Tricentis

Performance engineering at enterprise scale

Tricentis

Table of contents

1. **Enterprise performance testing: parallel roads to the same destination** 3

2. **The four-way intersection of enterprise performance testing** 5

3. **Siloed expertise doesn't scale** 6

4. **Seven ways to implement a standardized performance engineering approach** 8

    a. Promote deep collaboration 8

    b. Make things easy for non-experts 9

    c. Test fast to release fast 9

    d. Integrate the right tools for the right reasons 10

    e. Automate performance testing in CI pipelines 11

    f. Settle on a solution that covers all the bases 11

    g. Think cloud-native 12

5. **Conclusion** 13

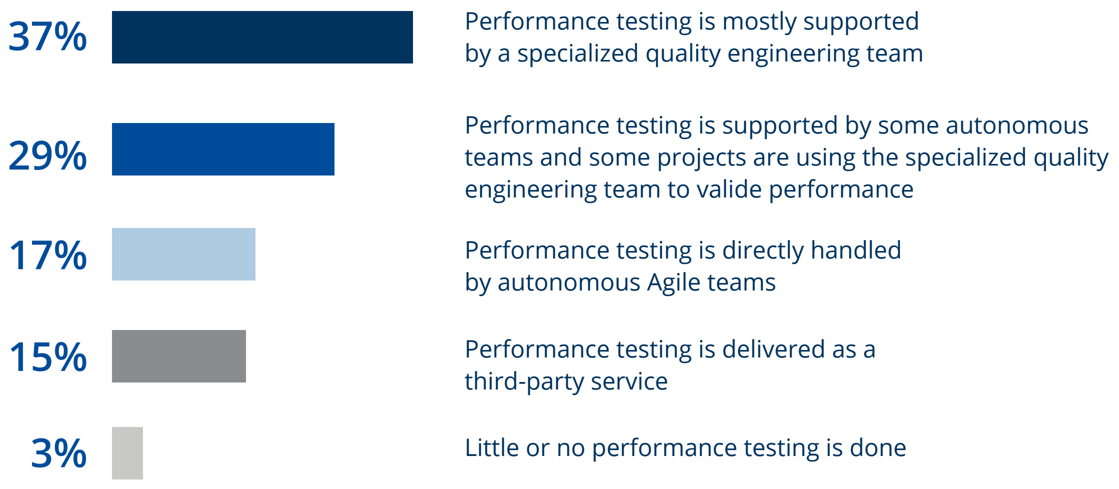
6. **About Tricentis** 14

ENTERPRISE PERFORMANCE TESTING: PARALLEL ROADS TO THE SAME DESTINATION

When you think about performance engineering at scale, scaling virtual users in performance tests probably comes to mind. Putting intense load on an application to ensure it can perform with hundreds of thousands to millions of concurrent virtual users operating under various user scenarios is traditionally what one thinks about when the words scale and performance come up. And that's still very relevant and critical to today's application testing. With DevOps, the meaning of scale is shifting. DevOps' raison d'être is to release frequently, moving from a major release every year to smaller frequent releases, at times daily. In this approach, in order to keep pace with faster release schedules yet still realize quality control, teams need a way to go from to one or two tests per week to dozens. Ten times the number of tests in the same time period? That's scale too. Here, scale is defined by the organization's capacity to execute performance testing as frequently as they have builds and code check-ins.

Enterprises are on a spectrum of DevOps adoption; many have taken an incremental approach. When it comes to application development, the reality is that enterprises use a combination of waterfall and Agile approaches, leveraging both centralized performance engineering teams — often Centers of Excellence (CoEs) — and autonomous DevOps teams. This applies to testing too.

How enterprises handle performance



SOURCE: SOGETI-NEOTYS REPORT, THE STATE OF PERFORMANCE ENGINEERING 2020

The traditional longer-release-cycle waterfall approach is very much alive and kicking within most enterprises. Enterprises are used to having a great deal of control over and predictability about what software goes out the door. When they say "go," they trust that the release will perform as expected. They are less interested in failing fast than in not failing at all. The blast radius is simply too great — especially for mission-critical core applications. These complex, multi-tier, highly integrated applications require end-to-end testing by teams with deep performance testing expertise.

However, competitive pressures demand faster, more frequent releases. Virtually every enterprise is steadily phasing in DevOps projects, where testing is done by distributed autonomous teams who are not first and foremost performance engineers. And as organizations move into DevOps, the volume of tests increases from one or two per day to one or two hundred per day.

How does performance testing that typically takes weeks to complete keep pace?

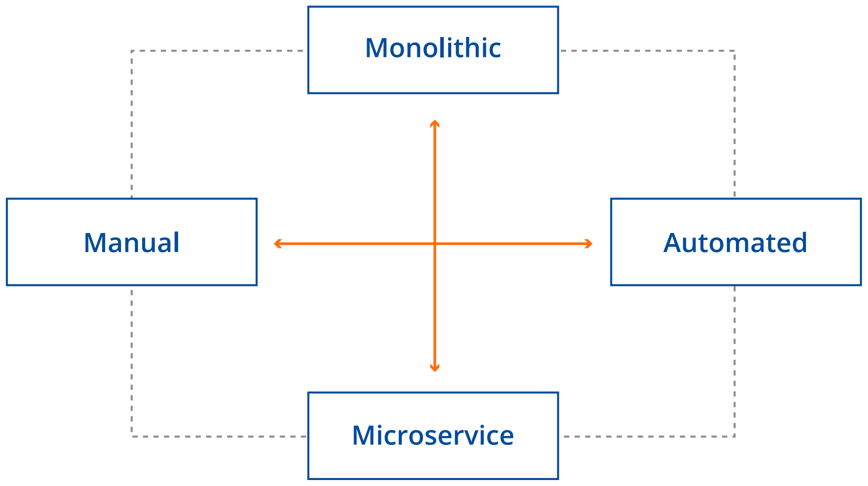
Performance and other QA tools that are natively architected to fit into both new and existing development processes are required. Having both dedicated performance engineering and DevOps teams standardize on a single performance testing platform that works equally well for both gives enterprises the predictability, validation, and assurance that they're used to but at the volume and velocity of automating in an Agile environment.

"The transition to DevOps challenges many enterprises to build out an Agile process where they can release faster but still have the quality controls they are used to. This challenge can be overcome with the right tools that navigate both new and existing development approaches."

THE FOUR-WAY INTERSECTION OF ENTERPRISE PERFORMANCE TESTING

The best way to think about an enterprise's performance testing needs is as a plot-graph quadrant. Along one axis is the methodology approach, with manual testing at one end and automated testing at the other. The second axis reflects the type of applications tested, with prepackaged monolithic "enterprise-grade" applications at the top and modern microservices-based apps at the bottom.

Enterprises have multiple performance testing needs



Manual testing is done by performance engineering experts to test how an application performs end to end, from start to finish. End-to-end application testing occurs late in the software development cycle, because all the various elements and components are tested as a holistic whole. It's quite complex, which is why it's handled by experts with specialized know-how, and time-consuming.

Automated testing occurs early and often. Performance testing is built into continuous integration (CI) pipelines as part of the pass/fail criteria. Autonomous teams of non-expert developers start testing APIs, microservices, and components right at the beginning of the development process and continue testing throughout the evolution of the application.

Monolithic apps are built on a single code base, with a number of different modules. Most commonly, they are highly complex business-critical core systems with many interdependencies. These include homegrown legacy applications and packed enterprise-grade applications like SAP, Oracle, Pega, Salesforce, etc.

"Today's enterprises have a combination of monolithic and microservices apps, manual and automated performance testing, on- premesis and in the cloud — and this will be the case for the foreseeable future."

Microservices-based apps are built on a collection of smaller, independently deployable modular components and services. Different distributed teams are responsible for developing (and testing) their own particular microservice on their own release cycle.

The reality is that performance engineering in today's enterprises involves a combination of monolithic and microservices applications, with both manual and automated performance testing, on-premesis and in the cloud. It's not a black-and-white situation; there are innumerable shades of gray. And the picture is constantly changing: organizations are adopting DevOps, may be migrating enterprise-grade applications from on-premesis to the cloud (think: SAP S/4 HANA) or refactoring monolithic apps to microservices-based architecture.

This is the actual lay of the land, and will be so for the foreseeable future.

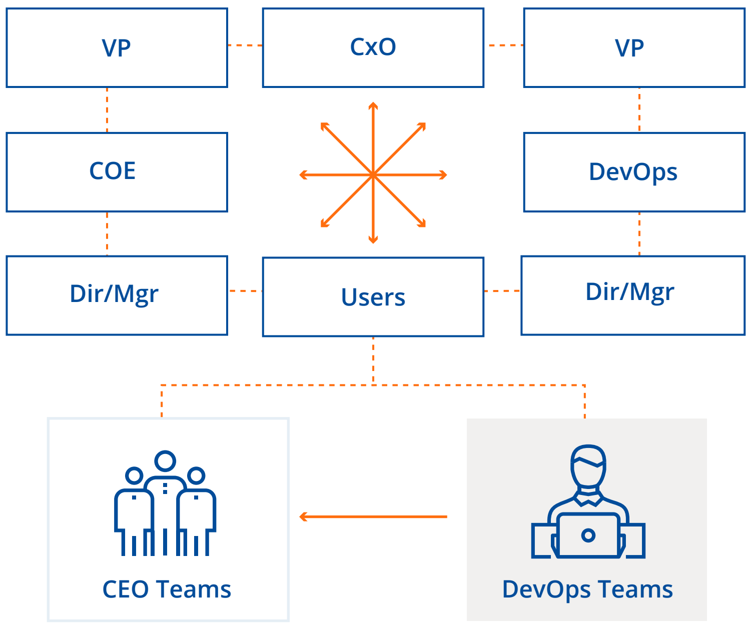
SILOED EXPERTISE DOESN'T SCALE

Performance testing enterprise applications has always required expertise. Applications today manage transactions that touch dozens of other applications and systems — and it's up to testers and developers to ensure that every release performs.

The modern application environment has become more complex — hyper-dynamic microservices-based architectures, multi-cloud deployments, automation, and integration — that only a handful of experts within an organization can reliably measure how full systems perform.

But there are simply not enough experts to tackle all of an enterprise's performance engineering requirements. Certainly not at the pace of today's release cycles. As a result, enterprises are finding that testing is becoming the #1 impediment to faster releases. Quality assurance is becoming a bottleneck.

Centralized and autonomous teams exist side by side



So, as enterprises adopt DevOps, how do they scale performant applications released daily or weekly, with a few scarce resources?

Since a few performance experts don't scale, organizations can instead develop a performance approach that scales.

Many enterprises have successfully adopted a scalable approach to performance engineering. What they all have in common is a standardized approach that meets the requirements for different types of teams, different types of applications, and different deployment models. Specifically, such a standardized approach:

- **Is easily adopted by performance engineering experts and non-experts alike**
- **Leverages a high degree of CI automation**
- **Supports a cloud-native approach to performance engineering**
- **Can be applied consistently to enterprise-grade applications like SAP and Oracle, Citrix-virtualized apps, and microservices-based architectures**

"A few scarce performance experts don't scale, so organizations need a scalable approach to performance engineering."

These enterprises have found that a standardized approach not only empowers autonomous DevOps teams but increases the productivity of all teams and makes the organization more resilient when key performance experts leave.

Implementing a performance engineering approach that the entire organization can standardize on is the best way to realize consistent quality across all releases all the time at the pace of today's development.

SEVEN WAYS TO IMPLEMENT A STANDARDIZED PERFORMANCE ENGINEERING APPROACH

So, what does a standardized approach to performance engineering look like? Where does an enterprise start? What should it consider? Here are seven considerations to a successful implementation.

1. Promote deep collaboration

Enterprise-wide performance engineering is most effective and efficient when it's a team sport. An approach that makes it easy for various teams to collaborate enables performance expertise to scale without adding more experts. This collaboration manifests itself in two ways:

- **Efficiency:** Having developers, performance engineers, business analysts, and others all working "on the same page" makes it easy to design tests with agreed-upon service level objectives (SLOs) that define measurable performance metrics — and ensures that everyone is measuring performance consistently and getting apples-to-apples results. This is much more unlikely when lots of different teams are all using lots of different tools. With consistent reporting, root cause analysis and trend reporting are easier across the board.

- **Effectiveness:** Performance engineering experts take on more of an enabler role. Instead of assuming responsibility for all testing operations themselves, they build out the building blocks that allow non-expert autonomous teams to test at the pace of development. They can structure the test environment, implement quality-control guardrails, set up automated CI pipelines, and embed best practices into performance engineering processes that empower decentralized teams.

2. Make things easy for non-experts

For different teams to use the same performance testing approach for their own specific needs, testing must be easy. Ease of use is what enables widespread adoption among teams who are not performance experts. Testing tools should have a short learning curve and not require specialized expertise. Look for tools that avoid the need to have deep coding skills — low-code and no-code approaches that leverage intuitive drag-and-drop, point-and-click functionality are best.

Testing should be flexible enough to adapt to the way testers (whether autonomous or centralized) work, not the other way around. Specifically, in addition to performing testing through a codeless GUI, the platform should enable DevOps teams to design and run tests <as:code> within the command line interface (CLI) or their day-to-day IDE.

3. Test fast to release fast

How quickly tests can be run is directly related to how easy the testing tool is to use. Tests take longer with tools that are tricky to learn. What distinguishes a fast testing tool from a slow one is the test script design/maintenance, test resource reservation, and test results analysis/reporting — actually pulling the trigger on tests is pretty much the same for all tools. So, capabilities that impact faster tests include how much manual effort and specialized know-how are involved in designing test scripts, whether scripts have to be rewritten from scratch every time code changes, how easy it is to reuse functional tests as performance tests, natively integrating performance tests into automated CI/CD pipelines, etc.

What separates a fast performance testing tool from a slow one?

Fast	Slow
Rapid test design	Difficult test design
Low-code and no-code approaches	All code approach
Automated script updates/maintenance	Manual script updates/maintenance
One-click conversion to reuse functional tests for performance	Manual conversion of functional tests for performance
Automates performance tests as code in CI pipelines	Does not natively integrate with automated CI pipelines
No expertise required	Deep expertise required

Bear in mind that testing faster not only benefits DevOps but also up-levels the productivity of the entire organization. Centralized teams can get more done in less time, freeing them up for more "expert-level" work such as new strategic initiatives, deeper analysis, DevOps enablement governance, and more.

It follows that the easier the tools are to use, the faster an enterprise can scale a consistent performance engineering approach across the entire organization. Everybody should be able to get up to speed on new tools in just a couple of days, with an enterprise-wide deployment in weeks.

4. Integrate the right tools for the right reasons

Look for opportunities to integrate best-of-breed or best-of-suite solutions in the toolchain to be a force-multiplier and "up-level" one another.

Functional testing

Repurposing functional tests as performance tests is one-click simple.

Application performance monitoring

Performance test results are consolidated with APM data in a single pane of glass view (Shift Right).

Continuous integration

Automate continuous performance testing within CI pipelines via on-prem or cloud CI tools.

Version control

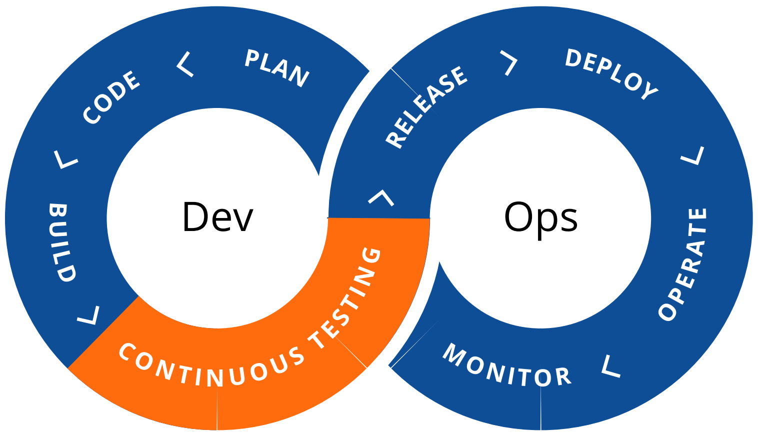
Leverage commonly used version control systems to kick off tests, and manage and share test assets.

Open API

NeoLoad open APIs connect to dev and business tools like Splunk, Tableau and Slack.

5. Automate performance testing in CI pipelines

Integrating automated performance tests into CI pipelines — continuous performance testing — is the holy grail of scaling performance engineering for autonomous DevOps teams. Given today's super-fast dev cycles, it's not just impractical but impossible for performance engineers to manually build, run, and analyze performance tests for hundreds of code pushes every day.



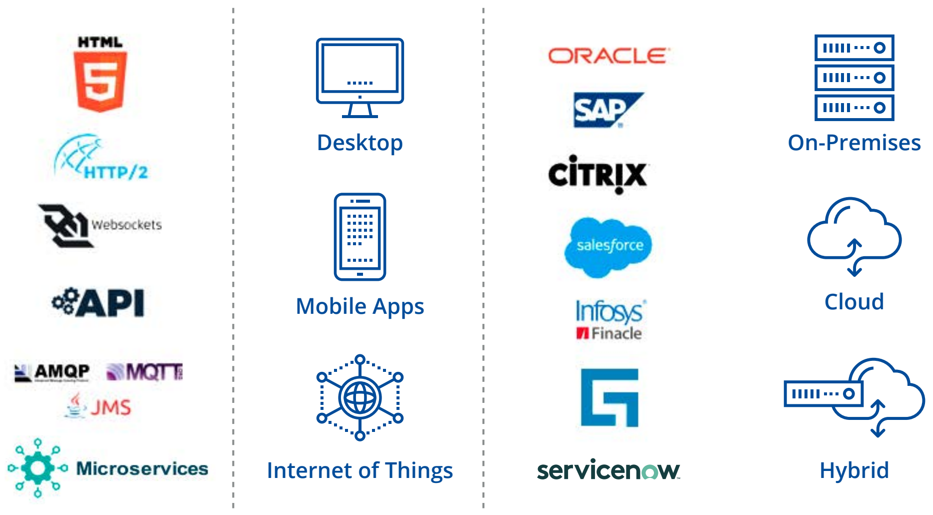
6. Settle on a solution that covers all the bases

To adopt a single, standardized performance engineering approach, an enterprise should first standardize on a performance testing solution that's designed from the get-go to support the full gamut of enterprise testing requirements.

- **Different teams and methodologies:** Enterprises today employ a mélange of methodologies that are carried out by centralized teams of experts (internal and external), autonomous development teams, or a combination of both. A standardized platform must work equally well for everybody.

- **Different types of applications and technologies:** The platform must also be able to test the full range of applications — from monolithic core systems and enterprise-grade packaged applications like SAP, Oracle, Citrix, Pega, Salesforce, Guidewire, Finacle, et al., to dynamic microservices-based applications. There should be a similarly wide technology coverage, from the latest frameworks to "older" technologies. Enterprises must be able to use the same solution to test the performance of all their apps end-to-end as well as test individual APIs at the individual component level. A standardized platform must work equally well for everything.

A standardized approach works for every team, every app, every technology



- **Different environments:** The platform should not tether the enterprise to a single deployment option. Virtually every organization's environment is some combination of on-premises, private cloud, and public cloud. As enterprises increasingly transition their applications to the cloud (or from private to public cloud, or back again), they need a solution that can test performance for complex migrations — e.g., moving SAP to an S/4 HANA implementation.

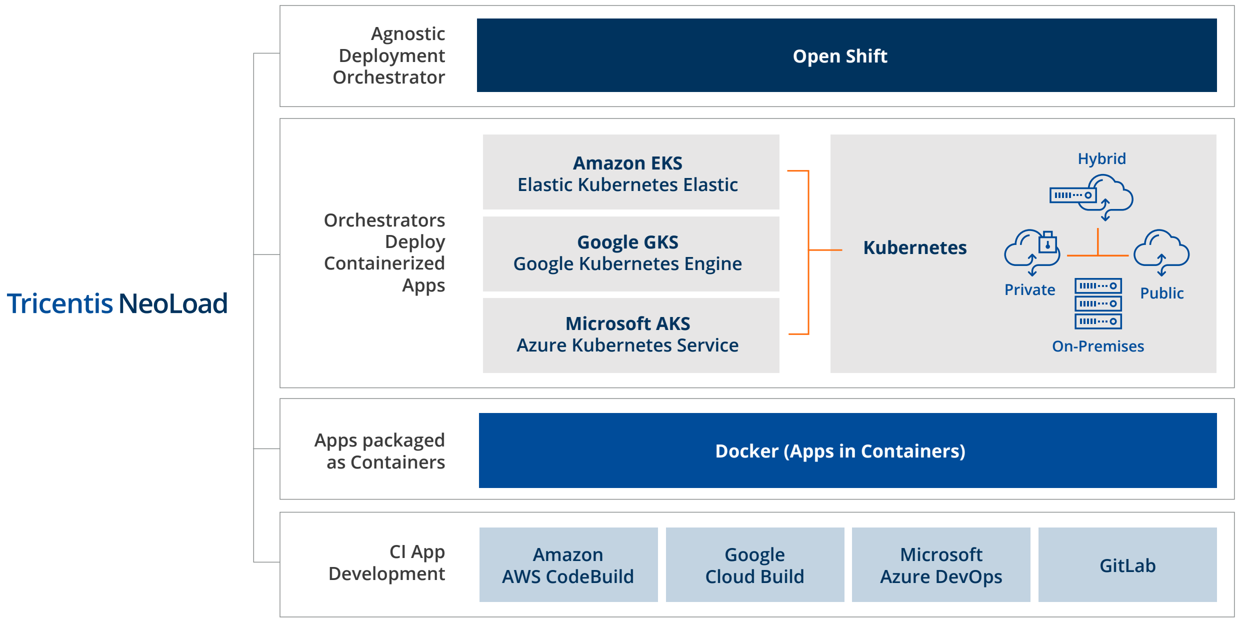
7. Think cloud-native

No matter where they are on their cloud journey, enterprises have to ensure that their approach to performance engineering is cloud-ready. Not only are applications moving to the cloud, but so are the software development lifecycle, process, and tools. An enterprise's approach to performance engineering should anticipate several complexities:

- **Different migration scenarios:** Whether the approach to SaaS is lift & shift, replatforming, or refactoring the architecture of packaged or on-premises apps, organizations need to be able to baseline performance, before and after, to ensure KPIs are met.

- **Multi-cloud strategy:** Performance engineering tools should be vendor-agnostic so that performance and scalability can be measured across different cloud providers (AWS, Google, Azure). If one cloud provider has a security breach, cost spike, or a service issue, organizations need to have their applications initially developed, and then measured, so that they can immediately shift from one operator to another without a change in user experience.

- **Cloud testing complexity:** Scalability isn't free. Enterprises should adopt an approach to ensure that scale doesn't camouflage non-performant code and make use of dynamic infrastructure to spin up (and down) testing resources as needed.
- **Cloud technology complexity:** The approach needs to work with every layer of the cloud (IaaS, PaaS, SaaS), across all layers of the cloud software development lifecycle: cloud CI tools like AWS CodeBuild, Google CloudBuild, Microsoft Azure DevOps, and cloud orchestrators like OpenShift, Kubernetes, EKS, GKE, and AKS. Performance testing needs to scale with the cloud-based software development model.



CONCLUSION

Enterprises expect and demand a high level of confidence in the quality of their software releases. The expertise to realize this confidence has traditionally rested with only a few specialists. But there are not enough experts to keep up with the pace of development as enterprises transition to faster, more frequent releases.

What's needed is an approach that scales performance engineering across the entire organization.

A successful approach standardizes performance engineering — especially performance testing — among different teams with different backgrounds and skill sets, for different kinds of applications. A standardized approach should be easy to use for both performance experts and non-experts alike, from CoEs and other centralized teams to autonomous DevOps teams. The same approach accommodates complex end-to-end testing of enterprise-grade applications like SAP, Oracle, Salesforce, and Citrix-virtualized apps as well as API testing in microservices-based architectures.

Having both performance engineers and DevOps teams standardize on a performance engineering approach that works equally well for both gives enterprises the predictability, validation, and assurance that they're used to but at the volume and velocity of automating in an Agile environment — quality at scale.

ABOUT TRICENTIS

Tricentis is the global leader in enterprise continuous testing, widely credited for reinventing software testing and delivery for DevOps and agile environments. The Tricentis AI-based, continuous testing platform provides automated testing and real-time business risk insight across your DevOps pipeline. This enables enterprises to accelerate their digital transformation by dramatically increasing software release speed, reducing costs, and improving software quality. Tricentis has been widely recognized as the leader by all major industry analysts, including being named the leader in Gartner's Magic Quadrant five years in a row. Tricentis has more than 1,800 customers, including the largest brands in the world, such as Accenture, Coca-Cola, Nationwide Insurance, Allianz, Telstra, Dolby, RBS, and Zappos.

To learn more, visit www.tricentis.com or follow us on LinkedIn, Twitter, and Facebook.

AMERICAS
2570 W El Camino Real,
Suite 540
Mountain View, CA 94040
Unites States of America
office@tricentis.com
+1-650-383-8329

EMEA
Leonard-Bernstein-Straße 10
1220 Vienna
Austria
office@tricentis.com
+43 1 263 24 09 – 0

APAC
2-12 Foveaux Street
Surry Hills NSW 2010,
Australia
frontdesk.apac@tricentis.com
+61 2 8458 0766