# Collaborative Load Testing for Modern Engineering Teams

## Introduction

Major trends and best practices in the software testing industry, such as Agile, DevOps, and Site Reliability Engineering (SRE) challenge traditional testing teams operating in siloed functions. Quality Assurance (QA) teams need to adapt when implementing recent trends in testing such as shift-left, automation, and accepting application failures as inevitable.

Modern engineering teams that have embraced these practices often find traditional load testing solutions extremely slow and inefficient — incapable of adapting to a faster-paced work environment. Any hindrances in the testing phase will slow down releases and negatively impact quality.

> According to McKinsey & Company research, best-in-class tools are the primary driver of Developer Velocity and are the top contributor to business success—enabling greater productivity, visibility, and coordination.

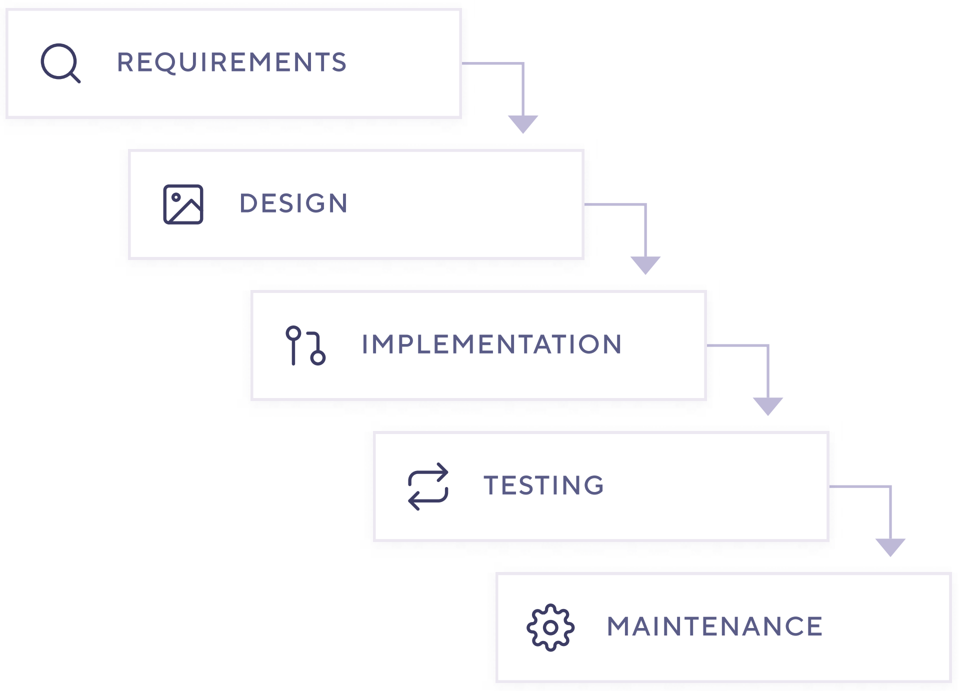
Load testing validates the performance and reliability that customers experience using your service. And today, **reliability and performance are shared responsibilities** across various roles in an engineering organization.

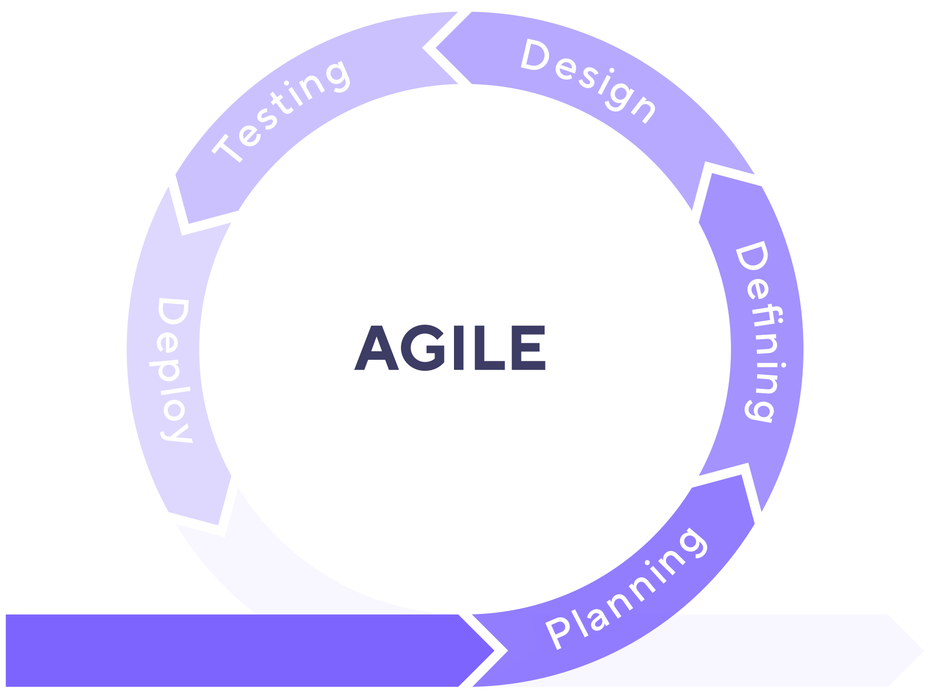
This white paper looks at trends, challenges, and best practices in the software industry that have impacted today's load testing processes. It describes how a developer-oriented and team-centric solution helps engineering teams effectively collaborate on testing to achieve faster software releases while instilling greater confidence in quality.

## Teams using traditional testing solutions face considerable challenges

For a long time, when the waterfall model was a widespread software development practice, performance testing was considered by most organizations a luxury activity. It was only carried out by organizations that could afford expensive performance testing solutions and a skilled performance engineering workforce.

The waterfall methodology has declined in popularity in recent years, yet many testing tools remain unchanged.





The wide adoption of Agile and the rise of DevOps have transformed delivery practices, changing the way we develop, test, and ship software for the better. Today, organizations that follow Agile and DevOps best practices create:

- Better products by experimenting and delivering software more frequently.
- Faster builds of software through better tools and team collaboration.
- Improved digital experiences, focusing on performance and reliability.

These modern engineering practices enable teams to move through the phases of software development faster and build products with exceptional quality. However, without a proper testing solution in place, teams struggle with **ineffective collaboration and the slowness of the testing process**.

> «The ability to access relevant tools for each stage of the software life cycle contributes to developer satisfaction and retention rates that are 47 percent higher for top-quartile companies compared with bottom-quartile performers,» per McKinsey & Company research.

Traditional testing solutions prevent teams from achieving the full potential of Agile and DevOps. Organizations need to provide teams with modern testing solutions and encourage developers to participate in the testing process.

## Work with developers to shift testing left

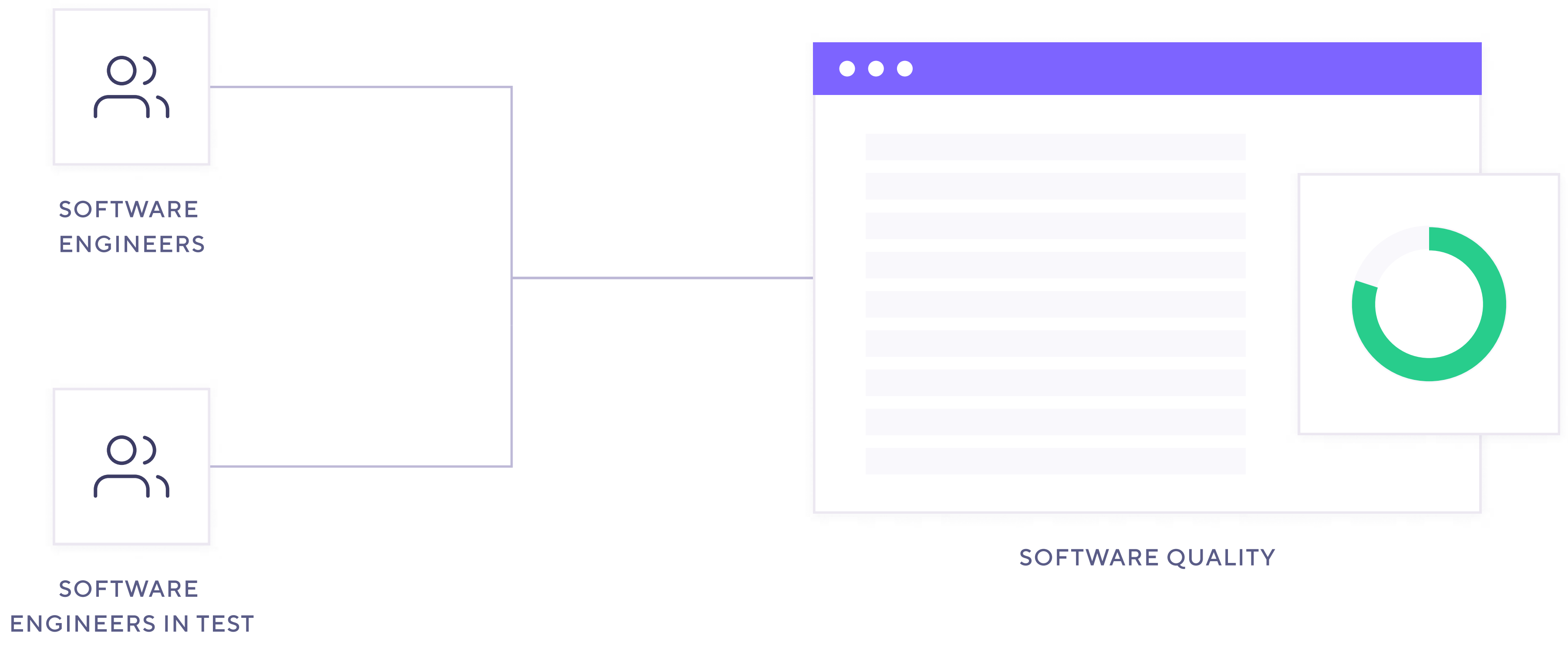
Developers build the software, so they know all the details to test it efficiently. Poor collaboration between the development and QA teams causes significant inefficiencies such as delays in application code and testing due to slow communication and unnecessary software rewrites. Delays are costly, not only when building new features but also when implementing software changes.

To avoid slowing down releases, teams need to bring developer roles into the testing process. Furthermore, testing processes and tools must adapt to developer workflows; otherwise, they will hamper collaboration. Modern testing solutions are designed for developers, and they deliver the advantages of new technologies, unlike traditional, QA-siloed tools.

The popularity of shift-left, where testing occurs early and often in the software development process, has grown steadily in recent years. In combination with Agile, shift-left testing encourages developers to participate in testing, pushing testing continuously into key stages during the software development and delivery process. Both practices make the **testing process faster, more effective, and of higher quality.**

> According to the 2020-2021 World Quality Report by Capgemini, shift-left continues to be the most important aspect to make testing more efficient.

We find that most modern software teams have shifted performance testing left towards development. This shift has changed the nature of engineering roles at organizations. For example, a new QA role called Software Development Engineer in Test (SDET) or Software Engineer in Test (SET) — a hybrid role between the traditional QA and development role — often participates in shift-left testing.

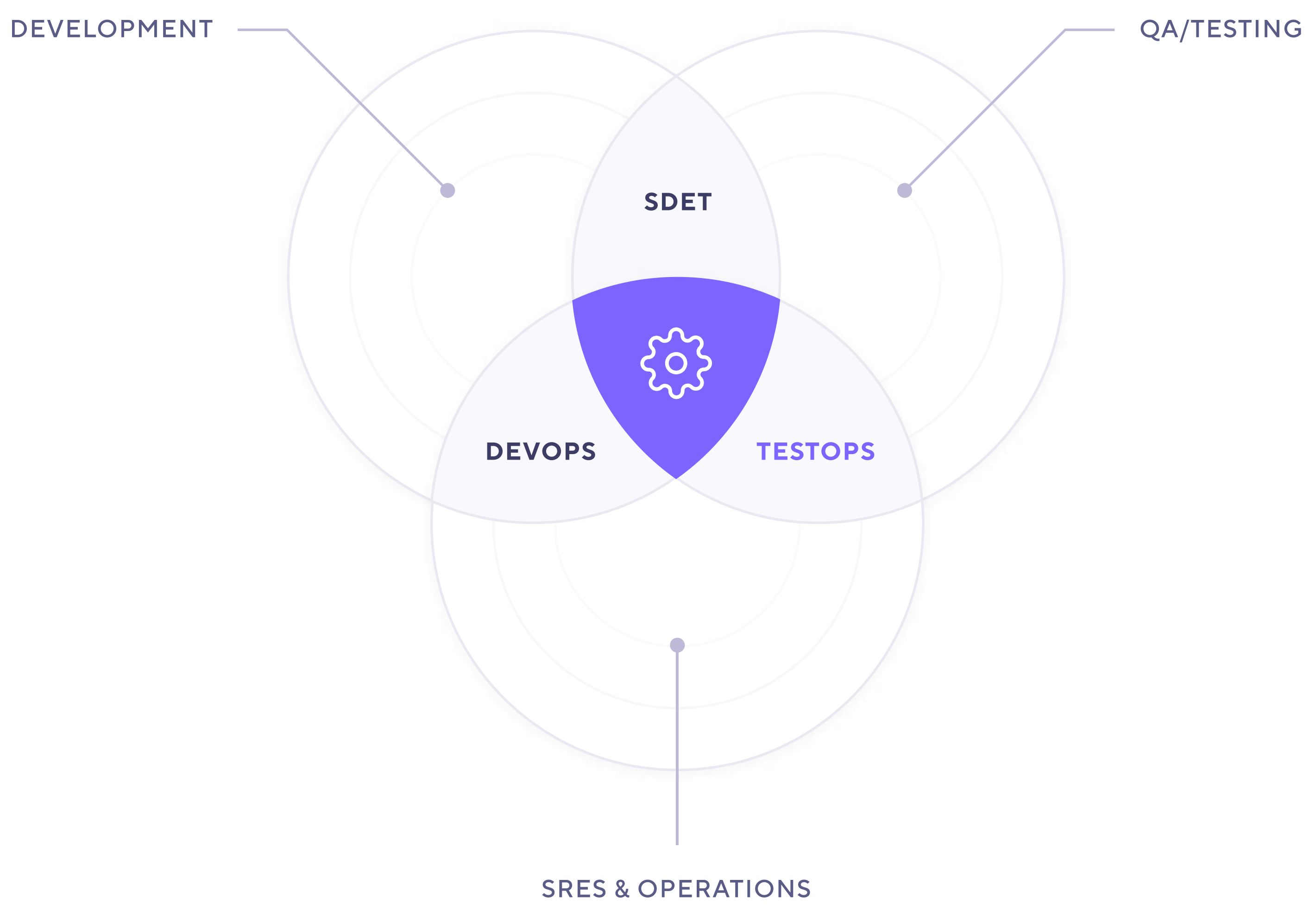


SDETs/SETs do not work in isolation; they collaborate with product managers to better understand the products and to define practical and realistic testing processes. They also actively collaborate with developers to improve the testing-creating, maintaining, and automating the tests. The rising demand for SDET/SET roles is the result of shifting testing left and breaking out of the QA silo.

## Accelerate software delivery with TestOps and continuous testing

In recent years, DevOps practices and automation have radically improved the entire software delivery process, including testing. Automation has eliminated most of the repetitive, unreliable, and time-consuming manual activities of traditional testing. DevOps and automation have allowed organizations to improve test suite quality, team productivity, and software release velocity.

When you have a fast software development and delivery process, engineering teams need a robust and reliable testing suite that can keep pace with continuous development and deployment. Such a testing platform helps teams **ensure the high quality of every release**. To make testing reliable, teams should perform different types of testing across various environments, including development, canary, QA, pre-production, and production. Teams should also automate testing in continuous delivery pipelines to prevent errors when shipping new features or experiments iteratively to the end-user.



Although automation and frequency vary depending on the testing type, continuous and automated testing are now standard practices and the end goal for most types of testing. TestOps is also becoming a widely adopted best practice for organizations that develop software.

TestOps is a discipline that blends testing and operations to support various teams in the testing process, ensuring that testing infrastructure always remains reliable and efficient. The main goal of TestOps is to empower testers to spin up the proper test environments and work with other teams to set up automated testing processes.

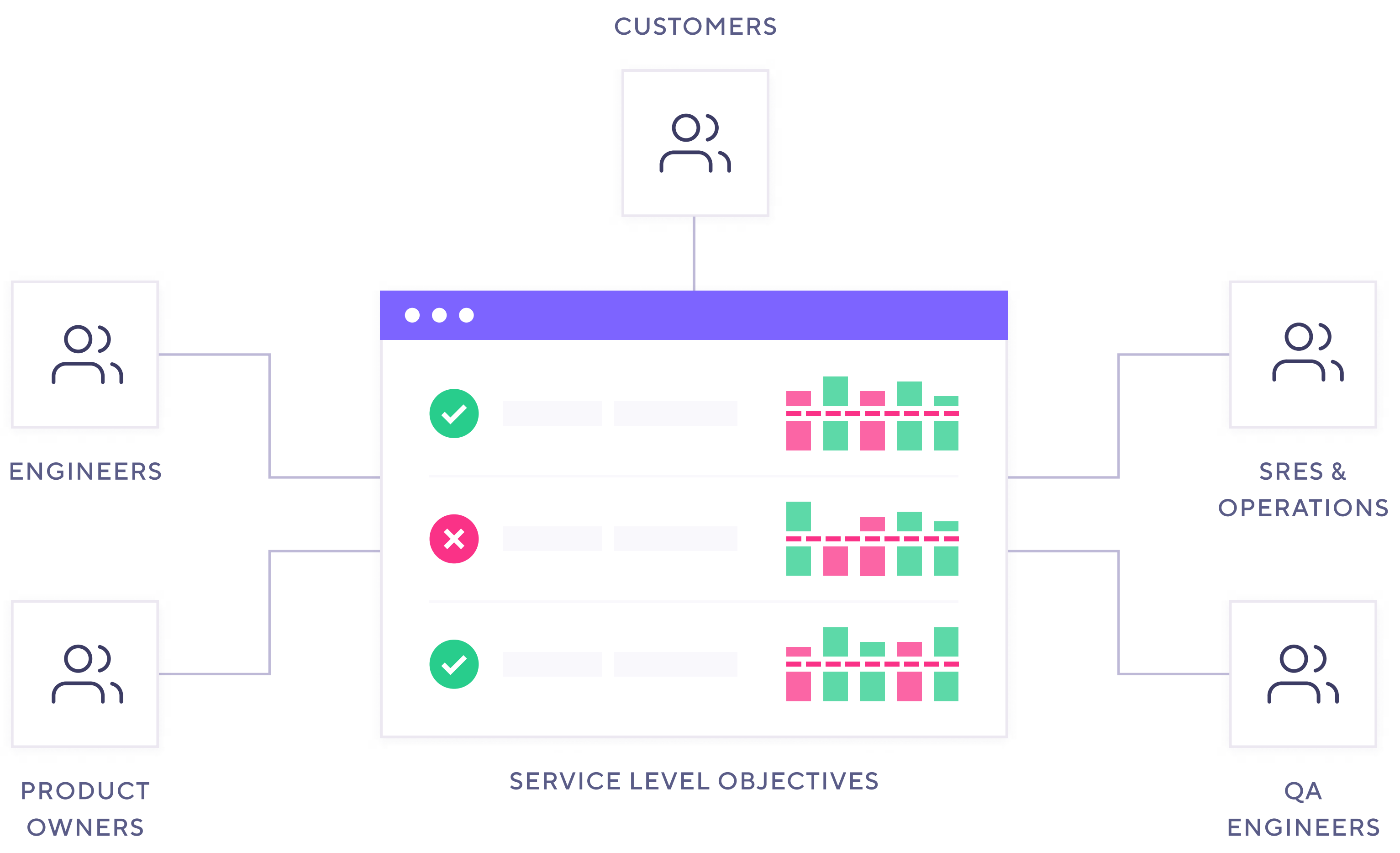
> The technology stack is the top aspect for successful Agile and DevOps automation. 65% of respondents consider the technology stack essential or very important, per the 2020-2021 World Quality Report by Capgemini.

In the past few years, the emergence of TestOps and the increasing need for test automation engineers have led to a growing shift to QA automation. The new roles in testing address the challenges of using modern testing solutions, infrastructure, and QA environments.

## Break team silos with Site Reliability Engineering

Site Reliability Engineering began in 2003 when the production team at Google created processes to achieve maximum feature and change velocity without violating the Service Level Objective (SLO) for any service. Today, the basic goal of SRE is to meet customer expectations in terms of availability, scalability, and performance.

Defining SLOs is fundamental to the success of engineering teams because it aligns core objectives across engineering, product teams, and customers — a common framework for the entire organization in terms of reliability and performance for the end-user.



Because SRE embraces system failures as inevitable, an SLO defines an organization's error budget — the percent of allowed errors for a period of time. Teams usually monitor the error budget in production, controlling the status of the SLO and addressing possible issues before they breach the SLO.

However, rapid software releases often introduce software failures or regressions. Lack of proper testing and working with SLOs only in production are a dangerous combination. Teams should mitigate the risks of near-breach violations by **testing SLOs earlier in the software delivery cycle.**

According to Catchpoint's SRE Report 2020, 30% of SRE respondents were participating in QA testing, while 41% of SRE teams were using testing tools.

When SREs participate in the testing process, they break the silos among engineering managers, QA, and operational teams, incorporating QA into the SLOs shared across the organization.

## Choose the right tool to enable modern testing practices

In order to keep up with the continuing transformation of the testing industry, organizations must align their processes, teams, and tools to the shared goals of performance and reliability. Testing processes, teams, and tools are all connected — failing in one will negatively impact the other two.

One of the main factors restraining cross-functional testing and developer participation is selecting the wrong testing tool. Without the right testing tool, «whole team testing» becomes difficult, if not impossible.

> The 2021 State of Testing Report by PractiTest found that 67% of respondents considered "Whole team testing" challenging or very challenging.

Choosing the right tool, along with best practices and talent management, can unlock the benefits of modern testing practices. However, the testing tool must enable **three critical factors**: developer participation, continuous testing, and collaboration.

### Developer participation

Developers form strong tooling preferences based on their previous development experiences. They choose tools that minimize barriers and boost their productivity. To help bring developers into the testing, a developer-oriented tool should:

- Support a scripting language so developers can create and maintain end-to-end tests efficiently.
- Be open source with a strong community aligned with the principles of the developers using it.
- Integrate well with other open-source tools and developer ecosystems.

### Continuous testing

Agile practices encourage continuous software delivery, and DevOps accelerates software cycles through automation. Testing earlier and continuously will ensure software regressions will not breach SLOs. In this rapid environment, the testing solution should:

- Test SLOs effectively.
- Integrate automated testing into software delivery cycles.
- Alert stakeholders of failures.

## Collaboration

Today, many roles participate in performance testing — SREs, SDETs, developers, QA testers, and business analysts, to name a few. While the bulk of the responsibility of testing usually falls on one team, other roles also incorporate quality metrics and testing into their activities. To enable effective collaboration, the testing solution should:

- Adapt to the workflows of participants from different areas.
- Enable cross-functional team collaboration.
- Provide centralized views, reports, and collaborative dashboards.

## Unlock a rich testing ecosystem with k6

Our teams at k6 have experienced all the challenges highlighted in this white paper. We've built two solutions to solve modern-day testing problems: k6 Open Source and k6 Cloud. Both solutions improve the testing process significantly, bringing cross-functional teams together for fast and effective performance testing.

k6 provides a rich ecosystem to support various team workflows and multiple choices when creating tests. And you can create tests easily with a scripting language or GUI tools. k6 Cloud offers a centralized platform for cross-functional collaboration in testing. Teams can skip the hassle of building and maintaining load testing infrastructure and focus their efforts on improving products.

Our k6 Cloud platform offers:

- **Scalability** - We've made k6 Cloud scalable by running hundreds of machines to scale the tests and process metric data volume.
- **Automation** - k6 natively supports the automation of performance testing, including valuable features like trending, test comparison, and notifications.
- **Collaboration** - Our platform includes key collaboration features, such as role-based access control (RBAC), projects, test concurrency, and advanced sharing capabilities.
- **Rich UIs** - Access to centralized data improves collaboration between managers and engineering roles. k6 dashboards provide an activity overview, and the rich UI gives detailed views of the testing results.
- **Integrations** - k6 Cloud provides additional integrations with monitoring tools and platforms for further test visualization, data correlation, and incident response.