PARASOFT®

CASE STUDY

Service Virtualization, Performance Testing, and DevOps at Comcast

By Frank Jennings, Director TQM Performance Testing at Comcast

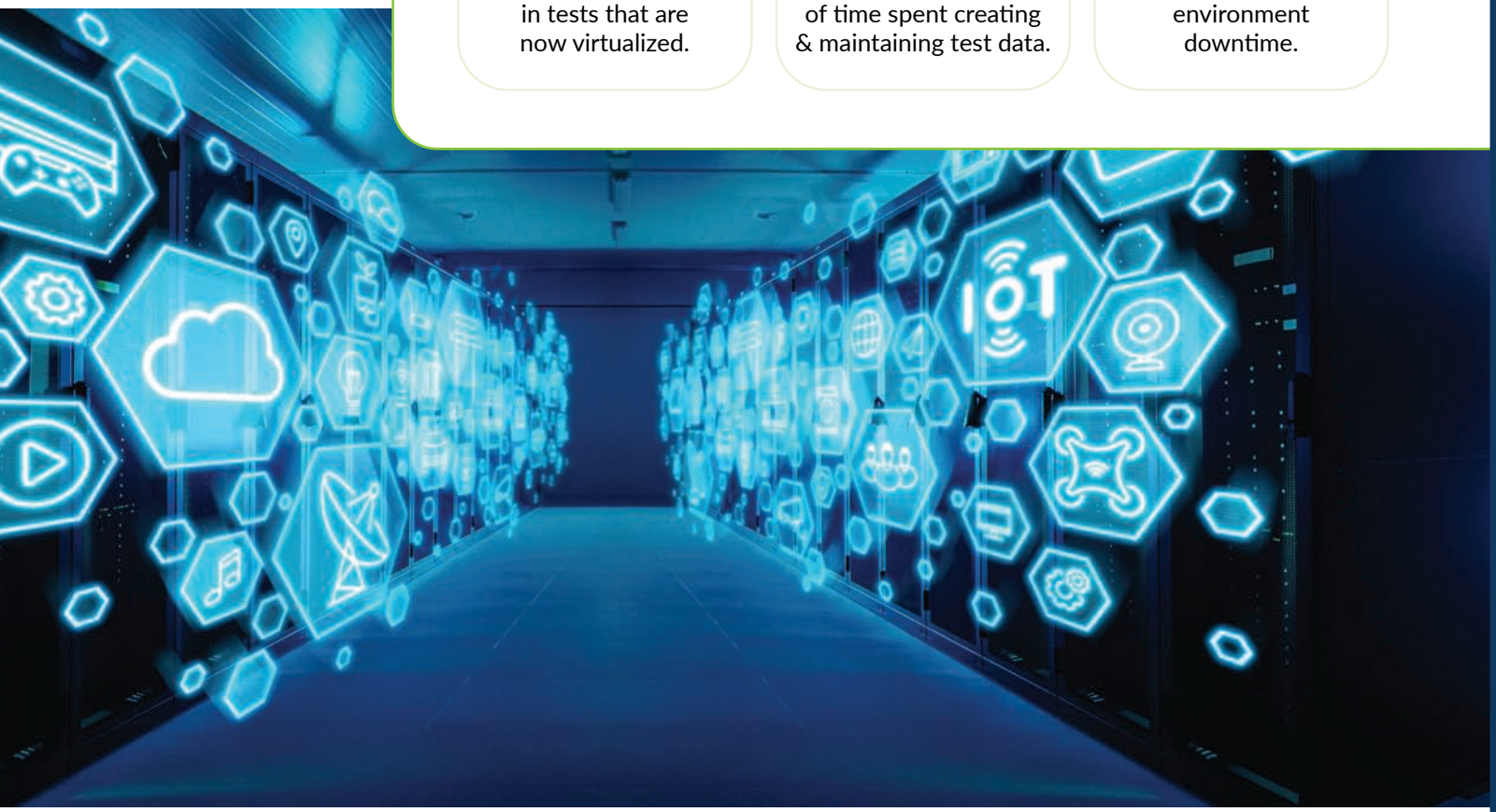
OVERVIEW

Before service virtualization, Comcast's Performance Testing team often ran into scheduling conflicts around sharing the test infrastructure. Sometimes downstream systems were not available. Other times, test engineers would try to run tests at the same time, which could affect the test results. This led to variability between tests, which made it challenging to isolate particular problems. Learn about the results Comcast achieved after successfully implementing service virtualization—and why service virtualization is a key component of our DevOps initiative.

98%
INTERFACES INVOLVED
in tests that are now virtualized.

65%
ANNUAL REDUCTION
of time spent creating & maintaining test data.

60%
REDUCTION IN STAGING
environment downtime.



THE CHALLENGES

My team at Comcast executes performance testing across a number of verticals in the company—from business services, to our enterprise services platform, to customer-facing UIs, to the backend systems that perform the provisioning and activation of the devices for the subscribers on the Comcast network. While our testing targets (AUTs) typically have staged environments that accurately represent the performance of the production systems, the staging systems for the AUT's dependencies do not.

Complicating the matter further was the fact that these environments were difficult to access. When we did gain access, we would sometimes bring down the lower environments (the QA or integration test environments) because they weren't adequately scaled and just could not handle the load. Even when the systems could withstand the load, we received very poor response times from these systems. This meant that our performance test results were not truly predictive of real-world performance.

Another issue is that we had to work around frequent and lengthy downtimes in the staging environments. The staging environment was not available during the frequent upgrades or software updates. As a result, we couldn't run our full performance tests. Performance testing teams had to switch off key projects at critical time periods in order to keep busy– they knew they wouldn't be able to work on their primary responsibility because the systems they needed to access just weren't available.

These challenges were driving up costs, reducing the team's efficiency, and impacting the reliability and predictability of our performance testing. We knew we had to take action—and that's why we started looking at service virtualization. Ultimately, we found that the time and cost of implementing service virtualization was far less than the time and cost associated with implementing all the various systems across all those staging environments—or building up the connectivity between the different staging environments.

MEASURABLE RESULTS FROM SERVICE VIRTUALIZATION

We turned to service virtualization for two main reasons. First, we wanted to increase the accuracy of performance test results. Second, we were constantly working around frequent and lengthy downtimes in the staged test environments.

> ***We used to need two weeks to performance test the code once we got it in our staging environments. We shrunk that to just two or three days.***

Our initial focus was on the biggest pain points in terms of scheduling conflicts within the performance testing teams, unavailable systems, and systems where our testing would impact other development or test groups. Since we started, we've been able to virtualize about 98% of the interfaces involved in our tests, and we saw a 65% annual reduction in the amount of time it takes us to create and maintain test data (factoring in the time we spend creating and updating virtual assets). We also reduced staging environment downtime by 60%.

Since we can start working on our scripts versus virtual assets in the development environment, we typically have everything ready to go quite early in each sprint. We used to need two weeks to performance test the code once we got it in our staging environments (for example, with average load tests, peak load tests, endurance tests, and so on). We shrunk that to just two or three days.

SOLUTION BENEFITS

Our tests are now more predictable, more consistent, and more representative of what would be seen in production. Moreover, we're also able to increase the scope of testing in many cases. For example, we can't put production loads on certain actual services, but when we're working with virtual services we can ramp it up with production-level loads and get realistic responses, both in terms of data and performance. We can really isolate the AUT, not just from a performance testing perspective, but also from a performance profiling perspective. Instead of just telling development, "This is the performance of your system," we can also say, "This is where we're seeing the bottlenecks and this is where we think changes might improve the throughput of the application."

The key benefit for the performance testing team is the increased uptime and availability of test environments. Service virtualization has allowed us to get great utilization from our testing staff, complete more projects on time, and also save money by lowering the overall total cost of performing the testing required for a given release.

SERVICE VIRTUALIZATION AND DEVOPS

Beyond performance testing in our staging environments, we've also been able to use service virtualization for everything from unit testing and regression testing in the development environment, to baseline performance testing in early downstream environments, to functional and regression testing in the QA/integrated environment, to manual/exploratory testing in an environment that's quite close to production (but uses virtual assets in some cases).

All the configuration and deployment of virtual assets for the various environments is automated as part of our DevOps infrastructure. Environments automatically switch between virtual assets and actual assets according to the business rules we've defined—for example, based on what endpoint the traffic is coming from, the data contained in the test packets, and so on. This service virtualization solution has enabled us to achieve continuous testing as an integral part of our DevOps process.

TAKE THE NEXT STEP

Find out how to choose the right service virtualization solution for your organization. Download the whitepaper.

ABOUT PARASOFT

Parasoft helps organizations continuously deliver quality software with its market-proven, integrated suite of automated software testing tools. Supporting the embedded, enterprise, and IoT markets, Parasoft's technologies reduce the time, effort, and cost of delivering secure, reliable, and compliant software by integrating everything from deep code analysis and unit testing to web UI and API testing, plus service virtualization and complete code coverage, into the delivery pipeline. Bringing all this together, Parasoft's award winning reporting and analytics dashboard delivers a centralized view of quality enabling organizations to deliver with confidence and succeed in today's most strategic ecosystems and development initiatives — cybersecure, safety-critical, agile, DevOps, and continuous testing.