined for speed, reliability, and maintainability. Test engineers should partner with developers to create automated tests with quick feedback loops, so automation is an integral part of the sprint. Automated testing is essential within your DevOps process to gain the desired speed without increasing error.

GETTY IMAGES

Guidelines for DevTestOps

A DevTestOps approach starts with internal conversations that educate and incentivize leaders and decision-makers. But it also requires proper research to implement a personalized strategy for your organization. When you're ready to take the initial steps toward a DevTestOps approach, consider the following guidelines:

1. Change the culture through education. To create a DevTestOps culture, your C-suite and DevOps leaders must agree that quality code is just as much a priority as quick deployments. Work closely with IT and development managers to define a testing strategy that leadership can buy into. This strategy should include the types of tests (API or UI), your test coverage goals, a best-fit test automation framework, test stability, test data management, test results feedback and — most importantly for the C-suite — ROI and risk mitigation.

2. Rethink speed. Time is one of the biggest deterrents to testing, so it's important to debunk the belief that speed and quality are mutually exclusive. Automated, short tests can deliver just as many meaningful insights

GETTY IMAGES

as long-running, end-to-end tests, but short tests can do it in minutes, not hours. Identify your organization's biggest hurdles and design a strategy that deploys tests (large and small) to overcome those roadblocks.

3. Automate strategically. While most companies prefer test automation, 50% struggle with applying test automation at appropriate levels, according to the 2019-2020 World Quality Report. Many existing automated tests tend to involve long execution times not suitable for short DevOps processes, so it's crucial to invest in the right tools and skilled resources for your business. Appropriate test environments and test data are required for the stability and repeatability of automated tests. Consider building automated tests internally or through a third-party partner with expertise in test automation for DevOps.

4. Consider application integration. Today's businesses have multiple applications inter-connected through a transaction flow. You must test all relevant integrations to ensure the integrity of the flow of data across your applications. For DevTestOps, this means your software and various applications' data environments must be in sync so testing can occur both within and across applications. In many cases, service virtualization or similar approaches can help simulate application integrations for testing purposes.

For experienced IT teams who are steeped in DevOps practices, shifting to DevTestOps requires a change in mindset. By refocusing on software quality and the role that testing plays, you can shed light on the gaps that need to be filled to accomplish a fully streamlined workflow. Although this adjustment requires an upfront investment in resources, time and energy, DevTestOps supports the ultimate goal of our industry: to deliver high-quality software quickly and at a low cost. Inserting "Test" in the middle of DevOps can serve as a powerful resource for helping you achieve the quality goal.

For experienced IT teams who are steeped in DevOps practices, shifting to DevTestOps requires a change in mindset.

Nancy Kastl has more than 25 years of experience as a consultant, manager, facilitator, and instructor in strategic planning, quality management, test management, measurement, project management, and process re-engineering at SPR Consulting.

How to Keep Pace with Agile Development Trends

The Agile model and its adopters are moving forward rapidly. Here's how you can stay on top of the latest improvements.

John Edwards
Technology
Journalist & Author

Living up to its name, Agile development methodology is evolving swiftly and nimbly. Hardly a week passes without news about a fresh Agile approach, tweak or innovation. Agile development's rapid evolution makes staying on top of emerging trends ever more important for developers and managers.

Keeping pace with the latest changes can be challenging, but it's well worth the effort, said Cole Cioran, applications practice lead, Agile development and management, at technology research firm Info-Tech Research Group. "Agile practices are entangled with the latest developments in software engineering," he noted. "Developers who follow Agile will also be exposed to practices such as DevOps, microservices, Dapps [decentralized applications], and much more."

Feeding the need to know

Separating key trends from background clutter is essential for anyone who hopes to become Agile-sharp. Kirstin Slevin, technical program manager at intelligent DNS and Internet traffic management technologies firm NS1, believes that extending and scaling Agile's scope is a leading trend.

"Agile is typically thought of as a process that works for software engineering teams within startups and small organizations," she said. "However, the principles of Agile can also scale and be effective for organizations of all sizes." Slevin added that Agile even has the potential to extend its influence beyond engineering to other enterprise departments, such as legal, finance and HR. "It's still relatively early days in terms of these types of adoption, but the potential is there," she noted.

iSTOCK

Agile adopters are rapidly moving toward optimized methods, processes, and testing practices designed to speed project development and delivery, said Hasan Yasar, an adjunct professor at Carnegie Mellon University's Heinz College of Information Systems and Public Policy. "Everyone wants to shrink lead time to the minimum required set of processes in order to meet market demand in the correct timing," he said.

GETTY IMAGES

"This requires the ability to respond on a dime, just like an athlete." Developing such a capability is now paramount, he added.

Cioran observed that many old beliefs are now falling by the wayside, noting that Info-Tech's current research dispels the "Waterfall bad/Agile good" trope. "Waterfall practices are a foundation to build the discipline and relationships with business and customers that are essential for success with Agile," he explained. "While many joke about 'WaterScrumFall', 'Agifall', and so forth, these are natural steps in the evolution of Agile practices and the evolution from the prescriptive waterfall-based practices common in project-centric organizations to the product-centric view built into Agile."

Research approaches

Every Agile expert has his or her favorite ways to stay on top of current developments. Yasar prefers the tried and true methods. "Using Google or other search engines is likely the most effective way to identify the latest Agile trends," he said.

The blogs and commentary provided on websites such as Agile Alliance and in Reddit forums and sub-forums are great sources to discover what's currently happening in Agile, advised Sanjay Chandru, DevOps director for IBM's Z mainframe hardware line. He also observed that keeping up with SAFe and other relevant certification programs has allowed his developers to expand their Agile knowledge. "It helps them grow in their current roles and sets them up for future opportunities as they gain experience," he explained.

Agile development trends are, by nature, constantly changing.

NS1's Slevin enjoys participating in Twitter discussions. "If I have a spare 20 to 30 minutes, I'll check out some discussions and often uncover new people and sources to follow," she said, noting that she's currently fascinated by the controversial Twitter conversations using the hashtags #NoAgile and #NoEstimates. "It's also beneficial to mix in other sources, like joining Slack and LinkedIn groups, attending agile development-focused meetups, and reading up on the latest blogs and books to get a variety of perspectives," she added.

Agile development trends are, by nature, constantly changing, observed Casey Gordon, director of Agile engineering for Liberty Mutual Insurance. "Fortunately, at-

tending local communities of practice, following leading 'Agilists' like John Cutler, Mike Cohn, and Stefan Wolpers, and slacking directly to teams like Hands-on-Agile, are all effective approaches to staying up to date with the latest trends," he explained.

GETTY IMAGES

Takeaways

Trends can be fleeting. "What's constant is the need to continuously improve based on continuous feedback," Chandru said. Understanding that innovation is constant, and keeping up with emerging and evolving trends, can be greatly beneficial to one's job performance and career. "The key is to adopt tried and tested methodologies that are backed up by empirical proof of benefits," he observed.

As they continue their journey, it's essential for Agile followers to ensure that they're working, thinking, and moving in tandem with their business partners, Gordon noted. "Once you've succeeded in creating an empowering and collective model to deliver, it's amazing the progress you can make."

John Edwards is a veteran business technology journalist. His work has appeared in The New York Times, The Washington Post, and numerous business and technology publications, including Computerworld, CFO Magazine, IBM Data Management Magazine, RFID Journal, and Electronic Design. He has also written columns for The Economist's Business Intelligence Unit and PricewaterhouseCoopers' Communications Direct. John has authored several books on business technology topics.

7 Ways to Improve Software Maintenance

Here are some approaches and steps organizations can take to perform software maintenance while creating as much time as possible for new software development.

Mary E. Shacklett
President of
Transworld Data

In 2019, Tidelift, an Opensource support and maintenance organization, conducted a survey of software developers that revealed that developers spent less than one third of their time (32%) developing new code. In the same survey, developers said that 35% of their time was spent on software maintenance.

My own experience in consulting with companies is that the amount of time spent on software maintenance is closer to 50%. In either case, the time spent on maintaining software prevents organizations from pursuing new projects and getting things done. At the same time, maintaining the software that you have created or inherited is a fact of life.

Software maintenance is defined as "a part of Software Development Life Cycle. Its main purpose is to modify and update software application(s) after delivery to correct faults and to improve performance. Software is a model of the real world. When the real-world changes, the software requires alteration wherever possible."

Given this, what steps can organizations take to perform software maintenance while creating as much time as possible for new software development?

1. Listen to your help desk. No function in IT has a better finger on the pulse of application performance than the help desk. The help desk gets all of the questions and problems from users. The people who work the help desk know from the calls they get which applications are most problematic, and why. If more IT organizations patched help desk insights into their application development brainstorming and performance evaluations, they would be more successful identifying areas of persistent application problems and failures so these areas could either be addressed fully by repairing them or retired and replaced with another solution.

The people who work the help desk know from the calls they get which applications are most problematic, and why.

Just as importantly, the knowledge gained from application trouble "hot spots" at the help desk can be learned from so that the same mis-

takes aren't repeated in new software development.

2. Engage QA. In too many organizations, developers up against tight deadlines tend to throw their work "over the wall" to QA at the last minute. Then, only partial application testing gets done before the app gets deployed into production. When the app goes live, there can be weeks of problem reports and troubleshooting, with fixes and workarounds resulting. Conversely, by thoroughly testing applications upfront for technical correctness, integration and usability, post-production software maintenance can be drastically reduced. To facilitate this, project managers need to plug in and ensure adequate times for software QA.

3. Consider a move to the cloud. Organizations using broken on-premises legacy software can consider making a break from endless maintenance by moving to a cloud-based version of the software that is offered and supported by the vendor. In a scenario like this, software maintenance is moved out of the shop and into the hands of the vendor. One disadvantage is that you never can be sure when the fixes or enhancements you want are going to get done — but the move could well be worth it if you can live with the inconvenience.

4. Sunset the applications that aren't returning value. Almost every organization has a legacy system that no longer delivers the value it once did. This is a time to consider sunsetting that system and potentially planning a "rip and replace" with a new system. Rip and replace works when there are few needs to integrate the system with other software that is running. In cases where rip and replace is viable, you can shift much of your system maintenance for the new system to the supporting vendor.

5. Always regression test. The impulse when you're under the gun to finish a project is to meet deadline and skip some of the quality tests. One critical test is the regression test, which places any application that is newly modified in a simulated production environment with other applications to test and ensure that integration with these other applications and called routines is working properly. When regression testing is skipped, risk heightens that a newly modified app will break or cause other pieces of systems to break because of a coding error that was introduced. This brings down systems and causes service outages.

6. Use a standardized procedure for installation of new software releases. Whether it's a security patch or a feature addition or fix, the installation of a new software

GETTY IMAGES

GETTY IMAGES

release should be uniform and standardized as much as possible for all the devices the software runs on. This keeps the software release process organized and relieves the pressure on the help desk and maintenance teams when software releases are not systematically distributed, and different devices are running different releases of the software.

7. Optimize your software maintenance team. There are people who love doing software maintenance, and they should be encouraged and rewarded for their work. Software maintenance is also a great area for new employees to begin their careers, because they get a thorough introduction to a variety of systems that the company runs. This will benefit them later, whether they remain in maintenance or move to another part of the IT organization.

Mary E. Shacklett is an internationally recognized technology commentator and President of Transworld Data, a marketing and technology services firm. Prior to founding her own company, she was Vice President of Product Research and Software Development for Summit Information Systems, a computer software company; and Vice President of Strategic Planning and Technology at FSI International, a multinational manufacturer in the semiconductor industry.

The Critical Need for Performance Testing

The days of “tossing a bunch of apps over the fence to QA” and seeing performance testing as an afterthought are over.

By Vicky Giavelli, Director of Product Management, Performance Engineering, Micro Focus

Over the past few months, we’ve all personally and professionally felt the impacts of life under uncertainty. I’ve spoken to customers in almost every industry as they adapt – from corporations ensuring employees stay productive remotely, to healthcare teams building new websites or tools that improve care, to universities and educational institutions coping with online teaching, to retailers experiencing a surge in online traffic, and government entities developing new tools that support their citizens. Software innovation is foundational to these new realities.

One thing is for certain, organizations will need to reimagine key aspects of their software development approach. Common to all these industry challenges is the need for great performing applications that are resilient and satisfy user expectations. But all too often performance testing is viewed as a last stage effort. Regrettably, saving performance testing to the end results in poor application design, higher costs of change, release delays, and production issues.

The cost of poor application performance cannot fall on quality assurance (QA) alone, and the days of “tossing a bunch of apps over the fence to QA” are over. Performance must become everyone’s responsibility – from the designer, coder, developer, QA tester to the performance engineer – so that teams design better performing software from the start as they quickly root out issues well before production. To deliver at the speed the business requires, organizations need to evolve from a siloed, late-stage performance testing model to one that

GETTY IMAGES

builds quality metrics into the release requirements before development begins.

At Micro Focus, we support a proactive, continuous performance engineering discipline that includes four key attributes: expansion of performance testing to new roles, tight integration into the CI/CD process, end-to-end performance monitoring, and continuous improvement. Combined, these elements ensure that teams can engineer performance early in the lifecycle through the end-user experience.

...engineer performance early in the lifecycle through the end-user experience.

But moving from performance testing to performance engineering isn't an easy process. Collectively, the team must transition from evaluating isolated results to understanding how all parts of the system work together. The overall view must encompass hardware, software, configuration, performance, security, usability, business value, and the end-user. In the end it is about collaborating and iterating on the highest value items.

The shift first requires a new approach to the tools, capabilities and infrastructure used across the organization.

GETTY IMAGES

Running disconnected tools, load generators or analysis capabilities in isolation defeats the intent behind an engineering approach and often leads to unexpected infrastructure and maintenance costs. Ensuring that tools are right-sized for the different users while exploiting test reuse, and managing or delivering shared infrastructure should be the first stop in developing an ecosystem for success.

With the core tools in place and test data coming in, teams need alignment on the best set of metrics and a collective view on how to leverage those results. A centralized approach to data collection allows teams to connect the dots between developer, CI and end-to-end performance tests. Incorporating telemetry, APM (application performance monitoring) data from production, and indirect user sentiment rounds out the necessary inputs for a complete performance view. Through data visualization, teams can then view real-time results and manipulate data to make smarter decisions.

The Micro Focus LoadRunner Family delivers all of this and more in an integrated set of enterprise-grade performance engineering solutions. Simply choose the right tool for the right job while leveraging a connected ecosystem that delivers smarter insights, tighter collaboration and better cost savings. The family includes LoadRunner

GETTY IMAGES

Professional for the most technical users; LoadRunner Enterprise for collaborative, globally distributed teams; LoadRunner Cloud for ultimate scalability; and LoadRunner Developer for shift-left testing using any IDE.

With any LoadRunner solution, teams can confidently test complex load, stress, and performance scenarios across legacy, website, and mobile applications – while benefitting from shared capabilities across the products:

- Enterprise coverage: utilize the broadest and deepest technology and protocol support
- Open and integrated: integrate with dozens of tools covering scripting, CI/CD, open source, APM and more
- Scalable and flexible: meet demands without increasing costs across assets, licenses and infrastructure
- Realistic: emulate real-world conditions with embedded network virtualization capabilities
- Intelligent: rapidly pinpoint root cause and decrease mean time to repair using shared smart analytics with real-time insights

With Micro Focus, go beyond traditional performance testing and enable true performance engineering to continuously engineer software quality at any point in the DevOps pipeline.