



The State of

Performance Engineering 2020

A Sogeti and Neotys report

The state of performance engineering 2020 – contents

Introduction to performance engineering	3
Chapter 1: Culture	6
Enterprise spotlight: IMA	15
Chapter 2: Organisation	18
Chapter 3: Practices	27
Enterprise spotlight: MAIF	42
Chapter 4: Building in performance engineering	44
Chapter 5: Outlook	57
Closing thoughts: the practitioners' corner	68
Acknowledgements, About the study, Respondents, Footnotes and references, About Neotys, About Sogeti	74



Introduction

In the first half of 2020, application performance became highly visible and imperative. At the time of writing this report, video conferencing, teleconsultation, online shopping, and many online transactions were all in the midst of extraordinary surges of use. Companies guided by a strong proactive vision in digital business have managed to navigate successfully in these uncertain external circumstances. For instance, UK home furnishing retailer [Dunelm reported a 100%+ increase in sales](#) during the lockdown weeks of Spring 2020.

Organisations are changing the way they operate — with a surge in remote work — and they increasingly serve their customers through digital engagement. More resilient and faster digital services are a major success factor for enterprises in the new normal. This is even more prevalent for some companies with digital platforms acting as the only channel of customer interaction. Almost all applications are likely to be deeply embedded within organisation core systems and as such, they profoundly impact the overall ecosystem — positively or negatively. What matters is the ability of an organisation to anticipate, learn and innovate to protect their brand, reputation and revenue.

Welcome to the 2020 State of Performance Engineering report

The discipline of performance engineering is often reserved for specialists. There is little data available to enterprises to help them understand the current practices and how other organisations are managing it.

[This research fills the void by combining the opinions of 515 senior decision makers and the perspective of subject matter experts from Sogeti, Neotys and outside practitioners to explore the landscape of performance engineering.](#) This report reveals the place of application performance in an organisation's business, the organisational structure around it, and the various activities and trends shaping the practice. Derived from our collective pool of experience from helping companies in a wide variety of industries, we aim to provide pragmatic recommendations for organisations willing to improve and enhance the business value their performance engineering delivers.

This report is designed to benefit all stakeholders within an organisation that deal with application performance including: executives, business and product owners, architects, developers, quality assurance engineers, tool coordinators, infrastructure engineers, and system administrators.

What is performance engineering all about?

Performance engineering is a collaborative discipline focused on achieving high levels of application performance to benefit the business. Performance engineering contributes to making IT an enabler

and disrupter of change. It involves multidisciplinary professionals with complementary backgrounds and skills working together to achieve a common mission. Application performance is the overarching principle for which we bear equal responsibility and accountability.

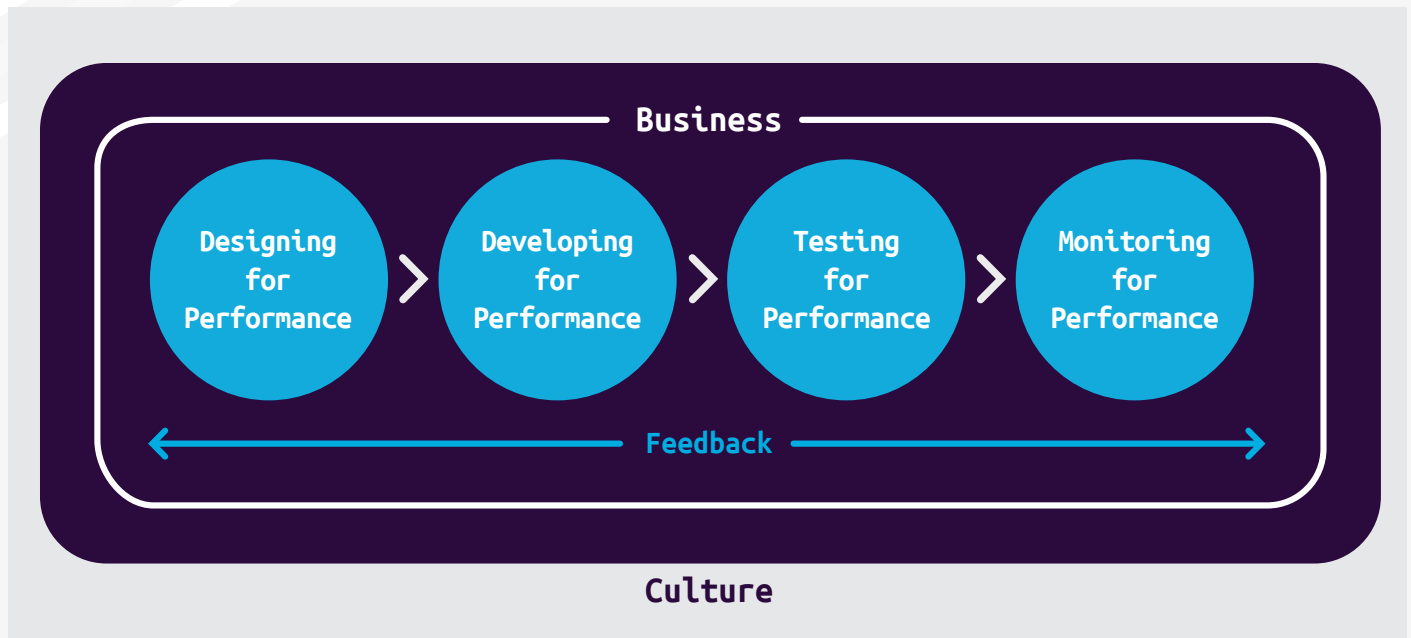


Figure 1: Performance Engineering

As such, performance engineering is much more than just running test scripts and analysing production metrics. If performance testing entails certain processes and steps to determine faults, performance engineering assesses the entire system to identify where and how different pieces can be optimised and:

- Ensures business continuity under changing usage patterns
- Introduces new services faster and improves the quality of experience for end users
- Controls costs and more specifically, infrastructure expenditure.

Chapter 1 is for executive leadership responsible for setting the direction and culture of the company. They need compelling reasons to drive transformative change. We examine how application performance is valued and what its impact is on the business.

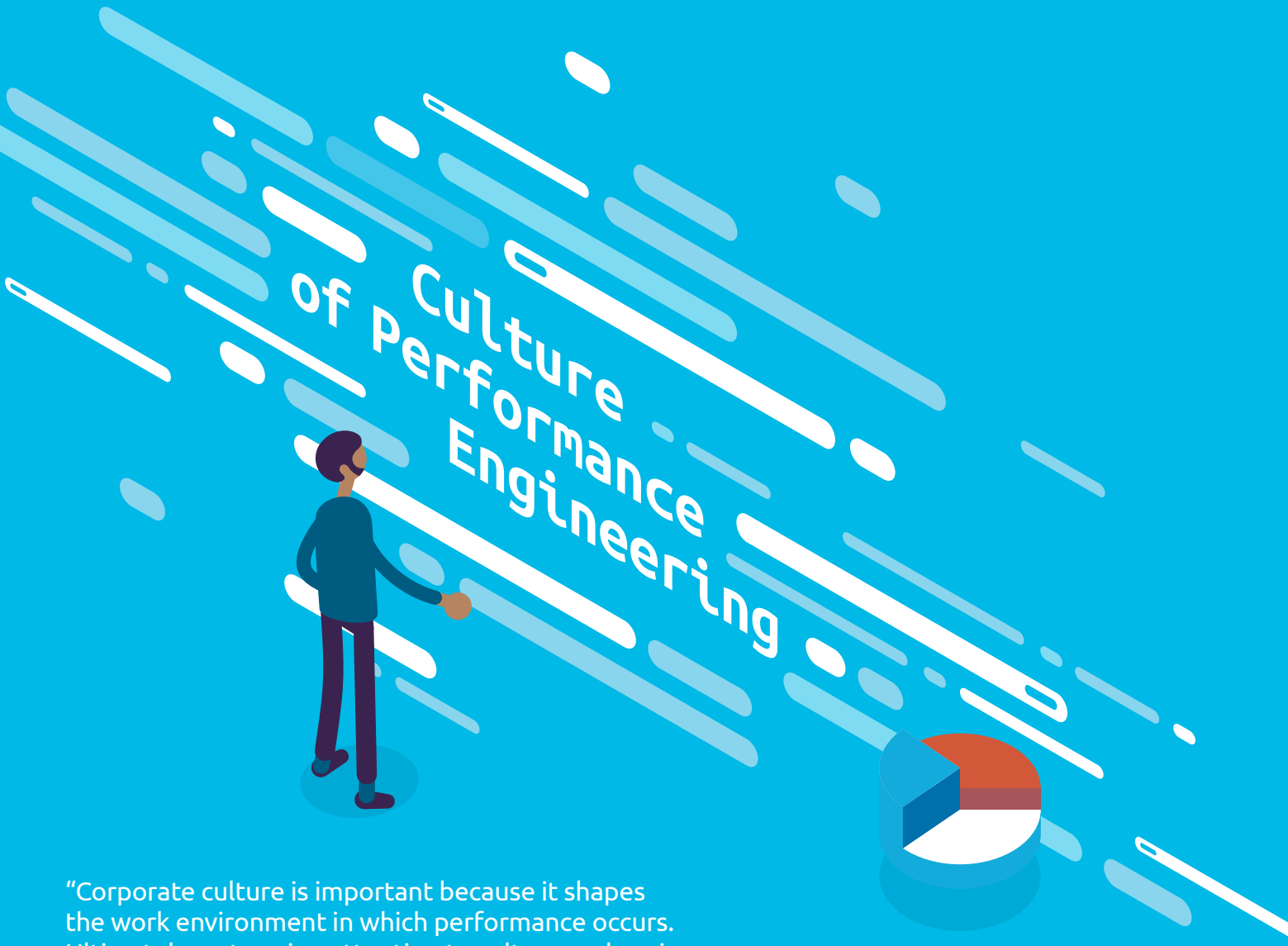
Chapter 2 lifts the veil on some of the key organisational components in building a performance-first culture.

Chapter 3 covers some of the building blocks and fundamental capabilities of performance engineering. Through a variety of use cases, we review why they play a significant role in achieving the mission.


Chapter 4 dives deeper into the underlying technical foundation and necessary integration as we start to incorporate performance engineering in the lifecycle.

Chapter 5 investigates the likely evolution of performance engineering.

We wish you an excellent read.



Culture
of Performance
Engineering



“Corporate culture is important because it shapes the work environment in which performance occurs. Ultimately, not paying attention to culture undermines an organisation’s profitability and sustainability”

— Torben Rick¹

Chapter 1: Culture of performance engineering

As leading business change guru Torben Rick [observes](#), the corporate culture shapes many aspects of a business including the way application performance is regarded, managed and implemented. It can also affect how people approach performance engineering and the effectiveness of the work they do. At the same time, performance engineering can affect the success of the business and the culture and strategy in terms of the way both the people and the business work.

In this chapter, we examine how application performance is valued and how this translates at the various levels of the organisational levels. In turn, we review the business impact of your culture of performance and provide recommendations on how to perfect your approach to performance engineering.

Where does application performance fit in business priorities?

A company publicly announcing a software glitch can lose much more than shareholder value. In 2018, the BBC reported that [10% of users would leave a website for every additional second a page takes to load](#). We should all be aware that poor application performance hinders employee productivity, degrades customer experience, reduces process efficiency, diverts resources and impedes many aspects of overall business performance.

With so many recent failures of software putting business and careers in jeopardy, the results of this survey are deceiving. They indicate that while many companies rely

on the performance of their applications, few are willing to invest in improving and supporting it. Staggeringly, only 21% of respondents claim application performance is an integral part of the brand equity. The disconnection may stem from operational issues — for example time pressures and budget constraints, and strategic issues — namely the lack of C-level understanding that causes repercussion beyond the financial impact. The real cost of poor performance is a combination of money, time and reputational damage.

The importance of good performance engineering has been known for many years. This is evident in examples like the 2006 interview with an ex-Amazon employee who explained that during various tests Amazon discovered even [small sub-second delays in the website would result in substantial and costly drops in revenue](#). More recently, in 2018 [Google research](#) claimed that “as page load time goes from one second to 10 seconds, the probability of a mobile site visitor bouncing increases by 123%”. End users have limited empathy for the effort in making a service perform. They just want things to work.

In 2020, the key expectation is fast, reliable and trustworthy software.

In 2020, faith in application performance means faith in the business as a whole.

Application performance is key, yet addressed tactically rather than strategically

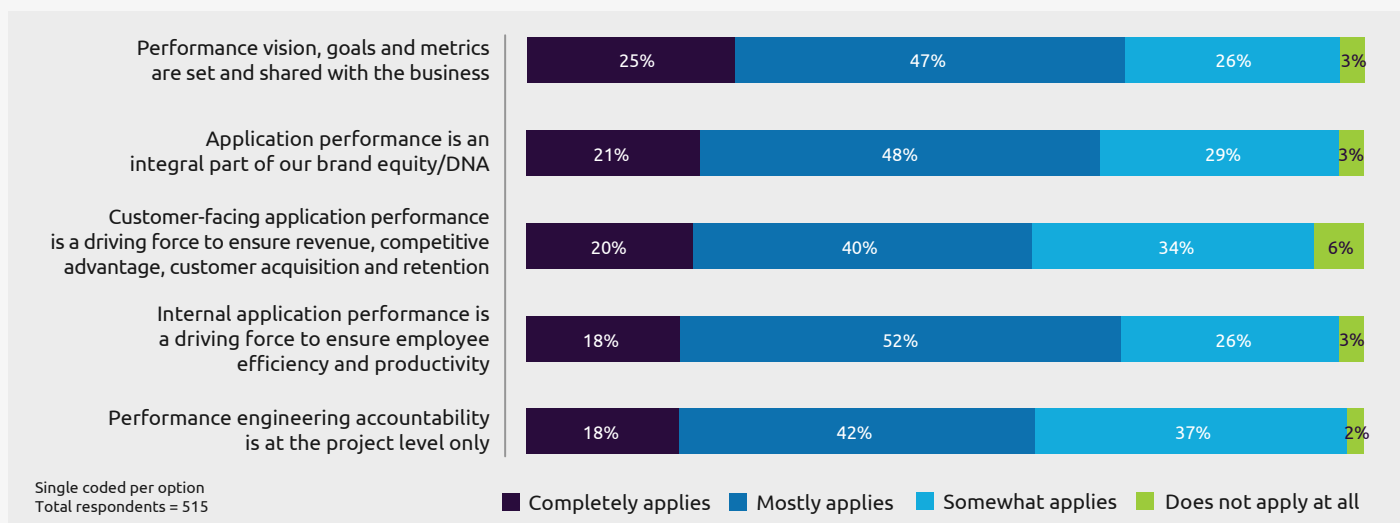
Most organisations restrict the measurement of application performance to traditional load and stress testing. One way to remove the restriction is to connect business outcomes to the performance of the computer

systems. Inherently, business stakeholders need to actively collaborate to identify the key business metrics in enough detail to be measurable, practical and useful.

A challenge for technical staff in these organisations is in getting the attention of the business stakeholders. Often, a more practical approach is for the business to lead the initial discussion and the Chief Technical Officer (CTO) to drive subsequent discussions and the necessary changes throughout the organisation. This gives them the authority, scope, and influence to be the voice of performance engineering. Therefore, those

who represent performance engineering are likely to improve it most by ensuring both the business and the CTO understand how performance engineering can help them achieve their objectives, and how they can help to maximise the effectiveness of the performance engineering to make this happen.

25% of the respondents connect system performance directly to their business outcomes. This connection provides a common bond and helps companies determine whether their approach to application performance fits their specific business requirements and



Question 1: How well does each of the following statements apply to your company?

business logic, and adapt accordingly.

At the vanguard of the industry, 20% of respondents believe performance is a driving force to making their business successful. These organisations often measure performance throughout the software development lifecycle, including production. For example, there are businesses who consider and measure performance from an architectural standpoint, at a UI level, an API level and at a data level.

A healthy indicator of the relevance of performance is when people across the company truly consider the topic in their various discussions. They may do so at an Agile 'story' level, for instance, to help the software development team incorporate appropriate performance measures in their work. A corroborating sign is when performance is visibly measured and the results freely available throughout the business. There are several examples later in this report of companies who have made performance pervasive.

Incidents remain the greatest driver of change

We have found two main catalysts that drive behaviours in how organisations approach performance engineering. The first is where the organisation has experienced significant and adverse performance issues. The second is through the influence of people who infuse performance engineering into a company's culture. These people often gained their expertise elsewhere before joining this organisation and are able to share authentic and motivating examples that inspire their colleagues to enlarge their thinking about ways to improve their practices.

“The difference between average people and achieving people is their perception of, and response to, failure”

— John C. Maxwell

A company with an online e-commerce website discovered an estimated 20% to 25% of their business was lost due to poor performance of various application services. They discovered these performance issues, together with several other issues, contributed to customers being dissatisfied and abandoning their shopping.

These issues included poor performance of the shopping cart and of the payment gateway. They also found further delays in pricing updates and various issues related to mobile devices.

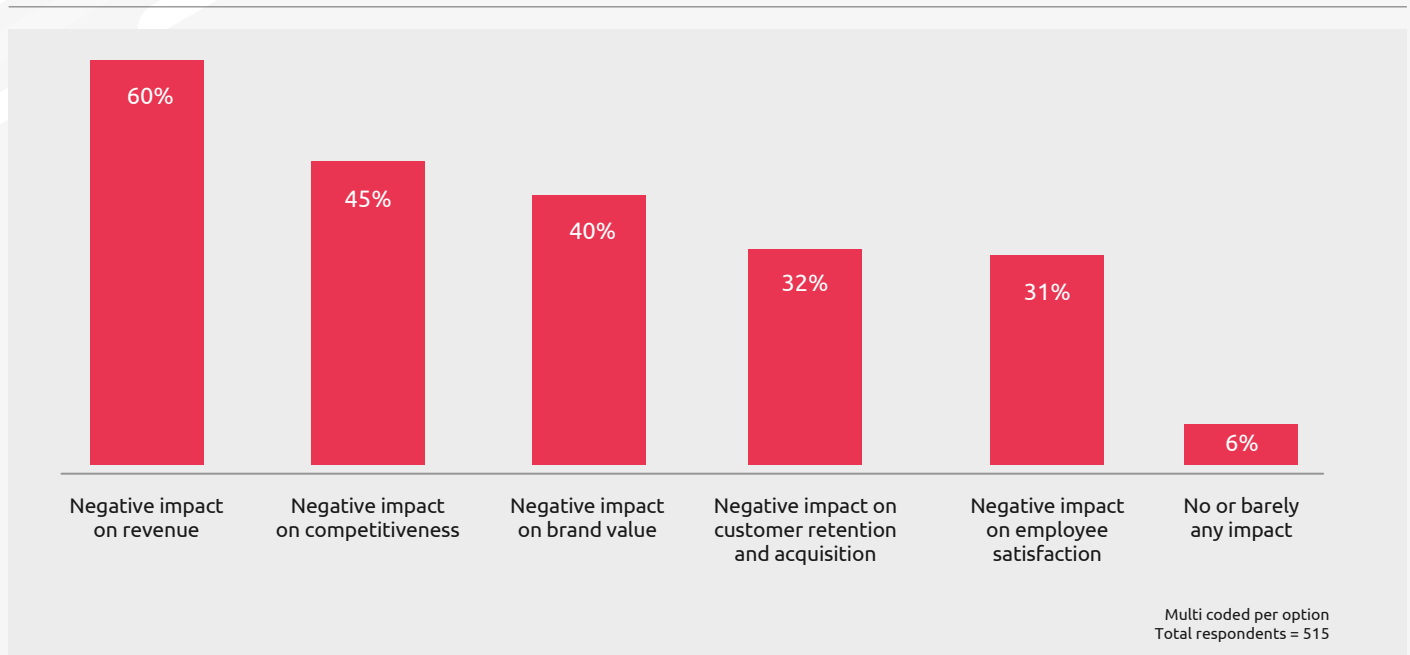
After investing in a new initiative to increase the quality of service, in which they established performance engineering as part of their culture, they increased their revenue by 15%.

The company used several Enterprise Performance Management (EPM) solutions to measure the business implications of what was happening in production. Through these software tools, they learned more about their customers' behaviours so they could improve their systems to address shortcomings in their production systems.

A good example of the first catalyst is through the impact of COVID-19 on customers of a regional bank in the USA. Before COVID-19, they had not given much consideration to the performance of their systems. However, they realised the importance of performance as they sought to expand their COVID hardship website in order to handle an increase in activity and provide a good user experience to

their customers who had already suffered the impacts of COVID-19 on their lives and finances.

Furthermore, good performance may be part of the culture for additional reasons, beyond direct business metrics. For example, better systems performance may also improve staff morale by enabling them to work more efficiently and effectively. After all, few of us enjoy our



Question 2: Which of the following negative business impacts are likely to occur when application performance is poor?

work when the systems perform poorly. Companies that invest in improving systems performance show they care about their employees.

Some businesses now include performance requirements in their contracts with their suppliers. These requirements often have penalties associated with them where the supplier earns less if they do not meet the performance requirements. As an example, a large telecom company agreed to meet or exceed a number of smartphone sales for a leading device manufacturer within a specific period from the launch date of a new device. Internally, they needed to set even tighter internal performance metrics to provide a safety margin if performance issues occurred during the critical sales period.

Application performance incidents are often an outcome of organisational failures that could and should have been addressed. One of the challenges for teams is how to foster accountability and ownership for the many and various SLAs, SLOs, KPIs, targets and penalties, especially if they are responsible for the development of an individual microservice. One of the key concerns is whether they have identified relevant and useful metrics. Another challenge is working out how they can map business metrics to identify suitable performance metrics for their code. Often doing so is an iterative process that benefits from relevant experience.

Green IT remains a peripheral topic but is becoming more prevalent in conversations, although few companies have solid data to determine its overall impact. During the survey interviews, we only met the topic in a few distinct areas, such as in the design and performance of data centres. There are [certain point solutions](#) that inject

resource parameters into a test benchmark. Salesforce are one of the vanguards in this area where they measure the impact of their business holistically in terms of Green IT and even provide software tools to enable their customers to do likewise. As of now, there is no official technical standard or label to validate the environmental impact.

See www.salesforce.com/eu/blog/2020/03/introducing-sustainability-cloud.html and www.salesforce.com/company/news-press/stories/2019/09/091819-wk/

The need to break comfortable routines

One of the largest Norwegian business schools experienced an outage of their online examination system. The issue was covered in the national newspapers generating bad press for the organisation. This provided the impetus to start a strong performance engineering practice.

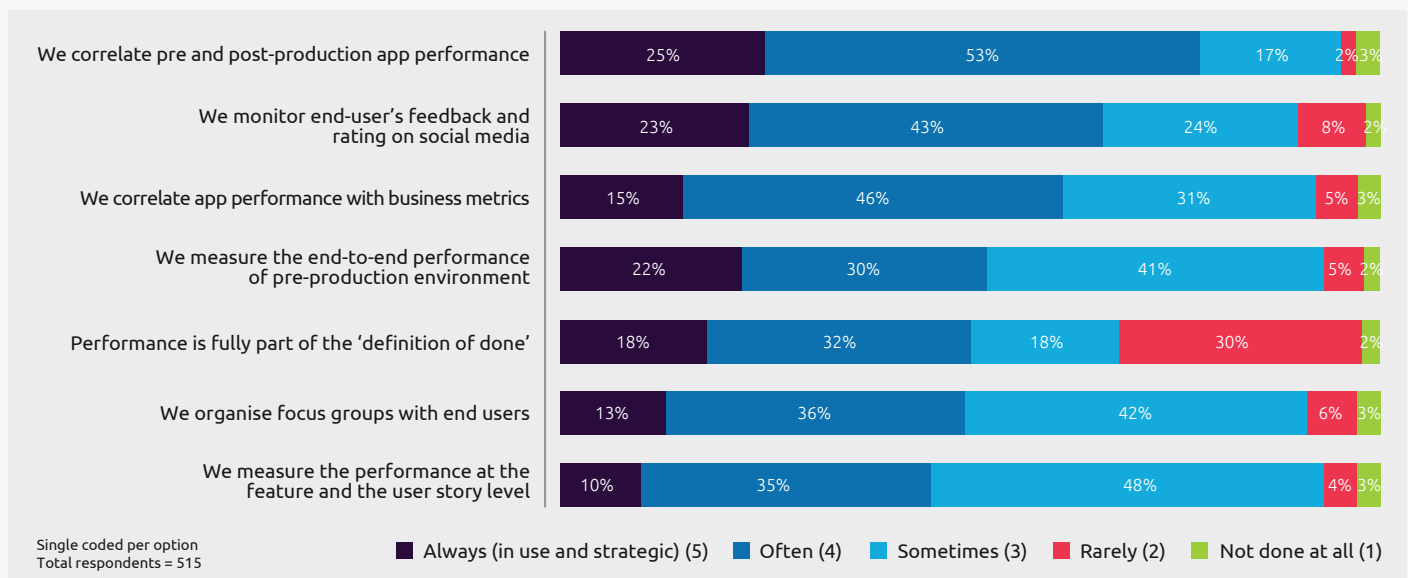
A multinational retail group observed a direct impact of poor application response time on the number of users and size of the average shopping basket. This was due to lack of end-to-end testing of individual software components, delivered separately. Although the measurement was not scientifically rigorous, the correlated degradation of financial results was obvious enough to generate a profound change in corporate behaviour towards application performance.

Let's defragment our measurements

The culture of performance engineering also means that technology, processes and activities should be measured. Application performance can be tracked in a number of ways, such as using customer satisfaction (human), business oriented (financial), analytical (journeys and tunnels), as well as technical (transactions per second). Another complementary approach is to see whether performance is part of the Definition of Done (DoD), whether it is included and used to decide whether a piece of work has been completed adequately.

Most mature organisations ensure they assess how their software performs from a business standpoint in addition to technology-facing metrics. However, currently only a minority of organisations truly do so at the moment. Teams may be working independently, for instance one team working on the user interface, another on the data, another on the infrastructure layers, and so on. Holistic performance may be absent even if the individual teams measure the performance of their software.

Evidence of an organisation that understands performance engineering is one that provides an API



Question 3: How do you track and measure the various aspects of application performance?

platform for a large insurance company. Their teams have learned to consistently ask questions about performance for each piece of development and to decide whether to set explicit performance requirements for that work.

Key recommendation:

To maximise the benefits of performance, align and bind technical and business metrics.

The Question 3 response of 13% incorporating end user focus groups is surprisingly high in our experience unless it includes apps developed for internal users, such as employees. Organisations that include feedback from users generally use surveys and monitor social media rather than focus groups of external users, such as customers. Another weaker approach is where organisations don't have direct measures of user-

centric performance, instead their measure is driven by perceptions or the number of tickets filed against performance.

Disturbingly weak expectations for response times

We believe it has become a truism that the responsiveness of applications impacts users' behaviour, whichever way that behaviour is measured, be it: session depth², return rate, or productivity. And yet, the survey results indicate that over 80% of the organisations surveyed are willing to accept response times of more than 3 seconds for their mobile apps. Paradoxically, the same individuals who consider 4, 5, and even 10 second response time satisfactory at work are likely to expect shorter response times as end users.

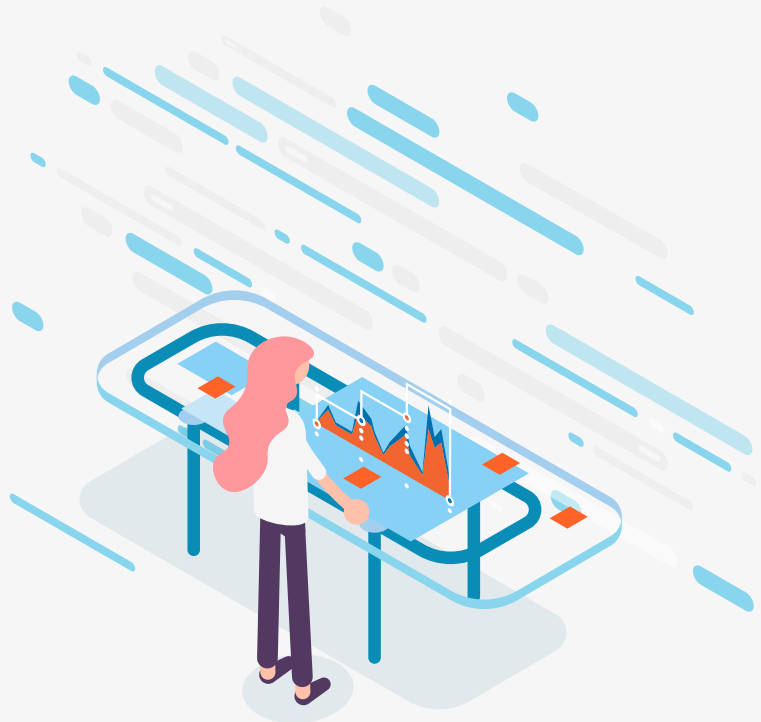
	No standard defined	10 + seconds	6 to 10 seconds	4 to 5 seconds	3 seconds	2 seconds	1 second
Web Application	0%	1%	9%	28%	41%	21%	1%
Mobile (App,Web)	0%	6%	32%	44%	11%	6%	0%
CRM (Salesforce, MS Dynamics)	0%	20%	35%	22%	15%	6%	1%
Packaged Applications (Oracle, SAP)	0%	12%	27%	24%	20%	16%	0%

Question 4: For each application type, what response time (in seconds) does your organisation perceive as quality?

Do we treat our users as second-class citizens while we expect a first-class service as users ourselves?

There may be even more at stake here than this notable difference in expectations. The majority of the people we surveyed have been in work for at least a decade and have reached senior positions. For these companies to remain relevant, they need to attract younger generations to join the business. However, many young adults seek and select jobs based on additional criteria, a new dimension — purpose³. Salesforce is an example of a company with a clear purpose. Their 1% pledge⁴ and their approach appears to be paying off in terms motivating people to join the company and ecosystem. In the ‘[Utopia for executives](#)’ research report, Sogeti defines purpose directionality as an outside-in perspective. What ‘purpose’ does your business serve in light of the direction that society is taking? Young people don’t want to join slow companies.

Implementing a culture of application performance starts at executive levels with a company-wide charter of quality and much tighter alignment between the various teams. Application performance should be everyone’s responsibility. It is time we prepared our respective organisations for better application performance governance, either via a reactive defensive approach or an opportunistic one. This might well result in organisational changes, which we will cover in chapter 2.





Enterprise spotlight: IMA

“Performance engineering is at the core of our business”

From emergency assistance to long-term support, IMA offers a range of solutions including motor, transport, home, medical, individual, international mobility, advisory and customer relations services. The IMA Group is mandated by its members and customers to implement guarantees or services that their beneficiaries may claim under their contract of insurance (assistance), services or asset management. Within this context, the company serves thousands of customers every day with a constant focus on the quality of service.

Vivien Delahaie, IT Environments Manager at IMA, explains the importance of application performance as part of the company culture.

“With 6,000 new customer cases created every day, fast application response time is a top priority requirement as it directly impacts our bottom line”

IMA support call agents create these cases every day to address customers' requests for assistance, which are managed by a central system. Each case generates an

average of 600 transactions, so their systems process around 3.6 million transactions per day.

The breakdown of customer-management system costs typically includes planning, development, implementation, training, and support. IMA considers the impact of poor application performance as a hidden cost and one worth managing. The longer a case takes, the more costly operations get, and the worse the end user experience is. IMA support agents are trained to adapt their working practices so they can provide phone customers with a good experience even if the performance of the system is poor.

The company discovered that improved application performance provided two complementary benefits. Firstly, call agents were more satisfied. They used the applications more and were able to learn faster because the systems were responsive. Most importantly, they enjoyed their work. Secondly, the quality of services delivered to the end customers was also improved, a key metric for the business stakeholders.

With this critical objective in mind, IMA has been investing in performance engineering for the last 2 years. Front office application reliability and responsiveness have been made critical in order to streamline the entire case management process. IT and business teams collaborate to align the actual application response time with the expected 'business response time', which is acceptable time for a call agent to process a case. As a fundamental aspect of IMA's efficiency, this shared vision of performance is reported at the highest management level.

“Achieving user experience goals is a continuous journey and application performance is evidence of this continuous effort”

The importance of application performance has nurtured a culture of performance engineering within the IMA IT organisation. This change in culture drives how applications are being designed, developed, and deployed.

Application performance is tested before production and monitored in production to ensure constant reliability and velocity. Each week, the application performance metrics are reviewed with both the IT and business committees to encourage shared accountability on objectives and measurements. The IT team fine-tunes the various components of a complex technology stack including a mix of web, HTTP, microservices, and proprietary technologies, such as SAP, in order to continuously improve the performance of the system. IMA measures their CRM performance via both technical (e.g. defect density and application response time) and business metrics (e.g. abandonment rates, engagement rates, and customer satisfaction).

“A new strategic approach made it much easier for business and IT to align expectations with the technical environment and constraints”

The entire workflow for a customer case process is so complex that it takes a whole open space wall to properly design. Testing this process efficiently requires a deep understanding of each phase and how they are

implemented. A change in approach was an absolute necessity.

Within the last 2 years, application performance has become a strategic initiative with the following changes:

1. A new horizontal service management team, which is part of central IT, ensures business requirements are met. This spans from the capture and design of the performance requirements to the definition of service level agreements. This team has been extremely instrumental in fostering this culture of performance.
2. A third-party team of high-skilled experts is available on demand. This team can help with automating performance testing and managing demanding performance testing activities.
3. The promotion of direct communication between the various stakeholders. This helps expedite the entire test phase, including faster test scripting and root cause analysis.

This shared vision has now been applied to the entire Software Development Lifecycle, before, during, and after delivery in production. The scope of performance testing depends on the requirements of each project. To limit the costs, performance monitoring in production is reserved for large and critical applications.

This journey to performance engineering made it possible to greatly improve customers' and support agents'

experience. For instance, the team was able to decrease the end-to-end process time by up to 6 seconds through the removal of useless loops in the code.

“Our short-term future is all about automation, while medium-term is cloud”

“In terms of automation, we need to be able to move towards automation to maintain our quality of performance in terms of delivery capacity while managing the delivery cycle. If we don't automate sufficiently the company needs to test more, which increases the costs”

Performance engineering needs to support cloud infrastructure, either as a result of the migration of existing apps or the development of cloud-native ones. With less control on the technology stack, performance engineering activities might prove more complex.

IMA has embarked on a journey to democratise performance engineering for all new services and new features. This will result in empowering developers to code with a performance mindset and increased performance testing automation.

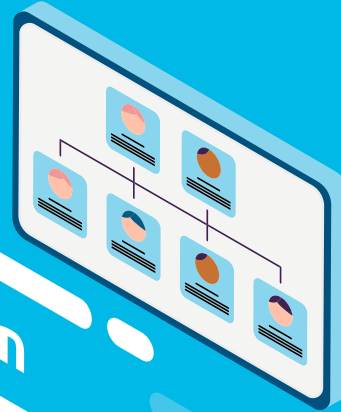
IMA continues to support the performance engineers as they provide new services and features within the business. By focusing on the performance engineering within the company, IMA continues to empower developers with a more focused mindset towards performance as well as increasing the use of performance test automation. All of this results in reducing both the

time needed to fix future problems and the amount that the testing costs.

IMA Group

www.imagroupe.eu | [LinkedIn](#) | [Twitter](#) | [Facebook](#)

Organisation of Performance Engineering



Every company has a distinct organisational culture that influences how employees work, as well as tying employees, suppliers, and customers together. The organisation's identity is connected to its structure, orientation, and dynamism. In turn, the identity of the organisation is likely to affect the adoption of performance engineering both as the overarching theme and underlying supporting activity.

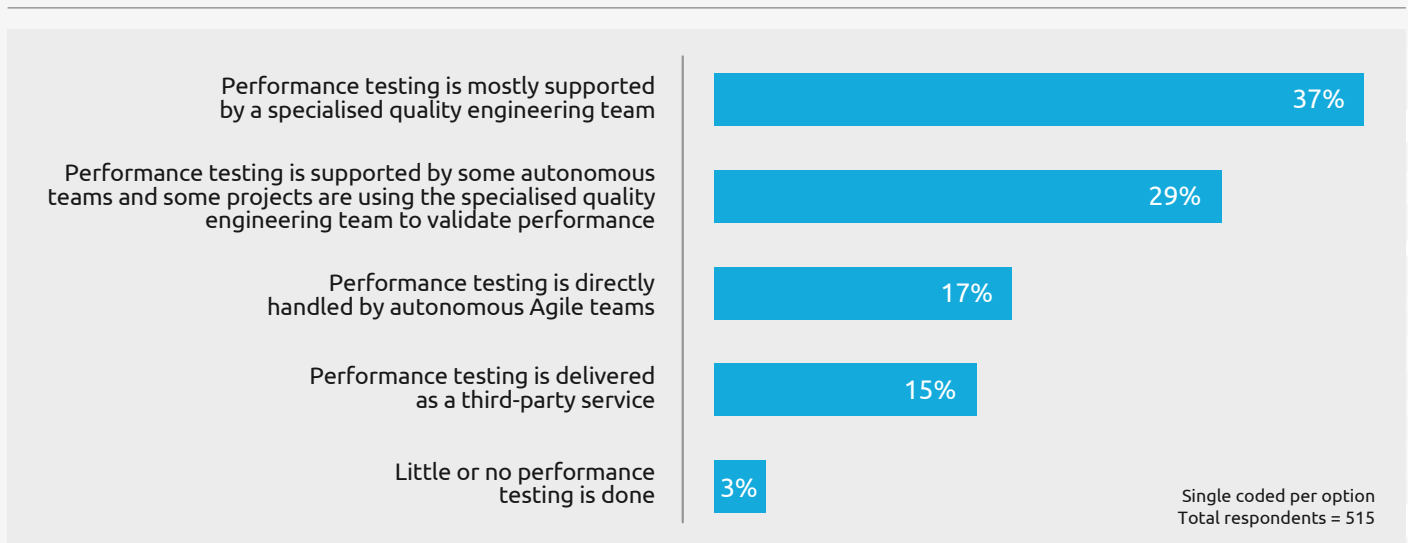
Chapter 2: Organisation of performance engineering

This chapter aims to help you understand the effects of your current organisational structure and working patterns. By understanding these, you may be able to adapt your organisation in order to improve the results of your performance engineering to meet your business objectives.

Modern trends in the organisation structure

In recent years, time to market has shortened: businesses and, by extension, apps are increasingly customer-driven. In order to achieve this, there is a need for greater velocity and for greater responsiveness.

The results of our survey show that businesses have a variety of approaches to organising their performance testing. 37% predominantly use a specialised quality engineering team, 29% have a collaborative approach, 17% of the development teams work autonomously and 15% apparently outsource the work entirely. We



Question 5: How would you best describe your current performance testing organisation?

are surprised so many companies seem to outsource their performance testing given the contradictory trend for the development teams to own their code and get involved in performance testing. One of the clear trends is that Agile teams are actively involved in performance testing in some capacity (46%). We expect this trend to increase.

Who undertakes the performance testing for companies seems to be in flux and likely to change in the next few years. We believe there are two tectonic movements affecting performance testing: the adoption of Agile development practices and the move to DevOps.

Agile enables development teams to work more autonomously and take more ownership and responsibility for their work. It also leads to more frequent iterations in terms of developing and releasing software by the team. These Agile teams therefore need low-latency and easily repeatable performance testing. The traditional mode of operation where performance testing was scheduled for a specialist performance testing team and happened once, just before production, isn't fit for purpose (has it ever been so?). A concrete example from one of the experts is where the 'performance testing experts' wanted to: "...do the performance tests, qualify the complexity of the test, commit and three weeks later, provide the results and move on to the next project". Agile sprints start and finish in less time!

The movement towards the teams doing some or all of the performance testing has a profound effect on the people who were, and may still be, part of a centralised body of experts who specialise in software performance.

What is clear from our interviews with experts is that there is an opportunity for competent specialist teams to evolve and become more relevant. Conversely, those who are weak, slow, or remain isolated will decrease in relevance and value now they are no longer the gatekeepers, and no longer mandatory. The role of the experts is evolving in parallel. We cover this topic in the next section.

"Performance testing is still mainly carried out by specialised teams" (37%)

This includes two types of team: a traditional 'centre of excellence' and one that is oriented to be DevOps-ready.

The best days of a single 'centre-of-excellence' group who were responsible for performance are in decline. The old role of gatekeeper lacks credibility. Instead, staff are expected to be self-directed and work as peers, gatekeepers are seldom to participate in these teams. Specialised quality engineering teams are not 'bad' unless their attitude to working with others is. Healthy teams are able to reduce mistakes made by those who are unfamiliar with performance engineering (yet need to do it as part of their role). This organisational model does not foster a strong corporate alignment on application performance.

Embedded into a DevOps initiative, a low-latency core team of specialist performance engineers can help multiple development teams concurrently. They can also help infuse good engineering practices throughout

multiple teams. These modern centralised bodies are positioned as end user advocates to further assess the end-to-end customer journey. They help the teams conducting tests that are outside their direct ownership (e.g. from point assessment to continuous integration). They accelerate the transformation of feature teams via concrete experience and know-how. They keep performance at the very heart of the definition of quality. As part of this, they renew practices, review tooling, and challenge methods to scale uniformity and reusability of assets across organisations, with sustainable innovations.

“Performance testing is directly handled by autonomous Agile teams” (17%)

Agile and DevOps demand drastic changes in performance testing activities, whether it be in the design phase, in making supportive processes available (data, service, environment), when the tests are executed, or its overall orchestration. This really implies a mindset at a strategic level, of methodology and processes, with a strong technological component. By analysing the results of our survey, we realise companies moving to Agile often overlook the importance of performance testing. Performance testing in Agile development seems to be nearly as neglected as it was in Waterfall software development where performance testing was done in the 11th hour of the validation process of a release candidate. This apparent neglect can also lead to unclear and untrustworthy architectural requirements.

The ‘Academy’, immersive initiative for a sustainable culture of quality

A national logistics company established an ‘Academy’ at the heart of their transformation program to Agile. This consists of a small core team who are responsible for developing new knowledge and skills necessary to adapt to the needs of the program and to changes in the ecosystem.

The Academy is as a long-term investment, with the purpose and role expected to evolve over time. It is intended to be a way to guarantee the integration of all stakeholders into the culture of this company. Application performance is intended to be a gold principle, one that must suffer ‘zero loss of consciousness’ over time, meaning no loss of knowledge as people leave and join the teams.

New collaborators are encouraged to fully participate and understand performance within the company (and new ways of doing things). This Academy encourages the propagation of knowledge throughout the company and across the various geographies the company operates in. This initiative is designed to remain adaptable and scalable to support future business needs, the rise of new technologies and the expectations of future generations of IT engineers, as well as their social and societal constraints and opportunities.

When left to non-specialised people, performance often becomes siloed and expensive.

A European government body decided to adopt a Docker architecture. Each feature team leader was tasked with measuring the performance of their deliverable. They did so independently and used their own measurement system and metrics. When their assessments were combined, an architect with overall vision realised the proposed system architecture was going to be ten times larger than what was actually needed. The oversized architecture would have been hugely expensive. By including a specialised performance architect they were able to integrate all the components and right-size the architecture for optimal performance and cost.

Performance engineering activities within feature teams should contribute to the Definition of Done (DoD). Where practical, they should be implemented in sprint test strategies, automated in test scenarios, and used to help monitor quality. The objective here is to blend performance into the application lifecycle as a core function. Doing so increases developers' awareness of performance. A fast feedback loop enables the developers to address performance concerns before they become a much bigger issue. Performance is treated as one of the aspects of quality.

In Agile software development practices, finding the right balance between autonomy and cross-team collaboration can be complicated. In some cases, the choice of technology may impose practical constraints, e.g. some software licenses lock performance tests specific users or computers rather than allowing them to be run wherever they are needed.

“29% are adopting a collaborative approach”

Lightweight apps including those designed using APIs and microservices may suit smaller and more frequent performance tests. If so, the collaborative approach is considered the most promising. A centralised body advises feature teams on the right strategy — be it tools or methodology, and participates in the automation effort. A possible pitfall is where this centralised body is restricted to acting as a provider of skills rather than an enabler of industrialisation and repeatability. A collaborative organisational model increases the likelihood of achieving substantial planning and modelling even before the first line of new code is written. A collaborative approach is also pivotal in closing gaps between multi-discipline teams.

In our experience, as companies adopt Agile development practices, there are far reaching and ongoing effects in terms of performance engineering, including performance testing. The effects include almost every aspect of the work: from who selects and pays for the tooling, to the skill sets of both the specialists and those who get involved in performance, for instance as

a developer in an Agile team. Unless companies and the people who end up getting involved in the actual work actively adapt, they risk squandering money and energy while also doing tests that are not relevant.

There may be a few isolated islands where 'traditional' performance testers and performance testing practices will remain for a few more years. However, for many, the changes are happening around them and to their roles whether they like it or not. Conversely, when the organisation is aligned, performance engineering is likely to be encouraged through helpful working relationships and matched needs and expectations.

The evolution of the role of performance tester

In living memory, the skills of a performance tester were predominantly in using performance testing tools and analysis of the results. They designed the tests, the ramp up time, traffic profiles, and ramp downs. They were often part of a specialised, dedicated team of performance testers. A key skill was the ability to identify bottlenecks then find and engage an expert, e.g. a DBA, a network specialist, to address the problem. Their team controlled the budget for the testing, including the often extremely expensive license needed to use the load testing tool.

There is now an emerging role of someone who performs effective and relevant performance testing in a DevOps environment. Currently there are two seemingly similar roles. Performance testers who understand how to create automated tests using performance testing tools,

Pervasive performance

A 500-employee online retailer has 4 production releases per day, with high-end expectations on performance. Live dashboards in open spaces continuously display production data, early signs of incidents, incidents, and alerts. These are visible to all the stakeholders regardless of their role or responsibility. They enable people to work cohesively as a team so they can respond quickly and effectively, for instance, to minimise degradation while waiting for a correction.

A UK pharmacy has installed live screens within their open space offices to increase collaborative transparency and accountability on each of the app's development pipelines, including the behaviour in production. All the teams can see the information regardless of who developed them.

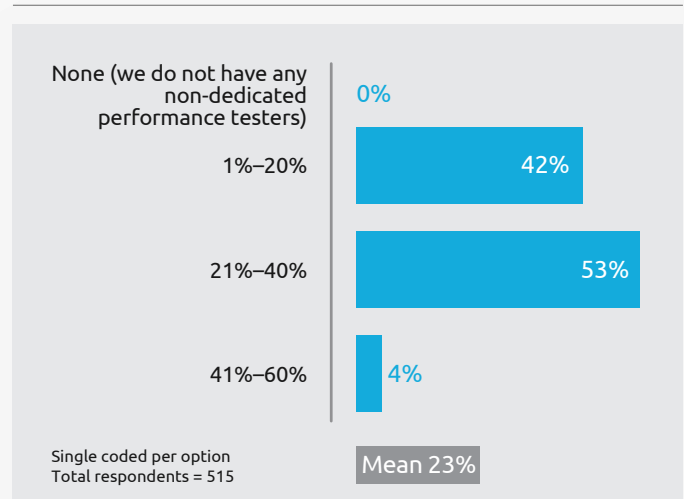
A leading provider of premium gaming solutions with a very strong DevOps culture has invested in live screens continuously displaying both technical (e.g. Jenkins automation) and production analytics (e.g. the number of people connected to online games). The technical teams have direct access to information about the performance of the business, enabling them to see how their work contributes to the success of the business.

and 'DevOps Automation Specialists' who help automate build pipelines, and so on.

DevOps automation specialists are often unfamiliar with performance testing tools — more relevantly they are unfamiliar with what is needed to design relevant performance tests. They are able to integrate them and use them, yet they might not recognise the importance of testing beyond the 'obvious'. In contrast, performance testers might understand how to design tests and use performance testing tools in a controlled, mature test environment, but not understand how to design and integrate performance tests to run in a pop-up test environment (that lasts for minutes rather than days) and into a continuous build pipeline. There is a shortage of experts who can define a suitable holistic automation strategy. Businesses need people who can help bridge the gaps in expertise. As the answers to the following question indicate, gone are the days when all the performance testing was done by specialists.

The measurements of success and progress are also evolving: from technical measures to business measures. That is not to say that technical measurements are unimportant, however, the focus is on business and revenue measures over technical ones (www.agilemanifesto.org/).

As we move from performance being undertaken purely by dedicated teams to the democratisation of performance engineering, the choices of tools is likely to change in tandem. Tools that take months or years to learn how to use (and where people invested lots of time and money to acquire tool-specific certificates) are likely to be



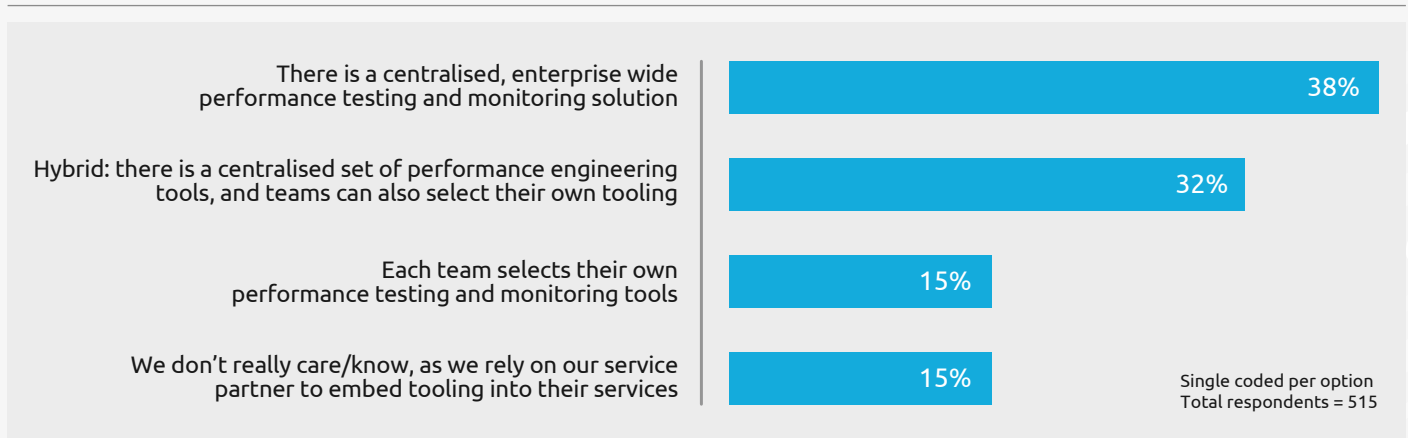
Question 6: What proportion of your performance testing is carried out by non-dedicated-performance testers?

replaced by tools intended to be low-code and low-touch, where the smarts are an intrinsic part of the tools.

This role will evolve alongside the Agile transformation. As an example, all activities that traditionally pertain to the design phase (creating, writing, and updating test cases) will be done reactively in response to changing requirements and frequent code changes. Organisations on their way to their Agile (of whatever form) will need rigorous and systematic performance testing that fits into the same cycle, the same sprint. They also need to enable stakeholders—from business analysts to developers and testers—to stay in alignment and remain flexible.

Performance engineers will need to acquire broader skills, especially in a DevOps context. These skills are likely to include additional approaches to software testing such as test-driven development (TDD), business-driven development (BDD) and even model-based testing (MBT) and how to complement these with performance testing. Data engineering, systems monitoring, and observability are also relevant competencies. They will also need to learn how to work effectively with a wider range of people and to be able to share accountability with the teams they work with. This profession is becoming 're-focused' on its true purpose: the user experience.

The interactions with the business teams are equally important in further understanding the requirements and validating the accuracy of the tests. This takes up to 18% of the total time spent by performance testers. These interactions are particularly important for new applications that do not have any history of performance indicators in production.



Question 7: Which of the following best describes your approach to 'performance engineering tooling'?

Modern trends in tooling

Our survey results indicate 38% of respondents have a centralised, enterprise-wide performance testing and monitoring solution and another 32% have centralised tools available if they wish to use them. Overall, the role and purpose of these centralised teams are expected to change. They are less likely to select and manage the various software tools according to the experts we interviewed. Now that the tools are being used across the engineering organisation, they are likely to have an influence on the choice of tools. The budgets are also moving from the quality assurance (QA), or testing, department to the VP of software development.

Some commercial software tools may limit who can use them as part of their license agreement, for instance to specific people in the organisation. If so, these tools may be less suitable for teams who want to do performance testing either at the project or team level, or continuously.

Getting ready for AI

As software development accelerates, performance engineering needs to keep up. Artificial Intelligence (AI) could help in several areas that are currently time-consuming, including: keeping the test automation scripts in sync with the software being tested, and generating test data based on production traffic.

AI may be able to help assess the relevance of pre-production performance testing too and, where it detects flaws, adapt the testing to cover some of them. We have been able to sample music at 44,100 Hz for decades, but can we also sample production traffic and behaviours just as frequently using AI to adapt the system so it continues to perform optimally in the circumstances?



Practices
of Performance
Engineering

The illustration features two stylized human figures, a woman sitting on a stool and a man standing, interacting with a large, light-blue digital interface. The interface displays a grid of blue and orange squares. Above the interface, the text 'Practices of Performance Engineering' is written in a large, white, sans-serif font. The background is a vibrant blue with numerous white and light-blue diagonal streaks of varying lengths, creating a sense of motion and data flow. Several icons are scattered throughout: a dark blue square with white code symbols '</>', a light blue square with a white gear icon, and a red square with white curly braces '{ }'.

Now that we understand the impact of the corporate culture and the organisational structure on application performance, it is time we provided deeper insights into how companies practice performance engineering. To reduce technical debt, comply with the release cadence, and maintain market readiness, business leaders are adapting their approach to performance engineering.

Chapter 3: Practices of performance engineering

As well as showing you what your peers are doing, the following chapter also aims to help you improve your methodology and identify areas of enhancement.

Performance engineering is demanding

As a short definition, application performance relates to the duration of the execution of tasks by a computer system and the load such a system can handle.

We believe the end goal of performance engineering is not just to understand end-user experience, but to ensure the various aspects of application performance

are constantly met. These aspects may include: reliability, availability, scalability, uptime, responsiveness, speed, and so on. For market-facing applications, telemetry related to the application's health should be correlated with the achievement of business goals and feature usage. The telemetry may include: traffic, session length, new account conversion, number of active users, bookings, return rate, etc.

Companies need to invest concurrently in people, processes and technology, although they can be flustered by the scale of challenges this investment imposes.

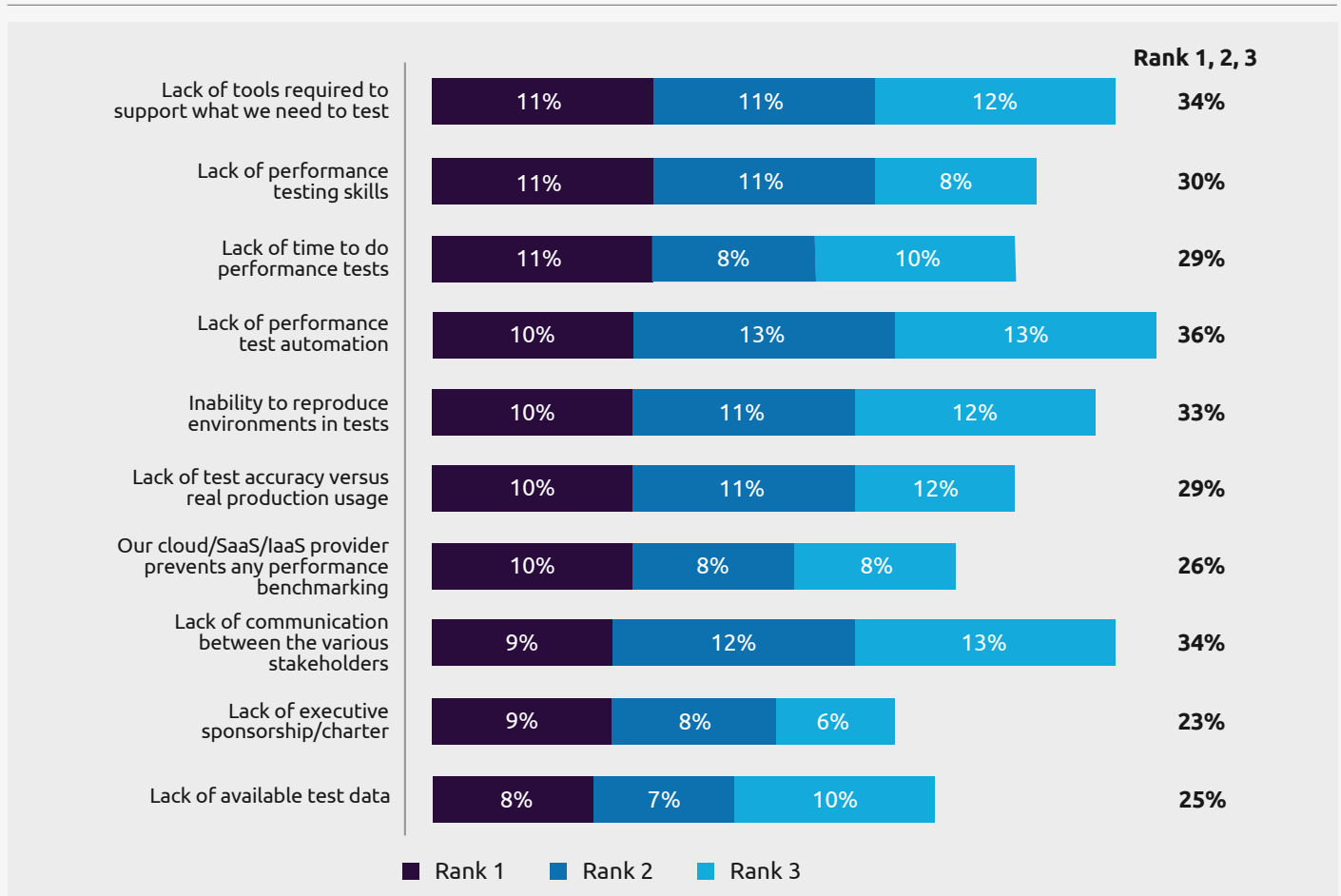
"34% identify the lack of suitable tools as a key challenge"

Selecting suitable tools seems to have been a recurring concern since the early 2000s. People's perceptions of the lack of tools may limit their thinking and their ability to identify tools that can help them address the testing they want to do. This perception may also be out-of-date as the range of both licensed and open source tools is actually quite impressive.

Almost all of the tools are capable of producing adequate user load from the cloud and on-premises machines working under different conditions. Further tools can provide a clear analysis of web application performance, pinpoint issues and identify bottlenecks. Many tools provide a free hands-on trial so they can be easily evaluated.

Adaptive content serving

As we mentioned in the first chapter, 10% of users leave the BBC website for every second of delay. The BBC implemented smart performance engineering, where they disable various non-critical elements on web pages when the site slows down due to load. As one of the BBC engineer said, "These will be low-importance things — such as a promo box at the bottom of a page — that are expensive on the server and few users will miss"⁵



Question 8: Which of the following are the main performance challenges that your organisation faces today?

Aspects differentiating load testing tools include: the range of protocols they support, the ability to automate and simplify the test design phase and the simulation of real-user behaviour. Other differentiators include: the speed to root cause analysis and the integration within the entire toolchain. The ability to minimise the maintenance of performance tests scripts as code evolves is also a significant benefit.

We do recognise the complexity for smaller companies to acquire the necessary tools and the skills to use them effectively, especially since the more complex a system is, the more investment in tooling is required. Even so, we still believe the main challenge is less about the tool than the methodology and the required skills.

Recommendation:

Many product managers see performance testing as black art and are unsure what's important when selecting tools for it. At a minimum, such tools should integrate within the software development lifecycle and improve the efficiency of test activities through automation.

Many respondents felt that having a lack of knowledgeable experts is one of the greatest impediments to achieving potent performance engineering. Their perception of performance engineering may vary significantly depending on if they focus on the topic or if they reduce it to a solitary job title; and the design choices, scope of activities, and required skill set varies significantly between companies.

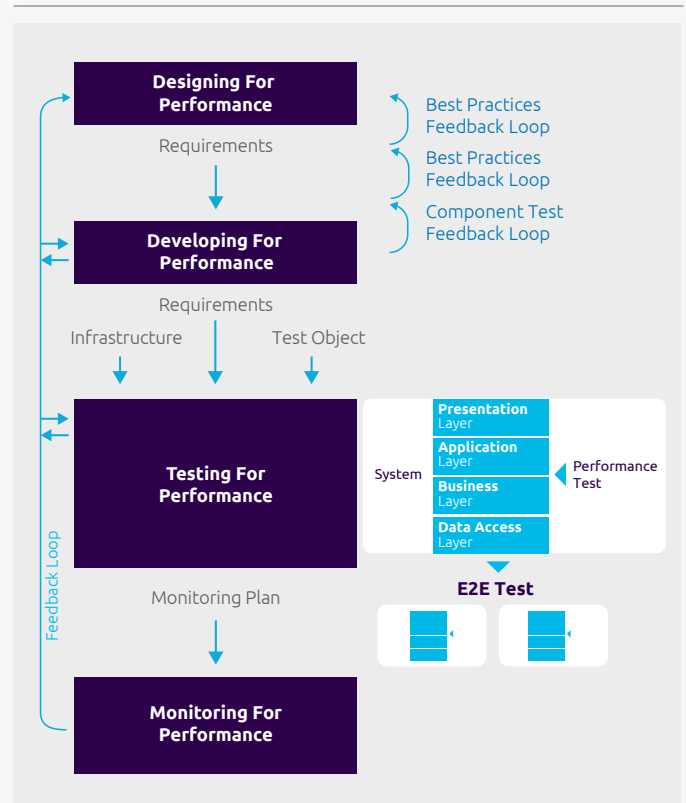


Figure 2: www.tmap.net/building-blocks/performance-testing

The role and responsibilities of performance engineers includes, but is not restricted to, the definition of design guidelines, design and code inspection in collaboration with developers, pre and post-deployment performance testing, and the monitoring of the system behaviour in production. Performance testing throughout the software development lifecycle is illustrated in the following figure.

“30% consider the lack of skills as the greatest impediment”

The potential range of skills is breathtaking. For instance, performance engineers may need to be involved in architectural decisions and configuration discussions, as well as understand the limitations of the networking, databases, storage and so on. They may also need to contribute to the business continuity planning as part of an overall disaster recovery strategy, collaborate with business to anticipate changes, be knowledgeable about cloud infrastructure and configuration settings, and be able to adapt proactively to production telemetry.

“25% lack available test data”

A functional tester with an appetite for performance testing should not be expected to run this activity immediately. Performance testing requires additional expertise from a design standpoint. Performance

Problem solving and a willingness to learn over expertise in tooling

Some interviewees are finding it difficult to upgrade the technical skills to keep pace with technology changes and new versions of development frameworks. As a consequence, performance testing is often reduced to the knowledge of a tool.

A multinational retailer had been using a commercial load testing tool for a decade. Given the cost of a license upgrade, the performance testing team had to perform their tests using the number of concurrent virtual users they already had. In-the-lab results were extremely positive but, unfortunately, didn't reflect the actual performance in production. The team believed that the resulting problems in production were not their responsibility as they had 'tested' the software.

When they were introduced to another tool and way of doing the testing, the performance testers blamed the new technology for providing misleading performance results. It took a 2-day in-depth workshop to help them realise their tests were inaccurate. Once they addressed these inaccuracies, they finally established a long-term alignment with the business' expectations.

adds depth and complexity to the work. The software needs to function under additional constraints such as time, memory, capacity and connectivity. New classes of bugs and issues can emerge when code needs to support concurrent activities. There are many flavours of testing, each used to focus on a particular subset of the behaviours of the system. Therefore, the people involved need to understand the various flavours, to acknowledge that the results can be unpredictable, and to recognise that the accuracy depends on both infrastructure readiness and integration with production analytics.

New skills can be developed via technical classes, through subscription to specialised sites, by joining online communities⁶, research through academic research⁷, guides⁸, and by participating in relevant events⁹. Wherever practical, upgraded knowledge should be practiced immediately by applying it on a smallish project.

The purpose of performance testing includes determining how an application behaves under multiple real-life contexts. According to ISTQB, test data refers to the 'data created or selected to satisfy the execution preconditions and inputs to execute one or more test cases'. It is a fundamental enabler to make performance testing fast, rigorous, and stable.

This low percentage of 25% surprised us. In our experience, companies underestimate the fundamental issues they face with the lack of proper test data lifecycle:

- The creation, combination and versioning of data is time consuming, especially when complex and multi-sourced. Even so, without it, test coverage risks being low and testing may miss critical combinations

- Performance testers are highly dependent on other teams to get access to data. Once the data has been consumed by a test, most testing teams rely on a database administrator to refresh the databases. Wait times for test data can jump to several days, even weeks, which prevents performance testing activities from fitting within the software development and release cycles
- Test data needs to comply with regulatory and organisational standards, especially with CCPA¹⁰, GDPR¹¹, PCI¹² and HIPAA¹³. Using real user data may break the

We should not overlook test data

One of the largest watch retailers had to migrate their SAP warehouse solution to SAP 4HANA. Their expectations for representative test data was significant. The team came to the conclusion that they had to make a copy from production as the source of test data. For various reasons, as the backup procedure took 2 days to complete, the test data could only be used once. In the end, the performance test was not sufficiently representative. Worse, the test could not be reproduced afterwards.

Without a proper methodology in place, it is nearly impossible to identify the ideal test data, which covers all application combinations with a smallest possible data set.

law and trigger large fines and additional penalties. Companies need trustworthy, relevant data that does not include any personally identifiable data (PII).

The challenge of accessing suitable test data and right-sizing it on demand is underestimated even by many performance testing specialists. It is also important to have not just the volume of test data but also the right

spread of combinations. Just having 'Joe Blog 1', 'Joe Blog 2' a thousand times is not sufficient. The data needs to be realistic to be valuable for performance testing.

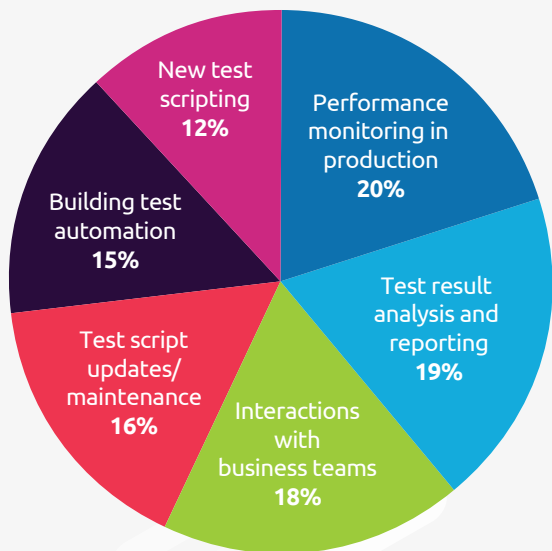
We encourage you to investigate the use of synthetic test data (not dummy data) as a way to preserve all the characteristics, diversity, and density of locked-up production data. Ideally, companies need ways to deploy a reusable, application-wide self-service mechanism.

The performance engineering agenda is packed with must-do activities

The role of a performance engineer has evolved from focusing on testing and tuning to the position of driving the ambition for the company, where they focus on the full lifecycle of application performance. This chart shows the amount of time spent on each of the key activities.

18% of the time is spent interacting with business teams. While this is a strong signal of maturity, it is often still done in later stages of the project lifecycle as point interactions rather than ongoing collaborations.

Businesses are intimate with their markets, they know their competition, and represent customers. All of this is relevant for performance engineering. Therefore, both the business and technical teams can collaborate on user persona, user flows, capabilities and requirements, and how they evolve over time. Of course, the business needs to provide feedback that is specific, timely, and meaningful.



Multi coded per option
Total respondents = 515

Question 9: Overall, what is the proportion of time your teams spend on each of the following performance engineering activities?

Performance engineering remains a labour intensive discipline with 12% of time spent on new test design and 16% on existing test scripts updates and maintenance. As the release cadence accelerates the need to create and update scripts more frequently will continue to rise. The test scripts need to remain current and adapt at the same pace as the testing. Performance testing is at risk of being held hostage if their role is to edit and update scripts manually. Furthermore, the more time and energy that's spent on scripting the less time is available for more strategic and valuable work.

Choosing more capable and supportive tools can substantially reduce the time needed to maintain test scripts. Helpful features range from providing ready to use technology and application frameworks, and accelerating the parameterisation of new tests, to re-purposing functional tests scripts for performance.

Other capabilities such as assisting engineers with understanding code changes and helping update scripts without having to re-write them completely are examples of power features that help performance engineers spend more time on analysing test results than on creating or updating test scripts.

Performance testing needs to be integrated with continuous delivery pipelines in order to support the speed of release across the life cycle — i.e. from component to the full system-wide load tests. 15% of the time is spent on the automation of performance tests. Suffice to say, this is critical in providing developers an early feedback loop and visibility on regressions. This includes the ability to test the main components during the build process, handle project maintenance

automatically, and adapt load policy continuously based on observations gleaned through regular monitoring.

These activities differ significantly from automation of functional tests. For example, performance testing often depends on provisioning suitable infrastructure. Let's take the example of a test intended to validate the authorisation credentials flow. A successful login is supposed to take no longer than 1000 milliseconds. However, the test results show that the test takes 2000 milliseconds at a single remote access point. Although the component is not performing to expectation, the root cause is uncertain. Performance testing then includes finding and investigating potential causes for the slowdown of these logins.

Not all types of performance tests are suitable for full automation: intense, long-running, large-scale, end-to-end performance testing might be better suited outside the core CI/CD process. We intend to research this topic in greater depth in next year's report.

Performance monitoring is a key enabler of a culture of trust. This activity is mostly done in production (20% of the time) to monitor:

- How the app is functioning and performing, and the context under which it does not do so adequately
- The various components of the system, e.g. network, memory
- The behaviours of end users, e.g. usage patterns, flows
- The sentiment and feedback of end users.

We believe monitoring should be extended across the entire lifecycle and have comprehensive visibility into all the activities related to application performance. Monitoring has a place within a shift-left culture. It can provide developers with a solid understanding of how the various components of the architecture are being used in pre-production. This proves very useful in anticipating the repercussions of planned changes, optimising database queries, and tuning system configurations.

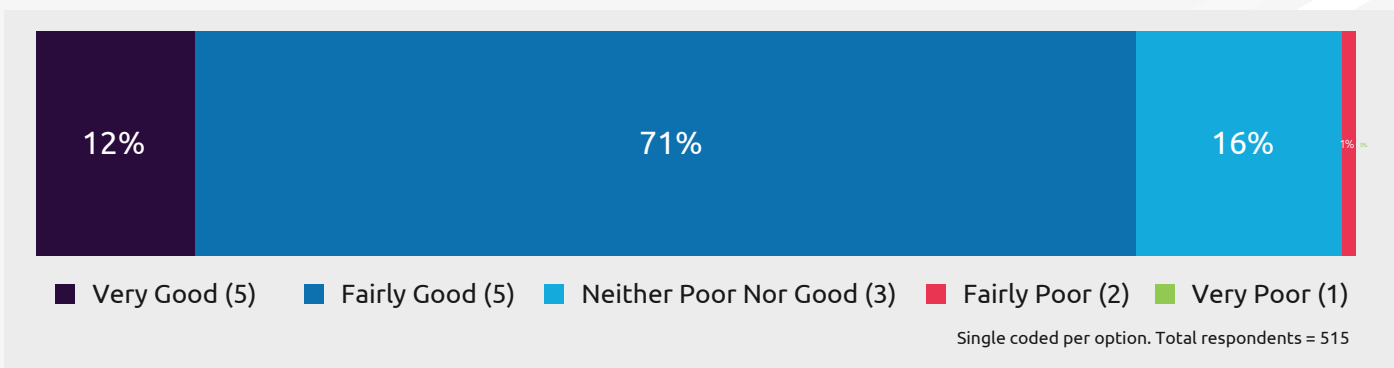
The urgency to increase team awareness

71% of respondents say their organisation is able to do a fairly good job of testing and finding performance issues before production. We suspect some people picked this answer because they hope it is accurate rather than because they have justified evidence that they can find many performance issues before production.

“We are not gonna do performance testing because it does not reflect production”

With these words, an IT leader at a large UK retailer provides a dramatic shortcut applied by far too many people.

In our experience, there is too little understanding of what performance engineering (and related subjects) actually involves. Performance is too important to be left to performance testers¹⁴. Often, performance engineering is regarded as technology-centric, rather than providing business-related recommendations. Done well, it supports long-term viability and sustainability and is a matter for all rather than for one or two ‘experts’.



Question 10: On a scale of 1 to 5, with 1 being poor and 5 excellent, how would you rate your organisation's ability to run accurate and reliable tests to detect performance issues before production?

There are many ways performance engineering can fail through poor behaviours¹⁵

An engineering company, a purported world leader in the performance of its systems, scrimped in their testing. They believed the only testing they needed was having a few people press buttons before going live to see if 'it' could handle several users. Even after having several bad deployments go live, they still don't plan for performance testing and they fail to see the cost of getting it wrong.

An insurance company purchased monitoring tools, but failed to use the results, or make them available to the engineering team.

Another company let support staff in production decide how to solve performance issues. Rather than solve the problems at an architectural/strategic level, they did what they knew how to do already. For instance, they added more systems (extra load-balancers, a new database, and so on). When measured solely on service level objectives, a production team may be tempted to buy unnecessary additional capacity rather than fix the causes of poor performance.

The most effective approach of performance engineering starts as early as the requirements phase (shift-left). However, even trying to obtain basic performance requirements can be a challenge for many projects and teams. Too many companies fall into the worrying pattern of using the often poor performance of their current system as their success metric, for instance "it needs to be as fast as the current system" or "no worse than the current system". This approach is a worrying and disappointing trend that continues from year to year for many teams. Given the vast growth in system performance being an intrinsic part of a business's viability, we are disappointed to discover only 12% of respondents rate their organisation's abilities to run accurate and reliable performance tests before production as excellent.

Test environments are failing performance engineering

"33% of respondents face difficulties in reproducing a test environment that mirrors production"

Often, problems occur around the lack of representability, unrealistic requirements and — again — the lack of collaboration with the infrastructure team. Test environments don't necessarily have to be an exact copy of production, but they need to have similar characteristics. Cloud infrastructure now facilitates the reproduction of highly representative production environments.

Missing the whole situation owing to technical prejudice

A provider of service solutions to the maritime industry launched a major project involving several solutions migrating to the cloud. The belief that the cloud infrastructure would be the solution to any performance bottleneck resulted in the absence of performance engineering strategy. Soon, the system faced severe performance issues. The recommendation was to augment the capacity of the various components, and more specifically the database. With such critical business impact, management approval was expedited. Early signs of improvement misled the team. The same performance issues with the same remediation reoccurred a few months later. The problem slowly grew worse over time and, again, went unnoticed until it was too late. After several similar cycles, a proper investigation took place. The root cause resided in poor database indexing.

The purpose of database indexes is to reduce the number of data pages that have to be read in order to find the specific record. Performance of the system often depends on using indexes effectively. Because the team had ignored the indexing, they spent lots of time and money trying to solve the wrong problem.

A performance engineering approach could have identified the root cause much earlier and helped avoid unnecessary delays and costs.

What happens when devtest and production are out of sync

A multinational oil and gas company used Chrome to measure the end user response of an internal application, whereas the corporate IT policy enforced Edge as the browser of choice. At the time Edge used a proprietary rendering engine. The tests were not fit for purpose because of the differences between Chrome's and Edge's rendering engines¹⁶. The differences in system resource consumption and effects on battery life impacted end users with limited memory laptops. A late audit on the test approach revealed additional gaps in methodology and provided strong recommendations in the company's move to DevOps.

An insurance company customer migrated one of their core business-to-consumer systems to the cloud, while keeping sensitive data on-premises.

The team did not anticipate that network limitations between the cloud and their internal environment could cause huge delays and timeouts. Customers in specific geographies started to complain. The poor network performance had such an adverse impact that the entire service was put on hold.

An immediate thorough performance test was conducted, which provided technical remediations. This issue could have been anticipated by including end user devices, browsers, and connectivity in earlier tests.

A company providing speed control systems for a Scandinavian company used mechanical disks for the test environment when the production was using SSD discs¹⁷. The tests were misleading and the test environment had to be changed to make the tests sufficiently representative. This obviously incurred additional unexpected costs that could have been avoided through proper collaboration between the various teams.

Performance tests are particularly sensitive to the details of a given runtime environment. While functional tests can run on a 4 core laptop or a 32 CPU rack server and produce the same results, the same is not true of performance testing. Infrastructure counts. Thus, for a performance test to be reliable, the infrastructure in which the tests run must be consistently appropriate to the need at hand.

A global pharmaceutical company intended to validate the performance of their business-to-business website. After simulating the load with several thousands of concurrent virtual users, the team realised this was not representative of

production usage and reduced the scope to 50 concurrent users, which was much more realistic. Here again, costs could have been minimised with an earlier cross-team approach for the application performance requirements. Such requirements were only designed through this empirical, capacity-based approach.

As a footnote, they reached the right answer by accident, where they chose their solution because of the technology they already had rather than creating a model, establishing the requirements, and designing based on these requirements.

Recommendation:

The ability to rely on consistent environments being available on-demand is a fundamental requirement for Agile performance engineering, one that enables performance to remain vital given the faster rate of change DevOps often imposes. Many teams use configuration management tools to reduce the complexity of managing infrastructure delivery. They can also use the same tools and the production configurations to help them accurately, easily, and reliably configure test environments. There will be essential differences in the configurations for specific test environments. Using the identical tools and sources makes these easy to identify, check and control.

Having a single source of configuration truth can also be used to ensure any test environments are correctly and automatically configured and operated. This ability to quickly and correctly establish suitable test environments on-demand facilitates rapid experimentation and innovation.

Let's flip our DevOps mindset around

OpsDev brings tools and practices from operations into software development and testing. It also helps development teams to consider the operational impact of their work before it reaches production. As we mentioned earlier, using configuration management from production to help configure and create test environments is one aspect. Using the same monitoring tools in production and testing is another as the developers and testers can see how their software will be monitored once it is launched.

We view OpsDev as an iterative process intended to enrich development and testing activities. Performance testing and monitoring are both independently valuable. However, the value increases significantly when they are combined effectively. We believe performance monitoring in production has become a standard, but is only applied to one third of all projects.

Using application performance management (APM) tools help teams understand real user behaviours for existing systems and define an objective baseline that will be used during performance tests. Being able to leverage facts is critical to avoid unrealistic expectations from the business side.

Production monitoring data provides access to insights into the application behaviour and the underlying interactions. Usage patterns can be modelled based on different network connections, and as research published by Google¹⁸ discovered the type and performance of the network can have a major impact on the usage. Performance can be improved by measuring the end-to-end performance from the end-user perspective.

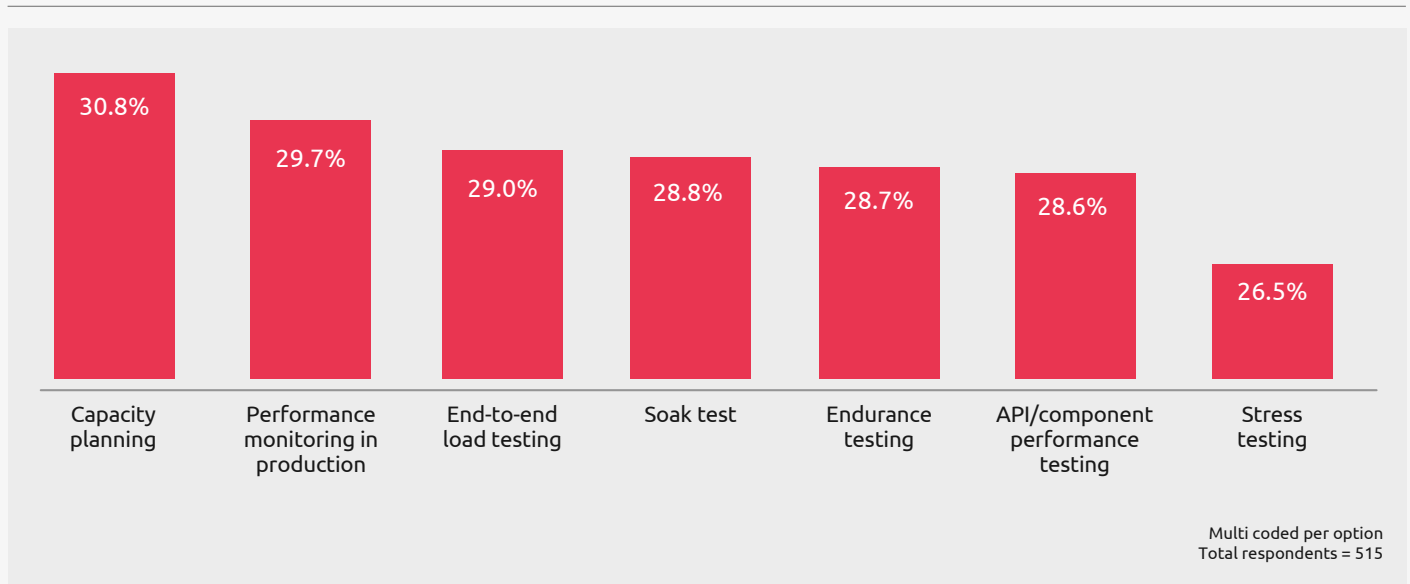
Technical details of the production environment, such as SQL breakdowns (execution plans), infrastructure components (CPU, disk, network, and memory), and heap size, can facilitate the creation of more accurate tests. This approach of using production information as a source can also help identify the differences between the various staging environments.

Monitoring and analytics tools can also be used during the various pre-production phases. By using common monitoring tools throughout the software lifecycle, a common language is established across the mix of development, testing and operational roles.

Some companies monitor their end users' feedback to help them catch performance issues quickly. When problems are reported by users, companies can investigate immediately. In parallel, they can also make fast, appropriate, and positive business reactions. As an example, they might add an announcement online and contact customers reporting problems to help ameliorate the situation. When users know their feedback is important they often view the company more favourably, so everyone wins.

For internal packaged applications, some organisations combine interviews of business stakeholders with the analysis of incidents logged in their service management

tool. Soon, monitoring tools will also detect if a user is over-clicking or refreshing the user interface as a sign of frustration.



Question 11: For the above activities, what %/proportion of all projects typically undergo each of the named performance engineering?

The benefits of correlating mobile testing and mobile analytics

In order to enhance the effectiveness of the tests, a national gambling company combined data from mobile heatmaps and web analytics. The engineering team was able to capture heatmaps of end users' flows. The business used them to re-prioritise development, and the quality assurance team to address the 'what to automate' question. Eventually, the combined data also proved valuable to define the various prerequisites to the load test plan (scenarios and steps).

Engage end users to get a system that is fit for purpose

A hospital network invested in a medical application to facilitate the creation and management of patient records. The development team assumed doctors would create or update records immediately after each consultation. However, huge spikes hit the system during lunch and dinner times because the staff batched their data entry until their next break.

In addition to the remediation of performance issues, the development team conducted seven interviews with the end users. This allowed the team to further understand the habits of the users which they used to help prioritise features and usage flows. These interviews led to a higher quality app, and strengthened their relationships and trust between the various stakeholders. Ultimately, it led to a system that was fit for purpose.



Enterprise spot light: MAIF

MAIF is one of France's leading insurers and ranked 1st in customer relations in 2020¹⁹. To achieve and maintain such a high level of service, excellence is required at all stages of the relationship. Reliable and fast applications are essential to enable the company to meet the needs of their customers in the best conditions.

“The culture of performance at MAIF? An application must respond within 2 seconds. It is a standard in the company”

At MAIF, performance engineering applies to internal applications used to manage customer records and accounting as well as to web applications used by the customers. The performance team manages all the performance engineering activities, from performance testing to application performance monitoring in production. Application performance objectives are defined jointly by the performance team, the project team (IT), and the product team (business).

A Requirement Analysis Document (RAD) formalises this collaboration and allows the stakeholders to specify the expected levels of performance and the test scenarios. As a basis for the common performance culture, the RAD document contains:

- Performance risk analysis: the identification of the software building blocks on which the performance engineering activity should focus because they represent the highest risk of bottlenecks, or because they are the most critical
- The user paths to be tested, to reproduce realistic usages
- The types of tests that will be performed (e.g. stress testing or load testing) and what are the expected performance levels
- An architecture diagram of the elements to be tested. Performance experts use this to help ensure they understand the details and limits of what they are expected to evaluate.

The performance team then performs the performance analysis and produces a performance strategy document.

At MAIF, performance engineering is based on a centralised team of performance experts, who chargeback their work to the various project teams. But the company is moving towards greater agility and a performance engineering organisation where project and product teams are more involved in defining performance objectives. This trend aims to decentralise the performance expertise, to bring it closer to projects, and thus make performance a structural element of the project, as early as possible in the development cycle.

Changing to more agility to deliver smaller and often.

The traditional cycle of software development, testing and release was driven by Information System Releases

(ISRs) and took place 2-3 times a year. An ISR integrates 20 to 40 projects, and the complexity of the system used to require six months to test the entire system. These projects are known as bricks.

Today MAIF's teams are more Agile and much more reactive. ISRs still exist, but they are lighter-weight. Each project puts the software bricks it has developed into production when that brick is ready. Releases are therefore much more frequent. So how can the performance engineering strategy be adapted to this acceleration of release cycles?

The number one criterion is to have a test platform that mirrors production on an ongoing basis. This is a prerequisite for carrying out conclusive tests. This is also the main challenge. The test platform is updated more frequently, every 2 months, and only with projects that have delivered new code and need to be tested. Performance testing now tests subsystems rather than the entire system. MAIF's performance offering is being overhauled, and in future the projects will directly update the test platform, as soon as the code evolves, so that the platform is constantly representative of production.

The second axis of evolution is to be able to test earlier. As for functional testing, the goal is to create performance test scripts as the changed code is delivered. And it is only by moving performance expertise into projects that tests can be updated in real time and automated as part of the continuous integration process.

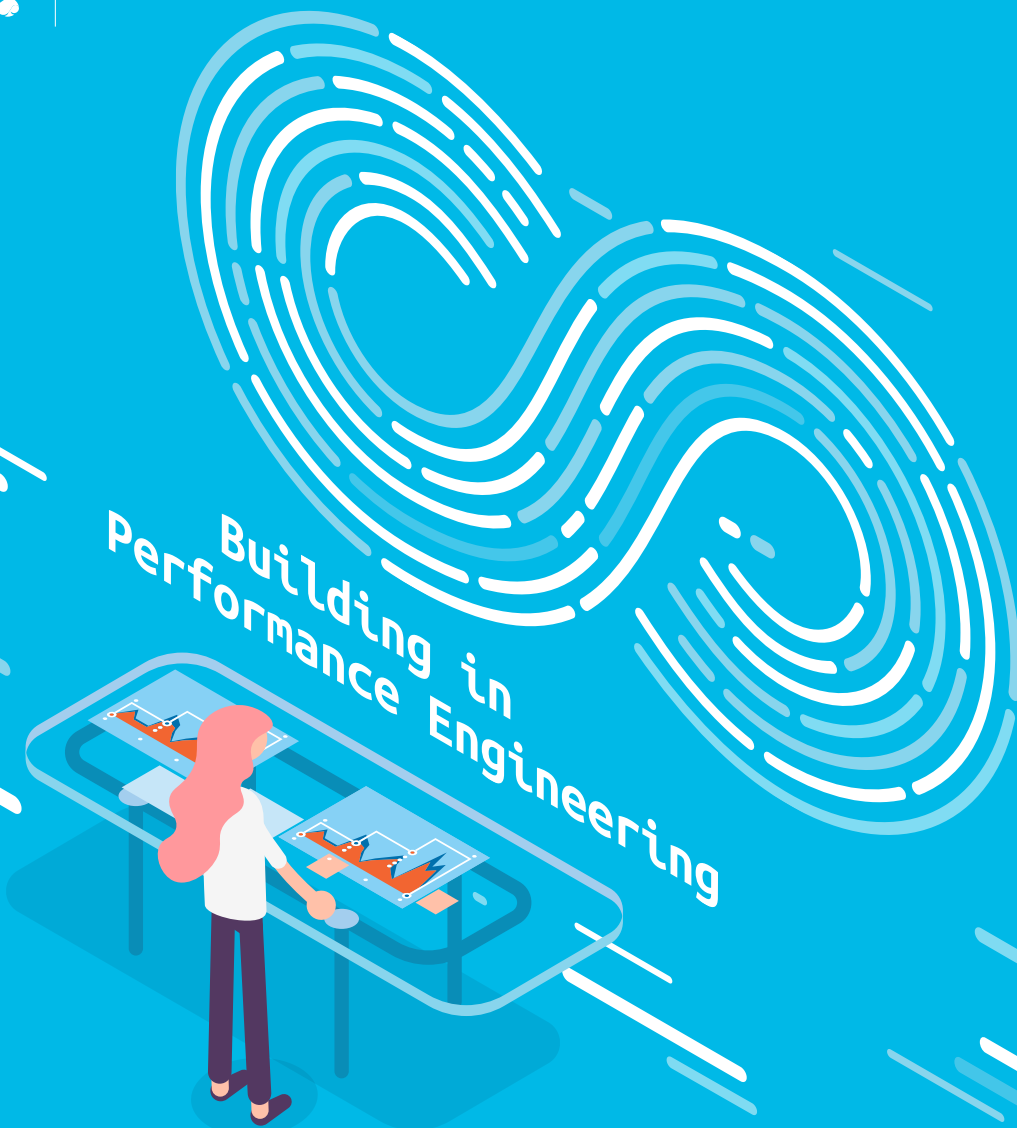
In this decentralised scheme, the main challenge remains end-to-end testing. When testing a brick, the

entire system must be running to reproduce production conditions. The aim is to generate the 'background noise' that will enable a brick to be tested in the realistic context of all the elements: this represents 200 to 300 servers and 80 performance test scenarios that take 3 to 4 weeks to rewrite when a new version is available. We can well imagine the complexity of the task.

Better communication is the key to successful agility.

In order to manage an increased pace of test platform updates and tests that need to be performed earlier, team synchronisation and communication are paramount. In order to fluidify exchanges, scrum meetings are held twice a week. They include performance experts and project teams. These meetings allow real-time communication, and therefore better reactivity.

To support the synchronisation of teams, MAIF improved its requirement analysis document, so that it specifies more precisely the business use cases that need to be verified. For example details such as specifying that the connection to the user personal space should not take more than 5 seconds is required. The overhaul of this central communication document helps to break down the silos between the project and performance teams. Communication is indeed the factor that makes it possible to bring together the performance experts, the project and the business teams, and in so doing, to speed up the production launch cycles while maintaining the same level of quality that allows MAIF to guarantee an excellent service through reliable and fast applications.



Building in
Performance Engineering

High-performance IT delivery is an approach that enables cross-functional teams to continuously optimise their activities and improve the overall software quality.

Chapter 4: Building in performance engineering

In this chapter, we look at how performance engineering activities are integrated with other activities to maximise the efficiency of our application delivery pipeline, based on the constraints engineering teams need to operate under. The objective is to identify technical ways to both optimise and innovate, while complying with core requirements: to contribute to the culture, minimise cycle time through automation, and increase accuracy through orchestration.

Continuous testing is the ongoing practice of testing across every activity throughout the application lifecycle, from requirements gathering through to production. The

benefits that continuous testing brings include faster feedback, better test coverage, and more developer involvement in testing. The main focus is to uncover and fix unexpected behaviours as soon as they are introduced, while ensuring the business value is being achieved as expected.

Embedding performance engineering into continuous testing

In order to optimise the delivery pipeline, we need to remove bottlenecks and inefficiencies throughout our performance engineering process. The figure below shows an example of a continuous testing pipeline. Tools are necessary to automate this process, reduce wait times, and reduce manual errors. However, tools can also add to the complexity and inefficiencies if they are not

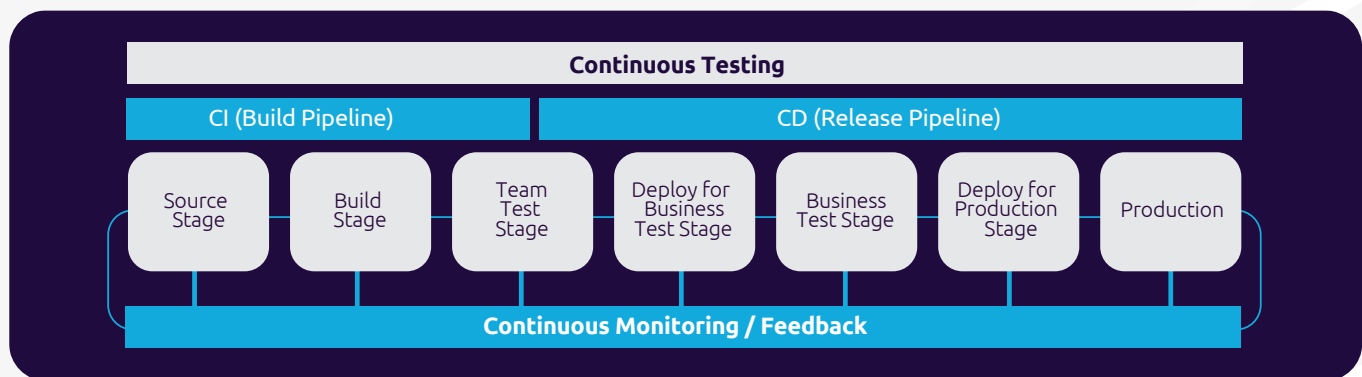
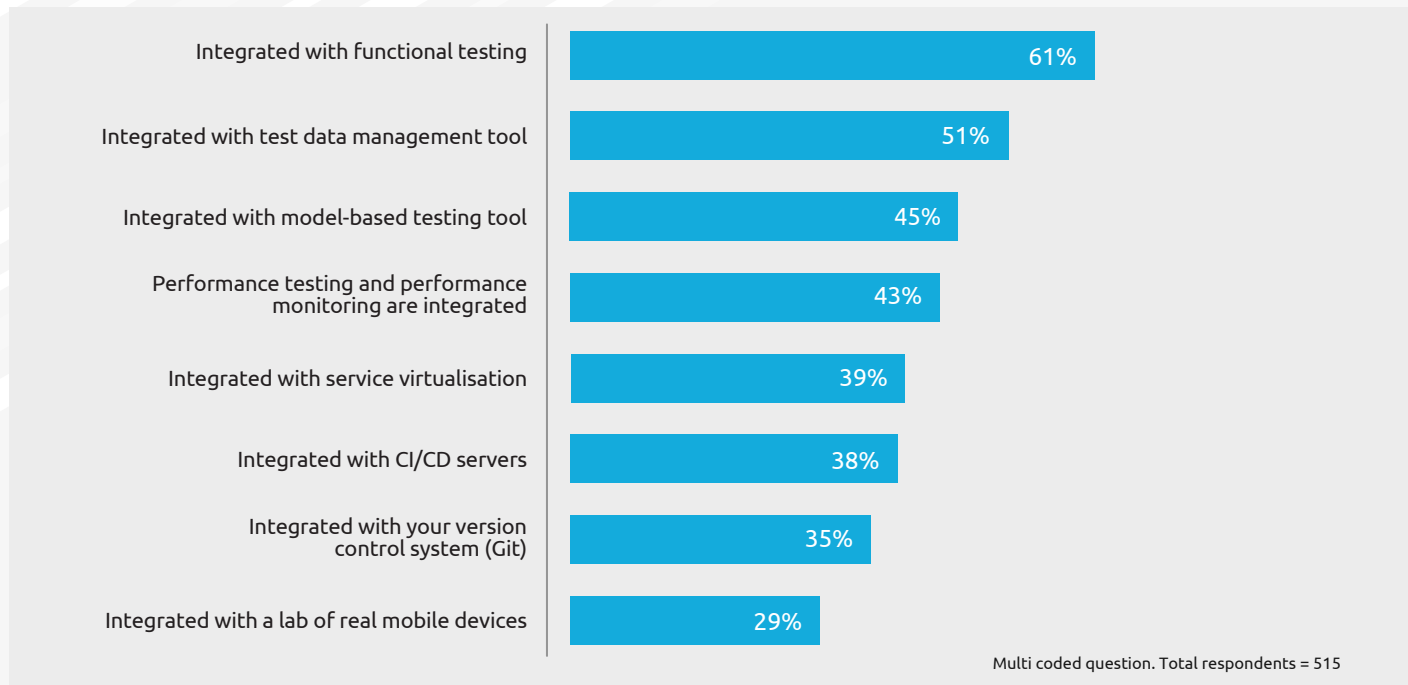


Figure 3: MAP for DevOps²⁰



Question 12: Which of the following applies to your organisation with regards to how performance testing is integrated into the software factory tool chain?

integrated. While the methodologies, activities, tools, and capabilities differ from company to company, and project to project, we will make a few observations on the trends from this report.

Most organisations are inclined to prioritise the automation of functional testing over performance

testing as it's easier to do and has fewer dependencies. Automated functional tests can provide some synergies in terms of measuring performance.

- They can be enhanced to record timing and performance metrics to provide an extremely lightweight single user performance test, for instance as part of a software build pipeline

- They can also complement traditional performance testing that uses protocol and API testing to test web front ends, as backend response times are insufficient in providing the visibility from the end user's perspective.

“61% integrate functional and performance testing”

During the interviews, a major insurance company mentioned they started conventional load tests but soon coupled them with automated functional tests to get an inner view on response time measurements. Some mobile testing tools can generate functional scripts that can be directly embedded into load testing plans. Team can create, record, and run one single script for performance and functional testing in a single and synchronised session. This trend is particularly relevant with Agile teams willing to benefit from existing functional test scripts to test user stories from the performance angle. The idea is really to simplify and accelerate the process. However, additional investment is still required to rewrite performance scenarios that were not necessarily designed for this purpose.

“45% integrate model-based testing”

Design requirements are important to build an accurate and representative test plan. However, many projects lack these, and their tests are bland and seldom include

scenarios that combine load with positive, negative, and edge cases. These weak tests limit the ability to react to in-sprint changes with confidence.

Model-based testing (MBT) generates automated tests by using models that represent the system. The work includes creating, writing, and updating the models in response to changing requirements. The models can be designed to recognise and accommodate the needs and expectations of end users at an early stage. MBT is already becoming popular to increase the accuracy of functional testing. However, it may offer greater potential for performance testing where it can model user flows and traffic patterns elegantly. Using MBT can enable rigorous testing that fits into the same cycle, the same sprint, while allowing stakeholders—from business analysts to developers and testers—to stay in alignment and remain flexible.

When the models are designed well they can help improve the accuracy and effectiveness of the testing. This enables risk-based testing: we can automatically focus on the right scenarios, rather than solely relying on our intuition and experience. Mathematical coverage measures let us know exactly what is being tested. Optimised automated scripts can then be exported to a performance testing tool, which will return the results to any test management framework. This becomes even more powerful when coupled with test data management and production analytics.

Performance testing activities are highly dependent on a large volume of data, for both existing and new environments (different contexts, different approaches). When test data is integrated with a test

data management tool teams can significantly reduce the effort required to prepare, refresh, and archive the data.

“51% integrate test data management”

Although 51% say they integrate test data management with their performance testing, the reality is likely more reflected in the remaining 49% of those interviewed, where developers and performance testers have no choice but manually fill the void, with negative consequences on either timelines, quality, or cost. Or worse.

“43% integrate performance testing and monitoring”

Only 43% of companies integrated ‘application performance monitoring’ with pre-production data.

By providing in-production app usage visibility, performance monitoring is becoming a fundamental tool in the performance tester’s toolkit. The objective is to improve the relevance of the pre-production activities through a proactive cross-team approach. A unified performance platform combines data from monitoring to test and assesses the quality of each build.

We want to make our testing very accurate. Monitoring tools can provide additional and deeper insights into the performance testing, e.g. by recording the behaviour of threads on the servers and by measuring the ecosystem.

We are aware of two forms of performance monitoring: real user performance and synthetic requests. Both forms can incorporate information from monitoring tools and the information can be used to calibrate the performance testing.

“It is a capital mistake to theorise before one has data” Sherlock Holmes (Arthur Conan Doyle)

One of the major retailers in France decided to develop their own billing system. This new system was supposed to be in production 3 years after it started, with an initial delivery after 6 months, and then quarterly updates. Various Agile teams were in charge of delivering the multiple components of the end system. The teams implemented what they considered to be advanced continuous integration level, supported by Kubernetes. They had an open source solution for generating data, a solution for mocking up unavailable services, that had to complement the load testing activities.

As very limited data was available, they struggled to measure the end-to-end performance and assess the design of the overall architecture. As a direct consequence of this, the development effort suffered from unacceptable delays and the team was eventually dismantled.

Real traffic can be used to design synthetic probes that run in production. The results of these synthetic probes can provide speedy and precise feedback about the production system, e.g. load variations, degradations, and gauge end users' experiences. [The 7 pillars of performance](#) are proactively measured: response time, capacity, efficiency, scalability, stability, resilience, and the impact of the instrumentation. A few stakeholders also shared that integrating performance testing and monitoring proves useful when requirements are unclear: risks are identified, documented and anticipated.

The remaining 57% will have to overcome organisational boundaries to reap the benefits of this integration.

Co-location is not the cure

One of the world's largest automotive technical organisations operating in 50 countries had the pre-production and production teams located on the same open space, on the same floor, in the same building. However, the internal structure and siloed responsibilities prevented stakeholders from engaging with their counterparts. Each team assumed that others were responsible for taking action. The diffusion of responsibility led to adverse business consequences. This is currently being addressed by a transformation aligned with DevOps principles.

"29% include real mobile devices during performance testing"

Poor performance is one of the top reasons for mobile user frustration yet not many respondents seem to use real mobile devices and virtual users to diagnose and resolve pre-launch performance issues. With several billion devices in use in extremely different contexts (location, surroundings, network, language, OS versions, gesture, sensors, etc.), it is extremely difficult to cover the wide spectrum of combinations. Many people in our industry seem to restrict the use of real devices to running functional testing scenarios, rather than also using them for performance testing. How can developers create mobile apps without regard to its performance on the underlying physical device?

Performance is approached from 2 different angles:

- Eagle eyes on the end user's performance: stability (crashes), responsiveness (response time, time to interact), ease of use (time to perform a transaction).
- Impact of the app on the endurance of the device. Performance tests can assess how much battery power is used by the core functionality of the app, and how much is being consumed by analytics (e.g. SDKs), checking locations, or constantly polling the server for ads to display.

Why include real devices with a cell connection — and not just Wi-Fi?

Network conditions can cause severe slowdowns, even with good quality networks. For instance, a transaction with a fair 3G connection will take 90ms to 200ms to fulfil instead of 5 to 20ms under Wi-Fi conditions. This generates 2 kinds of problems:

- **Functional:** An app user may not be able to complete a search activity because the network slows the communication so much that there are timeouts in the backend, ending the transaction before it completes
- **Performance:** Multiple transactions under various network conditions overlap, resulting in higher session concurrency. This creates performance slowness for all users, which can result in major outages.

The walking devices

A security agency needed to equip their agents with an encrypted communication application, which had to be available before for a David Guetta concert and the 2016 World Cup. To address performance, various techniques were combined to test the system:

- At 3 real-world locations 'walkers' used their mobile phones on different cellular networks and strengths
- A driver drove around with several devices running automated tests to assess performance under real busy urban conditions. These tests were monitored remotely by the engineering team
- End-users beta-tested the apps on their own devices
- Distributed load generators also simulated a total of 1000 concurrent users.

While this was non-trivial to achieve, it allowed an extremely accurate understanding of the candidate prototype. However, this approach can restrict test engineers:

- Some what-if scenarios are not possible, e.g. you are restricted to live networks
- Scenarios cannot be repeated accurately as the live network will change between executions, producing misleading test results and making it impossible to determine if issues have been fixed
- Testing across real networks could impact production systems, especially when load or performance tests are run.

This example is from several years ago. Today teams can leverage virtualisation technology (emulators, network, services, data) to re-create production environments at a fraction of the cost.

Towards a culture of metrics

We need to measure progress towards making our system resilient. We rely on telemetry to drive a continuous culture of improvement, increase transparency and visibility, and enable decision making. Computed metrics as part of quality gates prove invaluable in checking the improvement or degradation of the system.

As a prerequisite, information on application performance should be freely available without the need to go through lengthy approval, escalations, or favours. Achieving high levels of transparency might require organisational change to reduce any forms of resistance. As a minimum, we believe measurement is necessary in helping understand, predict, and troubleshoot application performance. It can also help prevent problems by enabling the system to be changed to ameliorate issues when performance degradation is about to occur. The concept of observability²¹ complements metrics as it focuses on assessing and improving how much of a system can be observed in terms of its behaviour and performance.

It's worth considering what might constitute useful performance indicators. In our view, 'Quality for DevOps teams'²² provides a valuable approach in helping identify

'good' metrics. In particular, it focuses on how a well-balanced set of metrics reveals aspects of both the effectiveness and efficiency of our activities.

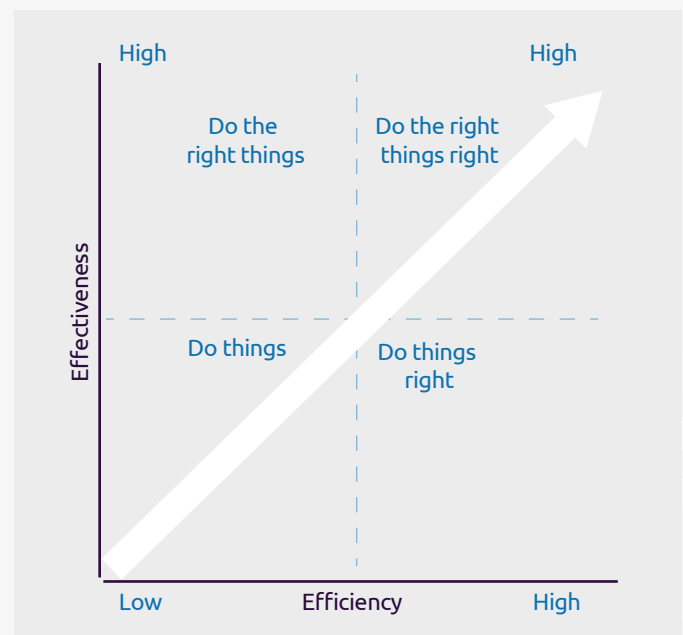


Figure 4: Productivity increases when both efficiency and effectiveness are improved²⁴.

We could also select metrics based on their usefulness, e.g. critical performance indicators (CPIs), and those that are key (KPIs). The following definitions are based on our interpretation of applying quality characteristics defined in the ISO 25010 standard²³:

- Effectiveness is the accuracy and completeness with which users achieve specified goals. For instance, useful business KPIs include customer retention rates, and rates of increase in advisory assets
- Efficiency is the resources expended in relation to the accuracy and completeness with which users achieve goals. For example, useful operational KPIs include an improvement in velocity. A quality CPI might be the defect slippage rate to production; a cost KPI could include the percentage of effort saved as a result of automation; and a speed-to-market KPI might be sprint velocity.

The metrics in this survey were selected to reflect high-level trends, and are not exhaustive. While 'response time' always scores high (57%) and production incidents (59%), we need to expand our thinking about measurements to include the effects of the work, including the business results, release quality and user sentiments. This discipline seems to be fairly introspective and disconnected from the larger context. The lack of alignment may propagate a culture of blame around outages and crashes rather than a culture of improvement.

While it can prove difficult to obtain a single view of performance when so many tools are simultaneously in play, organisations need to have a collective understanding of the value of what's being delivered. Any list of metrics needs to evolve according to the value it provides and the inspiration it generates. To provide our community with an objective perspective on which quality metrics are most critical for DevOps success, Tricentis commissioned Forrester to investigate the topic²⁵. This

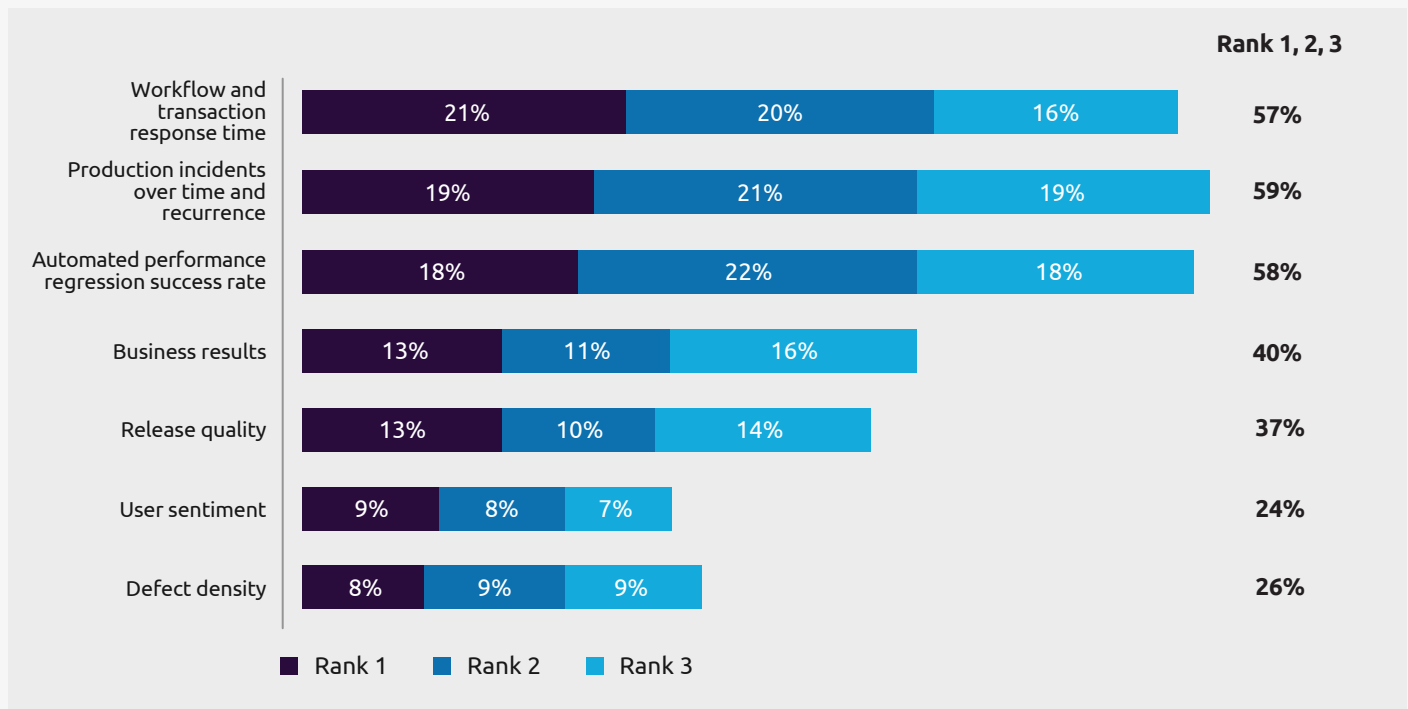
report comes with valuable recommendations on the lifecycle of metrics.

Obtaining a single view of performance is achieved through a fact-based, practical approach, one that encourages the correlation of technical KPIs with either commercial (websites) or productivity (packages apps) ones. Real-time comparisons mitigate the impact and cost of problems. Some companies have even created tailored synthetic KPIs, as an aggregation of standard technical metrics and business results.

For online transactions, the following commercial metrics could be considered:

1. Overall business performance: conversion rate, traffic, page views, session duration, bounce rate, etc.
2. Customers: repeat rate, customer lifetime value, etc.
3. Orders: average revenue, number of items
4. Profitability: customer acquisition cost (amount spent on performance engineering / number of customers), cost of selling, margin

Not all performance indicators can be structured. For instance, sentiment is difficult to quantify — in which case, how can you act on it? To facilitate the usefulness of user sentiment to performance engineering (24%), one could consider setting specific KPIs brought by natural language processing algorithms. This helps determine the polarity of feedback (negative, neutral, positive), its value (actionable, solved, archived) and the type and



Question 13: When it comes specifically to measuring application performance, what are your top 4 most important metrics?

level of emotion (happiness, calmness, disappointment, annoyance, frustration, anger). Social media activities are scanned and this can trigger alerts should negative impressions be shared. In the long run, comments could be used to remodel the existing strategy.

Green metrics for mobile apps

A repository of green metrics depends on the availability of measurable consumption data, inflows and outflows and use cases. To get a better understanding of the environmental impact of the system, we recommend selecting both device — and server-specific metrics.

For instance:

- Server CPU consumption
- Carbon footprint of electricity in the datacentre
- Occupancy rate and lifespan of the servers
- Number of device-server requests
- Size of a pre-installation on the user device (web plugin, mobile application)
- Cadence of updates
- Volume of cellular data generated by the app itself and by embedded libraries (SDK)
- Time-to-display measured across various contexts (type of devices, network).

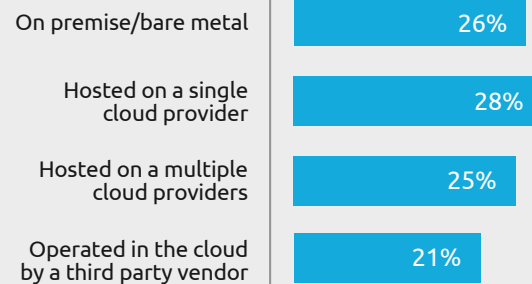
Performance engineering and the cloud

The benefits of cloud load and performance testing are very compelling: all the load required to truly test an application can come from external, elastic, on-demand sources. Teams don't have to build it out and maintain it themselves. Additional benefits such as enabling geographically distributed load helps performance engineers run more realistic tests.

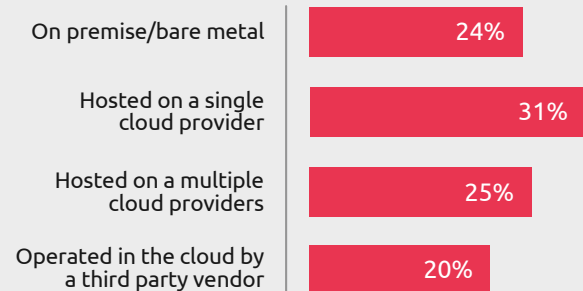
Our survey shows that the majority (74%) of the load testing infrastructure is operated in the cloud today and that this trend continues to increase as respondents see cloud testing accounting for 76% of the infrastructure in the next 12 months.

Performance engineering and the cloud isn't just about testing from the cloud. It is also about testing cloud based applications, whether they are native cloud

Today



In 12 months' time



Percentage question
Total respondents = 515

Question 14: How is your load testing infrastructure operated? Please provide the % breakdown for each option shown today and in 12 months' time?

applications, or systems migrating from on premise to the cloud.

Cloud computing is a disruptive technology innovation that promises on-demand computing power at scale, for lower costs. And it delivers! However, the complexity of cloud-based systems makes performance engineering even more necessary. Relying solely on elastic computing to scale in lieu of fully understanding the applications architecture can mask performance bottlenecks, raise costs, and defeat the purpose of cloud deployments.

There are several patterns of cloud migrations, from rehosting the application on an infrastructure as a service platform (IaaS) up to rebuilding a new approach with considerable changes to the application. The performance engineering strategy is worth adapting to suit the migration pattern to help ensure the migrated system will withstand the anticipated use.

When an application is rehosted, focus the load testing on the overall environment to answer questions such as: Do you have enough network bandwidth? Are your VMs provisioned with enough CPUs, memory, and appropriate disk to support the load?

When a server-based architecture is being replaced with a serverless approach, load testing needs to focus on application performance at access points rather than on the internal organs of the infrastructure. You have control over the code and datacentre regions. You do not have control over CPU or memory allocation, which is done by the service provider; hence, the focus on application behaviour.

Performance engineers are well aware of the challenges related to cloud systems:

- Troubleshooting across the delivery chain is complicated. Identifying which layer contains the actual issue may be complex. Using advanced APM solutions to monitor a cloud-based system during a load test provides deeper visibility into where problems exist
- Tests reproduce-ability is difficult to achieve. When running a load test it may be difficult to know what resources are actually allocated and the contention is
- Serverless does not mean optimal performance. Serverless functions become more prevalent in cloud computing. But poor performing code is wrong regardless of whether it's running in an application, on a virtual machine, or in a serverless function. Understanding the application's infrastructure and behaviour is critical to identifying performance bottlenecks.

There's a lot of benefit to migrating to the cloud. Companies can optimise their infrastructure costs and increase the volume and quality of service that they provide to their users. Ensuring comprehensive, state-of-the-art performance testing is part of the migration process.

Real-life example

A long-time leader in the Internet-of-Things (IoT) market sold the first connected automatic wrist detector for frail people to one of their corporate customers. This detector would communicate with a routing device, which would then communicate with a cloud-based server.

The company did not know what the performance requirements should be, but had a strong perception of what could be acceptable. They made all application components and tools available on a pre-production instance deployed on a small local cloud-based host. An external team assisted with the performance engineering aspects.

The team used injectors to generate the various types of load tests on a staging environment. In the absence of physical wrist detectors and routing devices, the performance testers used websocket scripts to simulate their traffic with the backend servers.

With the insights the team learned through testing the application components and tools they were able to provide configuration recommendations to the company and validate the throughput. This meant the company could launch the product with confidence.

Outlook on Performance Engineering



The next twelve months may well cross several major transitions in the economy. The turbulence of the impact of COVID-19 is already being felt throughout just about every business, organisation and team. Engineering, business leaders and teams will all need to adapt and ideally find ways to help their teams and businesses thrive.

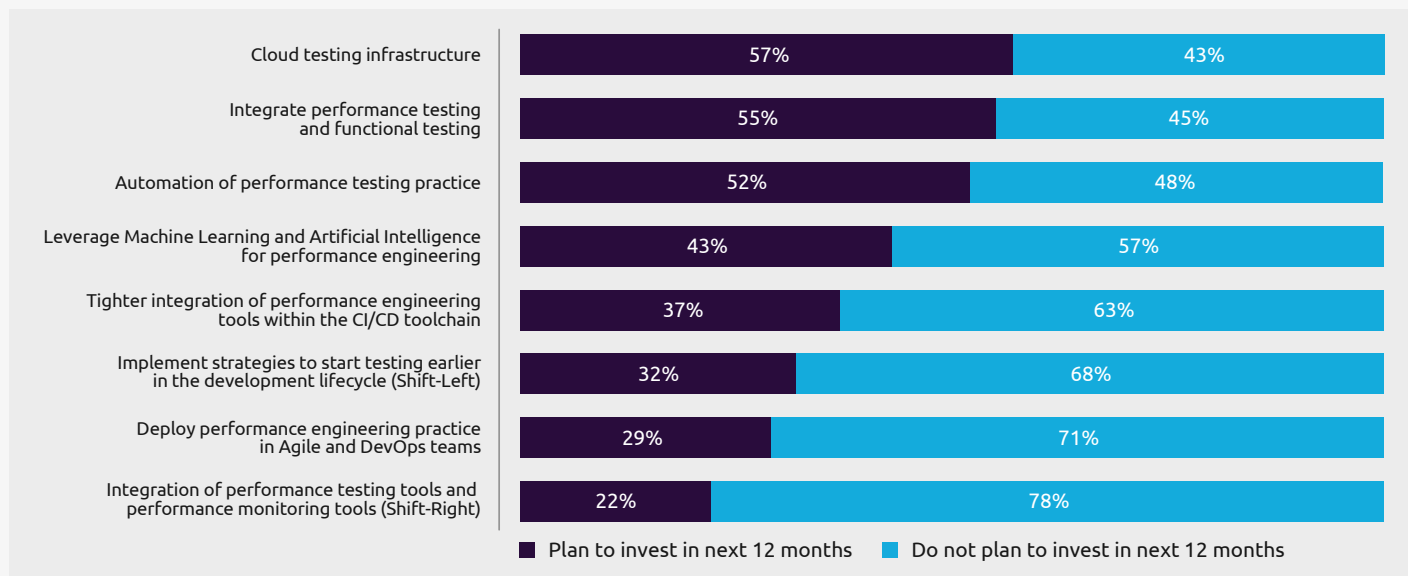
Chapter 5: Outlook on performance engineering

Some of the trends that started a few months or years ago will be accelerated. For instance, enabling people to work effectively from home, and in terms of migration to the cloud. We asked our participants for their perspective on the next twelve months in terms of their plans and vision for this period.

At least 50% plan to invest in three key areas: cloud

testing infrastructure, the integration of performance and functional testing, and automation of the performance testing practice.

Some teams still need to address basic challenges, such as improving their functional testing practice for those teams they may not yet be ready to focus on performance testing. Nonetheless, we see organisations accelerating both their migration to the cloud and also of their digital migration. They want to complete the transformation rather than being saddled with disparate responsibilities and practices. There seem to be two connected demands for the engineering teams. They



Question 15: Do you plan to invest in any of these in the next 12 months?

need to fit performance testing within short Agile sprints and integrate performance testing into their build and DevOps pipelines. These demands mean the teams are at risk of being seen as the bottleneck for businesses, which is not a great situation to be in.

As part of the migration to testing in the cloud, teams need to recognise and understand the many ways their projects can go awry. For instance, costs for cloud services can exceed 10x or even 100x what was expected, while limitations in the configuration may mean the tests are not representative or trustworthy. Bad data may be worse than no data in terms of test results as consumers of the data may not know enough to recognise that the data is bad.

Based on our experience and the survey interviews we believe performance testing needs to fit within the new timescales (of hours and days) and practices (continuous builds and deployments). Similarly, the people doing the work cannot be isolated from the rest of the teams. 37% of respondents plan to invest in tighter integration of performance testing tools, and 29% of the respondents will be investing in deploying performance engineering practices in Agile and DevOps teams.

Are your tools still fit for purpose?

Teams and organisations may need to reconsider which software tools will suit their needs given today's and future demands. These demands include: software running in the cloud, integration of performance and functional testing, automation of the performance testing practices, and deployment of the practices in Agile and DevOps teams.

Historically, performance testing was on-premises and often used tools that were locked to individual users or machines. These tools will become increasingly inappropriate to suit the needs of the work. Also, automation of performance testing extends well beyond the test design and execution, including automated test data and test environment management, go and no-go analysis, and integration with pipelines and monitoring.

In terms of functional and performance test integration, software such as mabl²⁶ can help streamline performance regression testing as part of automated CI/CD pipelines. Here, the performance testing is more likely to be small scale and immediate rather than larger volume and extended duration testing.

In today's context of organisations going Agile and adopting DevOps, the requirements of performance engineering tools are shifting along three main directions:

- As performance becomes everyone's responsibility, tools need to be suitable for whoever is involved in performance engineering. Developers are encouraged to implement 'test as code' at the API or microservice level early in the development cycle. QA teams can use low code testing tools even for complex end-to-end performance tests
- Modern tools need to be collaborative and support a global approach of performance engineering. Teams should be able to share test assets, results, and infrastructure
- Automation is key to accelerate delivery. With modern tooling, automation can happen at all stages of the

performance testing process. Some parts of test design and maintenance can be automated today and we expect that these tasks will be more automated in the future. Test runtime can be fully automated with CI/CD integration. Some aspects of test analysis and go /no-go decisions are automated today. Artificial Intelligence (AI) and Machine Learning (ML) should provide greater automation capabilities to help engineers embed performance testing into their software delivery chain.

Currently, pre and post production don't necessarily see eye-to-eye. They may be run and managed by different teams. The technology promises a lot but there's a lot of education, new processes, and good practices that need to be established.

We cannot educate people with just slides and theory. They need to hear from practitioners who have gained relevant experience and survived. Top-down support and leadership helps facilitate learning and education. Participants also need to identify what they are responsible for and what the key performance indicators will be in terms of their work in performance engineering, etc.

What limits your team's abilities to improve practices?

In our collective experience, many teams are still in the early to mid stages of the various activities, where performance testing is mainly standalone and manual rather than integrated and fully-automated. Automation is on the way but not yet mainstream. There is also a real risk that responsibility for systems performance

is being diluted and dissipated as more and more non-specialists are expected to do the work (but don't know what's involved). How much can teams trust the results of the work of their colleagues and peers in terms of performance testing?

Many teams are also in transition where the teams and team members are still learning how to work remotely, from home. Accelerating their abilities to work in a hybrid model with distributed working practices can also enable the team's abilities to improve their methods and their results.

One approach that has worked well at some companies is where a small team champions innovative and strategic experiments. They can be relatively compact and self-contained, able to decide and evaluate quickly, and where they maximise learning. Their role is to be pathfinders. They may discover many unsuitable routes, which can be evaluated and discarded quickly. Once they have found a productive path, they can then help the rest of the organisation to use it. In summary: they can see what's possible in terms of experiments that demonstrate positive improvements to the performance testing practices.

Being effective during a crisis

Innovative and adaptive companies have accelerated various plans and some are transforming their business. For instance, many companies have had to establish a streamlined and optimised online digital channel to serve their customers. Excellent performance of these channels is vital for these businesses so they are learning to actively engage with performance testing and improving

their practices. Other companies, including a major Asian airline, looked at the return on investment if they could speed up their testing. They realised the return was very strong and, while many of their aircraft were grounded, meant they were able to accelerate the work with little risk of adversely affecting their core business. They are now far better positioned to grow than competitors who sat tight and suspended their transformation projects.

Healthcare organisation launches new COVID-19 emergency funding site in a week

In a record time of just one week, a major healthcare organisation developed a new website to distribute emergency funding to healthcare providers during the COVID-19 crisis.

This aggressive timeframe was achieved through continuous testing of the constant code changes during the development phase.

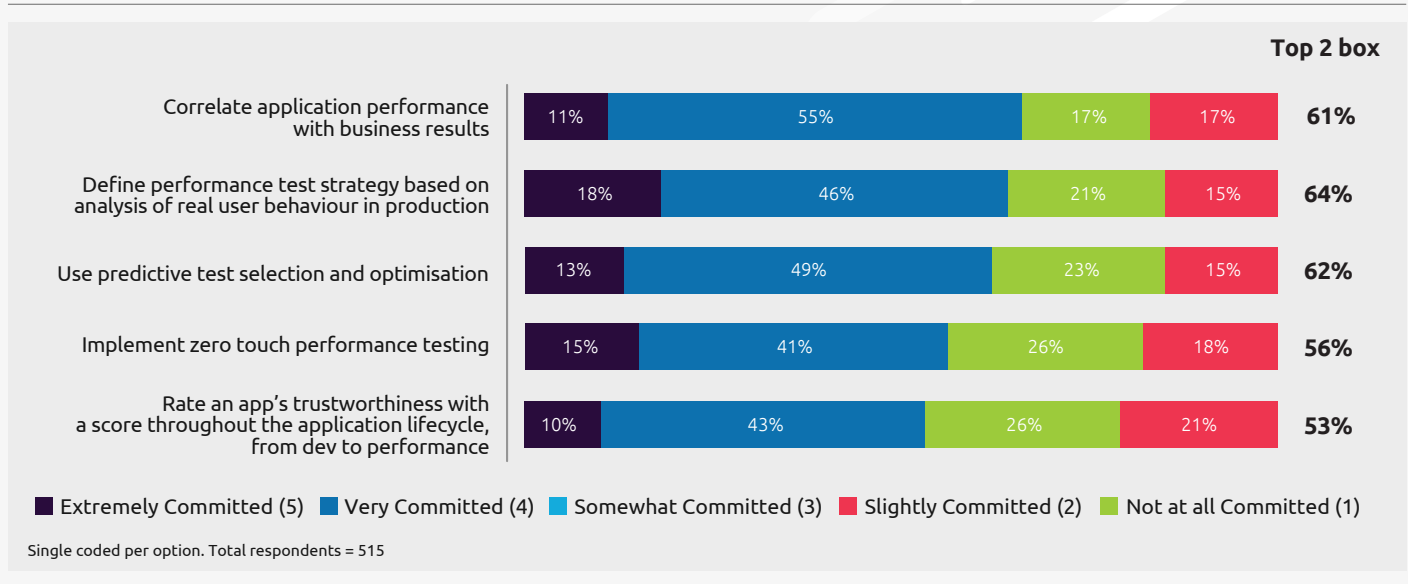
In order to ensure the scalability of the new website under realistic conditions of geographically dispersed locations, the organisation simulated virtual users with 35 load generators distributed over 12 different locations across the United States.

Although the expected traffic on the new website was 58 transactions per second, the performance engineering team was asked to validate the scalability at 12 times this capacity, and did so. They

undertook performance tests, including API tests, integration tests and end-to-end tests for the full application. These tests were executed throughout the software development lifecycle enabling the team to identify performance bottlenecks early and fix them rapidly.

Millions of people have been confined to their homes during the COVID-19 pandemic, and some have lost most of their usual incomes. Yet, they still need banking, insurance, and utilities such as electricity. They also rely on network connectivity and streaming services which in many ways are as vital as the more traditional suppliers here. For these businesses, the quality of digital relationships is paramount. The brand experience they are building is strategic because otherwise people will swap to a competing service as soon as they can, and once they leave these businesses they are unlikely to return.

Companies will increasingly expect their network and cloud providers to take care of performance and reliability of the services they use. While the cloud infrastructure providers will provide a certain level of support, they do not know a business's specific requirements or performance priorities. We are likely to see a mix-and-match approach to cloud and connectivity services, and to the tools used where teams will use bolt on customisation of various product offerings to give them what they want. Performance testing will need to adapt accordingly, to be able to test combinations and assemblies of systems and tools quickly and effectively.



Question 16: Using a scale of 1, not at all committed to 5, extremely committed, how committed are you to address and implement the following advanced use cases in the 12 months?

In relation to Question 16, the first topic of correlating application performance with business results is one we have covered earlier in this report and it is encouraging that companies are committed to continue investing in it in the next 12 months. There is plenty of scope for improvement in this area and tools that incorporate Artificial Intelligence may help in this area, for instance by identifying key metrics from production that correlate to business results.

Predictive test selection and optimisation relies on several foundations being in place first. These include

identification and analysis of real user behaviour in production, the core transactions and user journeys. It needs data from production and also in terms of previous testing to determine which tests to prioritise. The next challenge is where and how to run these tests. The more the entire end-to-end testing process is automated, instrumented, and malleable, the greater the potential to optimise and tune the testing. This leads to the concept of zero-touch performance testing.

Zero-touch aims for an ideal where the testing removes the need for humans in the loop, which is a vast

difference from how performance testing is generally done currently. An intermediate step is to find ways to reduce the frequency and duration of manual nitty gritty work. This work is often needed to revise test scripts by hand and through trial and error. Zero-touch performance testing is likely to emerge first for testing APIs and machine-to-machine applications where they don't need to deal with the many and various human interfaces.

During discussions with experts in the field, one of the key observations is the gulf between the willingness for companies to consider these advances in performance testing and their current realities. Education is vital to help companies transform their commitment to applying these concepts into concrete understanding of what's involved in making them practical and fruitful.

The tools and the companies are still in the research and proof-of-concept stages for some of the more advanced topics such as zero-touch and predictive test selection and optimisation. Often the basic prerequisites need to be fulfilled first in order to provide a launchpad for actually integrating these advances into their practices. Therefore, the trends and greatest potential for success is for companies to focus on executing the basics: shift-left, improving communication, and dismantling silos of teams and data. We will cover data shortly after considering how AI can help improve performance engineering.

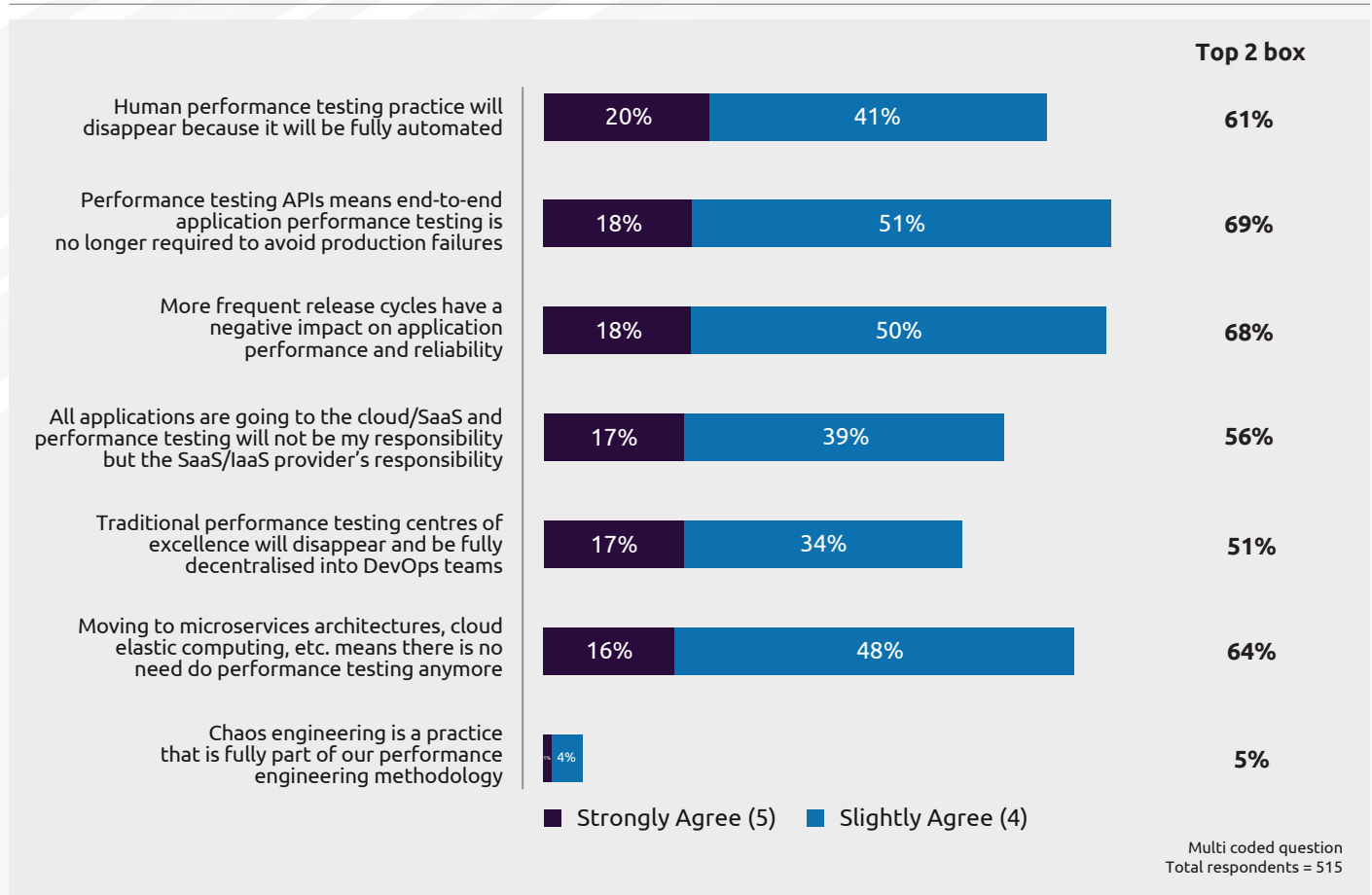
AI has the potential to improve many aspects of performance engineering both individually and collectively. It can augment many tasks, such as data correlation, pinpointing correlations between

performance bottlenecks and system usage, etc. For many tasks, the quality and utility of AI will depend on the data it has access to. For various reasons the data may be hard to obtain. For instance, the data may be siloed, or it may be secured as part of meeting legal and contractual obligations.

Medical patient data is a good example of data sensitivity. Similarly, businesses would need to trust the integrity and security of the tool before giving it access to financial data, such as profit margins per sale or per customer. Tool vendors have a responsibility demonstrate and enable their customers to validate the trustworthiness of their software. Various techniques are emerging in research to help protect the underlying data while still providing enough meta-information to help guide AI appropriately. Ethics of AI is a related topic that both vendors and users of these tools will need to consider and address appropriately.

Data in all its many forms has often been stored independently, whether test results, operational data and metrics, system logs, and of course the core business data. AI and Machine Learning (ML) both thrive on data and can learn and provide better decisions when they're able to access as much relevant data as practical. For many good reasons much of the data related to customers is protected and no longer free to use for performance testing. Techniques such as [differential privacy](#) can help protect sensitive data. Performance tools that can guarantee the privacy of the source data may become increasingly relevant in the next year or so.

These questions were intended to challenge the participants to draw out their perspectives on various



Question 17: How strongly you agree or disagree with each of the following statements, using a scale of 1, disagree strongly to 5 strongly agree?

strategic challenges. On the first topic, whether human performance testing practices will be superseded by fully automated alternatives, our experts drilled into the topic. We are a long way removed from this happening, yet the role and what people spend time on will change massively, with automation will replacing some of the current activities.

One aim is to reduce manual work by spending much less time on:

- Writing and mending scripts
- Creating and cleaning up environments
- Correlating variables.

Instead, they can focus on strategic aspects of performance engineering and help other stakeholders understand the complexities so they can make better decisions. They can also help developers find ways to improve the observability of their critical software components so they become integrated into a healthy living ecosystem.

One of the interesting dilemmas is how the role will change and what work will be performed by bots rather than people. We expect there to be plenty of demand for competent engineers who can orchestrate the testing and tuning of hybrid testing that uses bots. Another area where there is plenty of good work to do is in helping with the next topic.

68% of respondents are concerned that the increasing frequency of releases leads to a negative impact on application performance and reliability. We do not know if this is a problem of perception, how the teams work, or other causes, nonetheless there is clearly work to do to address their concerns. We recommend that performance testing needs to improve at least as quickly as any transition to Agile software development. Several key tenets of successful Agile development increase when performance testing is also integrated in the development and release practices. A good objective is to deliver pertinent performance information to developers so that they can improve the performance within the release lifecycle.

51% believe centres of excellence will disappear.

Certainly, it's hard to go faster if performance testing is highly centralised. However, there is still a place and a role for a core team of people who focus on the depth and breadth of performance engineering.

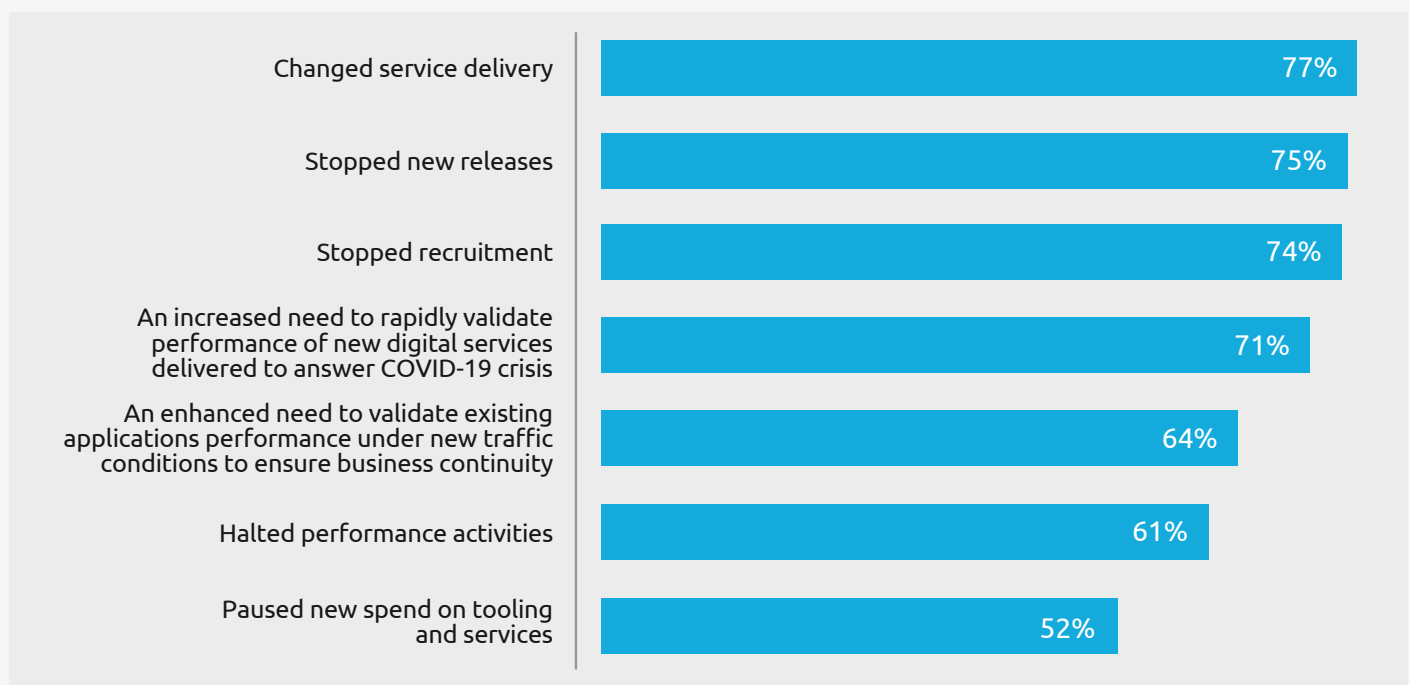
There seems to be a misconception that moving to microservices and cloud elastic computing will mean there is no need to do performance testing anymore. As one of our experts said: "These are both great, however have you checked your bill at the end of the month?". We are moving from monolithic systems to distributed, loosely coupled architectures often provided by a mix of companies — performance engineering is needed more than ever. Indeed one of the challenges is learning how to address performance when you cannot test everything,

for instance when using third-party services. The future of performance testing and performance engineering will change from traditional scripting and analysis to understanding networks, cloud architectures and AI.

You might ask how to test AI. We will find ways to do so. A few years ago, mobile seemed similarly alien and is now well understood in the industry. However, there are

critical differences between testing mobile and AI, for instance the ability to determine whether AI is correct and unbiased is much more nuanced and complex. The answers are not boolean.

Perhaps one of the biggest challenges is in learning and education to meet the demands of today's and tomorrow's performance engineering. There are few



Question 18: How strongly you agree or disagree with each of the following statements, using a scale of 1, disagree strongly to 5, strongly agree?

university courses available and a paucity of commercial or practical training beyond learning specific software tools. Industry practice and opensource projects such as chaos engineering from Netflix seem to be leading the way.

As we can see, respondents noticed wide ranging effects of the COVID-19 pandemic and crisis. The majority stopped releases, recruitment, performance activities and spending externally. And yet there's been a tremendous need to adapt quickly and effectively to validate changes to service delivery and evaluate the performance of new digital services. In the experience of our group of experts, these figures mask sector by sector differences,

Enabling other businesses to thrive

A pizza company had to close all their stores in London and Oxford in March 2020 and went from making 30,000 pizzas a week to zero. The founder wanted to support their customers and ended up creating pizza kits containing all the raw ingredients needed for people to make pizzas at home. They used Shopify to launch the service and have sold thousands of pizza kits per month.

They are one of many examples of businesses who were able to successfully pivot their business because the underlying online store was able to cope with the performance needed to support these e-commerce customers. Effective performance engineering is enabling others to survive and thrive.

where some sectors including hotels and airlines have stopped for the most part (apart from isolated examples such as the one we mentioned earlier in this report). Other sectors including healthcare, delivery services and telecoms have accelerated their frequency of software releases — presumably to cope with the increased demand for their products and services.

As many of us know first-hand, how and where we work has changed materially since the pandemic struck, with many of us working from home and communicating online over the internet to get our work done. Indeed, this entire report was created remotely without meeting in person. Performance of the communications and collaboration tool is becoming increasingly vital to organisations (and participants), bringing new demands and additional constraints. Part of our work may be to help ascertain the performance of these tools, the conduits and, at least as importantly, finding ways to help support new team distributions and working practices.



The Practitioners' Corner

Closing thoughts: the practitioners' corner

These contributions are provided by two participants in [The Performance Advisory Council](#), a Neotys initiative. The aim is to bring together experts from around the world to share their experience and knowledge regarding techniques, challenges, and best practices associated with several topics on the minds of today's performance engineers. These topics include DevOps, Shift Left/Right, Test Automation, Blockchain, and Artificial Intelligence.

Five trends of performance engineering

by Alexander Podelko

Most existing performance engineering trends are interconnected, so any specific list may be challenged. All mirror maturing IT macro-trends in general and software development in particular as performance, efficiency, scalability, reliability and resilience need to be built into processes. So here is my list:

1. Adjusting to Agile and CI/CD

A top trend is the increasing move toward continuous performance engineering. Mostly we are talking about continuous performance testing — any kind of tests measuring performance, even just unit test timing — to make sure that we don't introduce any performance issues. But it also means all the other performance engineering techniques needed to make continuous performance engineering successful (monitoring, anomaly detection, visualisation, root cause analysis, etc.).

2. Integrating performance engineering into DevOps

The two trends most often referred to — shift-left and shift-right — are actually just integrating performance engineering into the DevOps process. Traditional performance testing sat between development and operations, but this provides very limited value in DevOps. So now performance engineering is moving to be fully integrated into DevOps — with shift-left bringing performance engineering into Dev, and shift-right bringing performance engineering into Ops. We have always had a large degree of performance engineering in Ops, but now it is being merged with performance engineering in Dev, including performance testing.

3. Context-driven performance engineering

We've seen different performance engineering tools and techniques — such as APM²⁷, log analysis, tracing, observability, AIOps²⁸, user testing (canary, A/B, etc.), chaos engineering — advance quickly to give us more ways to mitigate performance risks. While none is actually revolutionary (practically everything has been around for a while, maybe under other names), recent advances of each option have taken performance engineering to a new level. They overlap somewhat in what risks they mitigate, and differ in their efficiency depending on context. Performance engineering strategy becomes increasingly non-trivial. As more options become available, an optimal combination of methods and techniques to be used (including their timing and extent) should be determined depending on the specific context.

4. Integrating everything: tools, processes, roles

For tools, we can see things going two ways. One, more clearly observed in open source software, is that many highly specialised tools are cobbled together in a 'framework' to provide full performance engineering functionality. That includes performance engineering tools as well as other kinds of products providing needed

functionality — such as data series databases and visualisation tools. The second, more clearly observed in successful vendors, is extending the functionality of the performance engineering tool (whatever its purpose was) in all directions and to try to provide an integrated solution. At the moment there is no agreed-upon name for these integrated tools: some names used currently are AIOps (Artificial Intelligence for IT Operations, Algorithmic IT Operations), ITOA (IT Operations Analytics), ITOM (IT Operations Management), DEM (Digital Experience Management/Monitoring), observability, etc. In a way, it is the same old dilemma — 'best-in-breed' vs. integrated solution — but much more focused on integration.

5. Chaos engineering: renaissance of reliability

Reliability testing is now becoming popular under another name: chaos engineering. It originated in the DevOps world, probably because it was practically ignored in testing. But it's now making its way back to testing. Chaos engineering is potentially developing into a new discipline: resilience engineering. It isn't well defined yet, although it has well established architectural patterns on one side, and testing practices — chaos engineering — on the other side.

Four complementary trends in 2020-21 that are here to stay

by Twan Koot

As technologies rapidly evolve, so does the way we develop software. For instance, the cloud and CI/CD are now standard in software development and have had a foundational impact on the very way we do performance testing. But it's not only technology that's driving a new way of working. End users are increasingly more demanding and expect 100% uptime and super-fast application response times. To meet the higher demand for performance to ensure we can detect, prevent and resolve issues quickly, we are testing earlier, testing more often, and are testing more in production. How these trends have changed the way we work, and how they impact us as performance engineers/testers, are outlined below.

1. Shift-left

Organisations are steadily shifting to an Agile way of working — or even adopting a total DevOps approach. These new approaches to developing software bring a greater adoption of CI/CD. As can be expected, due to the nature of CI/CD, performance engineers and testers have increasingly adopted a shift-left mentality. Continuous testing reports and quality reports are an example of shift-left. As organisations reach a state of shift-left, the next step naturally will be to shift-left even further. So, in 2020-21 we may

expect (and are already seeing) more and more tool vendors introducing features for faster and earlier testing capabilities.

With the introduction of performance testing at the unit testing level, performance engineers and development teams can test earlier and detect performance degradations. Observing and increasing performance at a granular, component level allows for efficient code, which can lead to more robust code with performance in mind. This trend is also gaining ground in organisations that leverage the capabilities of serverless architecture. Performance engineers need to adopt a new approach to test this type of architecture; therefore, I foresee an increased adoption of even further shift-left performance testing. In order to be able to help developers at this code level, we performance engineers need to understand software development and software architecture. Getting the skills for this type work should already be in the performance engineering mind-set and is becoming more and more important.

2. Everything as code

The term 'everything as code' came into being with the rise of the DevOps approach to software development. Cloud providers developed ways to enable infrastructure as code and CI solutions opened up more and more ways to code pipelines, the next step is performance testing as code. We've already seen some open source

solutions offering a code-only approach. Using an 'everything as code' approach makes the overall development process easier to automate, and code can be stored in a source code repository (e.g. using git) to enable easier collaboration and transparency. We're now seeing performance testing tools increasingly supporting an 'everything as code' approach. This 'new' way of creating and maintaining performance test scripts is again an enabler for more automation, which completely supports earlier and more frequent performance testing.

3. Dynamic test infrastructure

Performance testing is becoming more and more common throughout the development process, with more tests being executed for different types of performance testing demands. With the increase of automation within performance testing, teams can performance test every commit or release. This demands a different approach in not only testing strategies but also performance testing infrastructure. It requires a performance testing solution that is capable of testing multiple performance tests at the same time, to support all teams.

Technologies such as Docker and Kubernetes are far from new to the development world. But they help create the type of performance testing architecture that allows for a huge amount of concurrent performance tests

across the organisation. With tool vendors supporting these new technologies, it's up to us as performance engineers to get up to speed on these technologies and start leveraging them. The adoption of cloud combined with the new technologies will allow performance engineers to create testing infrastructure that can scale up and down to enable optimal usage of resources. This new performance test infrastructure is needed to keep pace with the velocity of testing done by all the development teams and ensure that capacity can either scale to meet demand.

4. Shift-right

With the introduction of DevOps, teams now have the responsibility to ensure that the production environment is running smoothly and without issues. This increases the need for monitoring tools to detect and spot issues. As performance engineers/testers, we are used to working with monitoring tools to gather information about infrastructure, etc. We all know the amount of data we can gather and the effort that goes into analysing data. As more performance tests are executed across the organisation, an automated way of analysing these results is needed.

APM tools and their AI capabilities (which really took off last year and continue to improve) are increasingly being incorporated into performance testing in a CI/CD environment. I expect that using APM or other monitoring tools will be extended

to test dev environments. With AI becoming more present in the monitoring tools, they offer a great way to automatically detect issues and reduce the amount of work needed in analysing all the data.

When looking at current performance engineering trends, we see a lot of things happening in the market that influence the way we work. With organisations migrating to the cloud and adopting DevOps as the new standard to deliver software, we performance engineers need to broaden and adapt our skillset. With more technologies enriching tools' feature sets, we can test in more new ways than ever before. All of the trends we're seeing in 2020-21 fall under the umbrella of the larger overall trend toward enabling a DevOps approach to performance engineering:

- Even more shift-left performance testing and testing at the code level
- Taking the performance testing infrastructure to the cloud and making it capable of scaling up and down by using technologies like Kubernetes
- Applying everything as code to performance test tooling to pave the way for easier and more efficient automation of performance testing
- Enabling monitoring tools like APM in test and dev environments to enable better and faster analysis of performance test results
- Embedding performance as standard in all steps within software development: design, development and production.

As an organisation, team, or individual performance engineer, getting up to speed with these trends is more relevant than ever. In my opinion these trends take performance testing to a new level and are here to stay.

Acknowledgements

Neotys and Sogeti would like to thank the contributors for their vision and expertise in creating this report. We would also like to thank the 515 IT decision makers who took part in this study for their time and contribution. Here are the people who contributed and made this report possible.

NB: Ensure you are logged in to your LinkedIn account before viewing the following profiles.

Writers

Antoine Aymer, Sogeti
www.linkedin.com/in/aymer/

Sylvain Fambon, Neotys
www.linkedin.com/in/sylvain-fambon/

Julian Harty
www.linkedin.com/in/julian-harty-5a010413/

Contributors

Paul Bruce, Neotys
www.linkedin.com/in/paulsbruce/

Stephane Brunet, Neotys
www.linkedin.com/in/stephanebrunet9/

Michel Burini, Sogeti
www.linkedin.com/in/michel-burini-39296a2/

Thibaud Bussière, Neotys
www.linkedin.com/in/thibaud-bussi%C3%A8re-a748834/

Vivien Delahaie, IMA Group
www.linkedin.com/in/vivien-delahaie-1a35912b/

Bruno Duval, Neotys
www.linkedin.com/in/brunoduvalatneotys/

Stian Erdal-Arvness, Sogeti
www.linkedin.com/in/stian-arvnes-36a8227a/

Andrew Fullen, Sogeti
www.linkedin.com/in/andrew-fullen-9781171/

Laurent Gaudy, Neotys
www.linkedin.com/in/lgaudy/

Thierry Gros, MAIF
www.linkedin.com/in/thierry-gros-b783b2125/

Benoit Heib, Sogeti
www.linkedin.com/in/bheib/

Maheshwar Kanitkar, Sogeti
www.linkedin.com/in/mrkanitkar/

Twan Koot, Sogeti
www.linkedin.com/in/twan-koot-a813a8b7/

Hemant Lal, Caggemini
www.linkedin.com/in/hemant-lal-68a7667a/

Deepika Mamnani, Caggemini
www.linkedin.com/in/deepika-mamnani-2205943/

Aurélien Morel, Sogeti
www.linkedin.com/in/aur%C3%A9lie-morel-b532712/

Rajesh Natarajan, Sogeti
www.linkedin.com/in/rajesh9/

Alexander Podelko, Oracle
www.linkedin.com/in/alexanderpodelko/

Inmaculada Ramirez-Perez, Sogeti

www.linkedin.com/in/inmaculada-ramirez-techdir-sogeti/

Gautam Reddy, Sogeti

<https://www.linkedin.com/in/gautam-reddy-92481140/>

Henrik Rexed, Neotys

www.linkedin.com/in/henrik-rexed-a85a8315/

Olivier Rundstadle, Sogeti

www.linkedin.com/in/olivier-rundstadler-9524811/

Siddhant Runiwal, Sogeti

www.linkedin.com/in/siddhant125/

Marcus Seyfert, Sogeti

www.linkedin.com/in/marcus-seyfert-227261aa/

Johan Strömberg, Sogeti

www.linkedin.com/in/johan-str%C3%B6mberg-b745b045/

Gopalkrishnan Yadav, Capgemini

www.linkedin.com/in/gopalkrishnan-yadav-14b7882b/

Reviewers

Marco Venzelaar, Sogeti

www.linkedin.com/in/venzelaar/

Ngaire McKeown

Designers

Ed Tilsley, Sogeti

www.linkedin.com/in/ed-tilsley-0372771/

Joanna Jozwiak, Sogeti

www.linkedin.com/in/joanna-józwiak-design/

About the study

The figures in this report are taken from the full study available at www.sogeti.com/explore/reports/state-of-performance-engineering/

Data in this report is from a detailed survey conducted by Coleman Parkes Research in May and June 2020, with 515 senior decision makers taking part. Survey respondents are from fourteen countries in North America, Europe, and Asia-Pacific. Only IT decision makers from large organisations with more than 1,000 employees were selected to take part.

The survey was conducted online with some telephone interviews where required and preferred by respondents.

The interviews were based on a questionnaire of 29 questions. Quality measures were put in place to ensure that the questionnaire was understood, answered accurately, and completed in a timely manner by the interviewees.

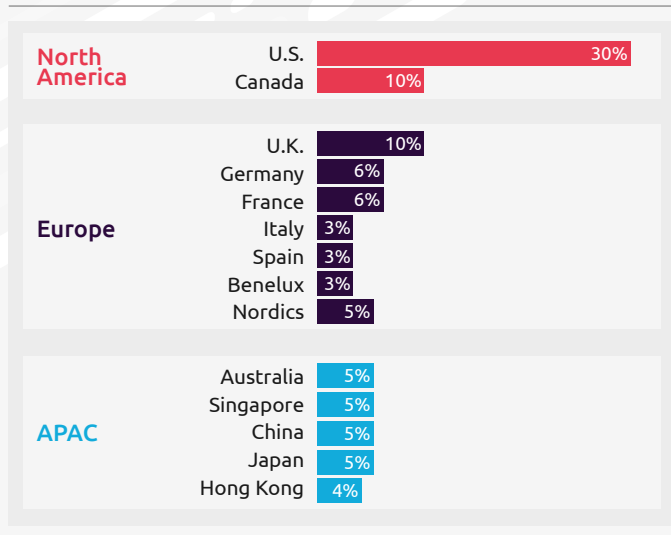
The database of respondents covers a variety of regions, personas and industries to ensure statistical representativeness in the context of the B2B market under study.

The survey questionnaire was devised by performance engineering experts at Neotys and Sogeti, in consultation with Coleman Parkes Research.

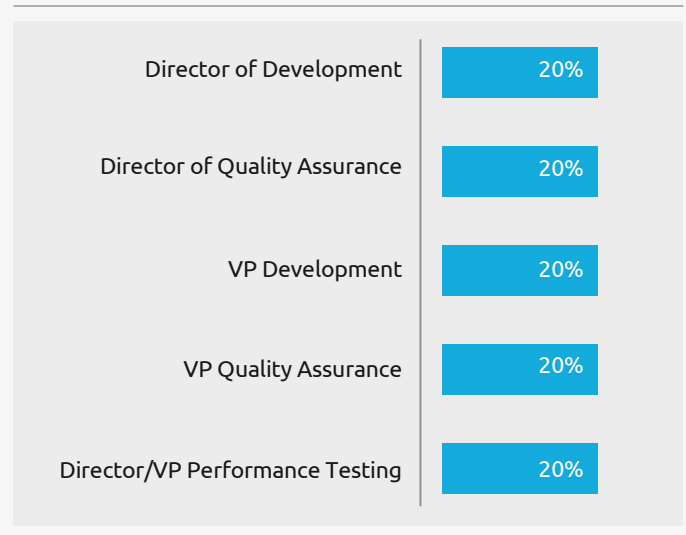
Respondents

Audience profile

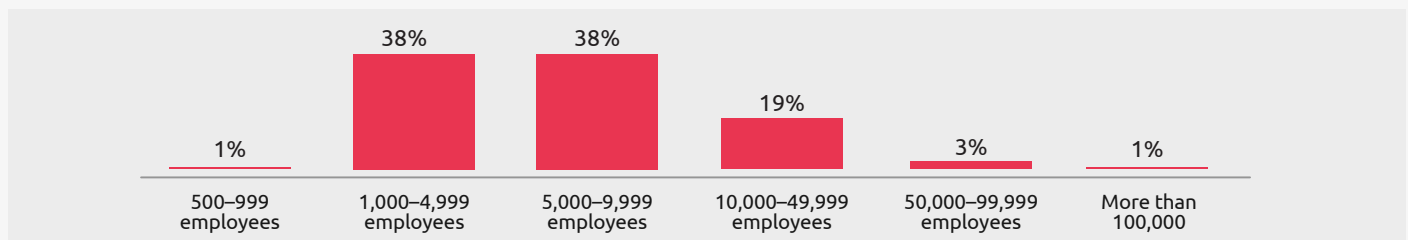
Functionaries with a strategic focus on the state of performance testing within the company



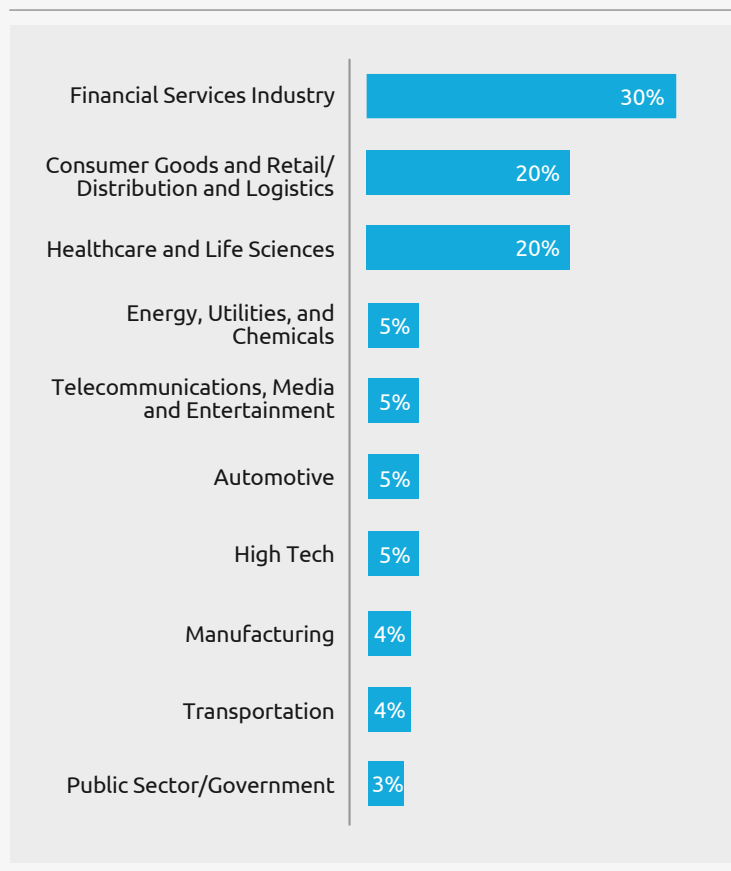
Country



Job title



Number of employees



Segments

Footnotes and references

1. www.torbenrick.eu/blog/performance-management/not-paying-attention-to-culture-undermines-performance/
2. www.sites.google.com/site/rtbforperformancemarketing/home/glossary/session-depth
3. www.forbes.com/sites/carolinebeaton/2016/10/10/how-to-find-direction-and-learn-your-professional-purpose/#4ee82c51686e
4. www.salesforce.org/pledge-1/
5. www.creativebloq.com/features/how-the-bbc-builds-websites-that-scale
6. www.ministryoftesting.com/dojo/lessons/a-quick-start-guide-to-learning-performance-testing
7. For instance, research presented at ICPE, the ACM/SPEC International Conference on Performance Engineering.
8. www.neotys.com/resources/whitepaper/practical-guide-to-performance-testing
9. www.techwell.com/software-conferences
10. California Consumer Privacy Act (CCPA), https://leginfo.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375
11. www.gdpr.eu
12. www.pcisecuritystandards.org
13. www.hhs.gov/hipaa/index.html
14. Paraphrasing: www.growthhackers.com/articles/marketing-is-too-important-to-be-left-to-the-marketing-department
15. https://en.wikipedia.org/wiki/Anna_Karenina_principle
16. www.browserstack.com/blog/chromium-based-edge/ provides a good description of what was involved and Microsoft's subsequent migration to Google's opensource rendering engine
17. Note: this mismatch can also occur with virtual machine instances in popular cloud services if teams choose inexpensive 'disk' for some of the servers
18. www.research.google/pubs/pub41590/ "Capturing Mobile Experience in the Wild: A Tale of Two Apps"

19. <https://entreprise.maif.fr/entreprise/notre-difference-assureur-militant/societaire-au-coeur-du-modele/satisfaction-de-nos-societaires-et-assures-une-raison-detre>
20. www.tmap.net/building-blocks/CICD-pipelines
21. www.zdnet.com/article/devops-and-observability-in-the-2020s/
22. www.ict-books.com/topics/quality-for-devops-teams-epub-info
23. www.iso25000.com/index.php/en/iso-25000-standards/iso-25010
24. www.tmap.net/building-blocks/Metrics-DevOps
25. www.tricentis.com/resources/forrester-research-on-devops-quality-metrics/
26. www.mabl.com/regression-testing/performance-regression-testing
27. Application Performance Management
28. Artificial Intelligence for IT Operations, or Algorithmic IT Operations

About Neotys

Founded in 2005, Neotys created its flagship product, NeoLoad. NeoLoad is a continuous performance testing platform for enterprise organisations who wish to standardise their performance engineering approach. Since its inception, NeoLoad has helped over 2000 enterprises throughout the world in retail, financial services, health care, insurance and more. COE and DevOps teams alike use NeoLoad to automate API and end-to-end performance testing in cloud and hybrid environments. NeoLoad helps these teams collaborate to release fast at scale while ensuring quality controls for application speed and stability.

Learn more about Neotys and NeoLoad at www.neotys.com

About Sogeti

Part of the Capgemini Group, Sogeti operates in more than 100 locations globally. Working closely with clients and partners to take full advantage of the opportunities of technology, Sogeti combines agility and speed of implementation to tailor innovative future-focused solutions in Digital Assurance and Testing, Cloud and Cybersecurity, all fuelled by AI and automation. With its hands-on 'value in the making' approach and passion for technology, Sogeti helps organisations implement their digital journeys at speed.

A global leader in consulting, digital transformation, technology and engineering services, Capgemini is at the forefront of innovation to address the entire breadth of clients' opportunities in the evolving world of cloud, digital and platforms. Building on its strong 50-year+ heritage and deep industry-specific expertise, Capgemini enables organisations to realize their business ambitions through an array of services from strategy to operations. Capgemini is driven by the conviction that the business value of technology comes from and through people. Today, it is a multicultural company of 270,000 team members in almost 50 countries. With Altran, the Group reported 2019 combined revenues of €17billion.

Visit us at www.sogeti.com





The information contained in this document is proprietary.
©2020 Sogeti. All rights reserved.

Sogeti Global Marketing Reference 00152.