

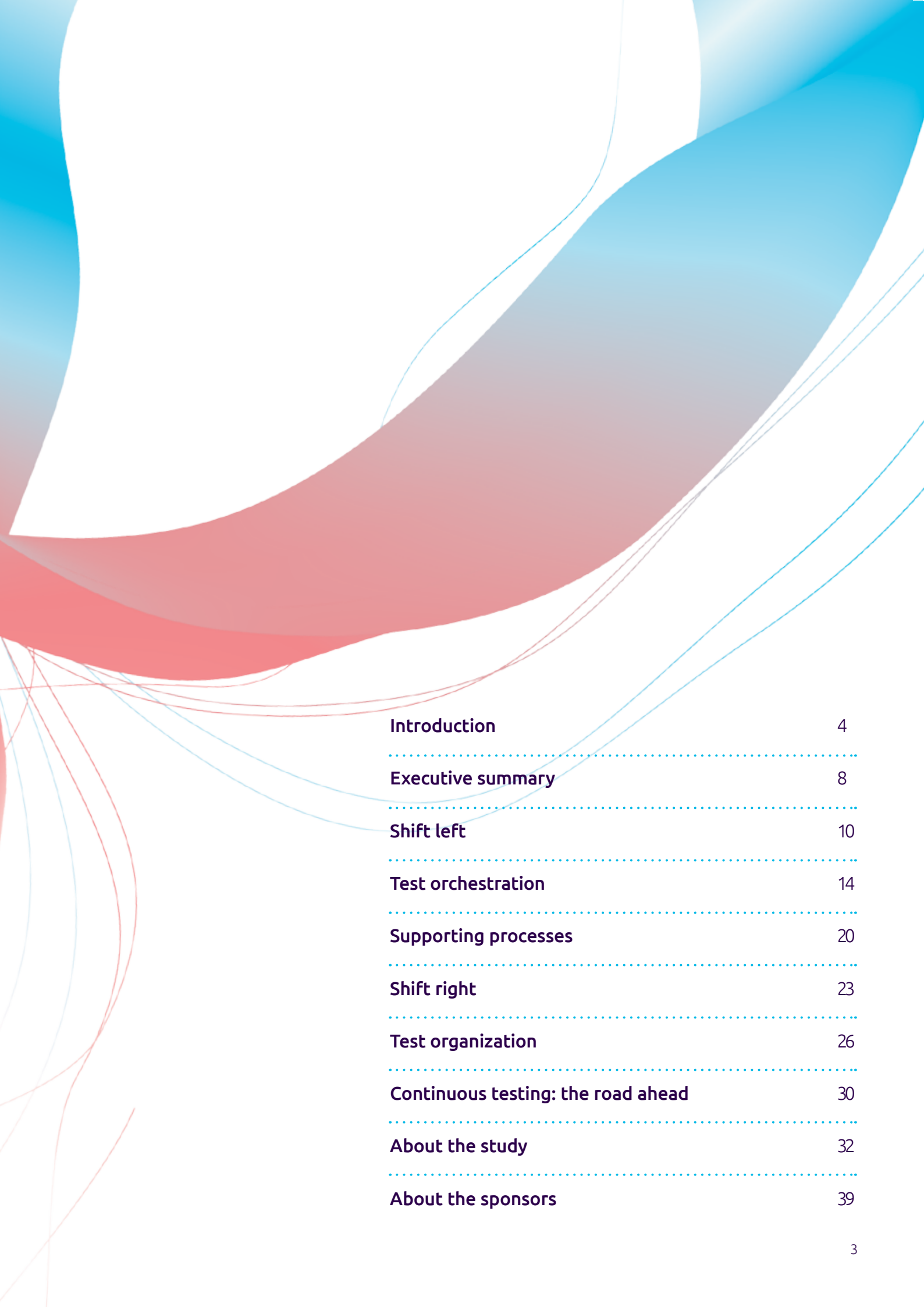


Continuous Testing Report 2020

In association with



Contents



Introduction	4
.....	
Executive summary	8
.....	
Shift left	10
.....	
Test orchestration	14
.....	
Supporting processes	20
.....	
Shift right	23
.....	
Test organization	26
.....	
Continuous testing: the road ahead	30
.....	
About the study	32
.....	
About the sponsors	39

Introduction





Mark Buenen

Global leader,
Digital Assurance and
Quality Engineering,
Cappgemini Group

Agile development methodologies and DevOps have become the standard for software development in all industries. Smart software solutions are pivotal for development, selling and delivering products and services.

The adoption of agile and DevOps methodologies is driven by the ever-growing demand from business leaders and customers for more, better and easy to use software solutions. Improvements and updates in these software product are required instantaneously. Speed and innovation are the buzzwords.

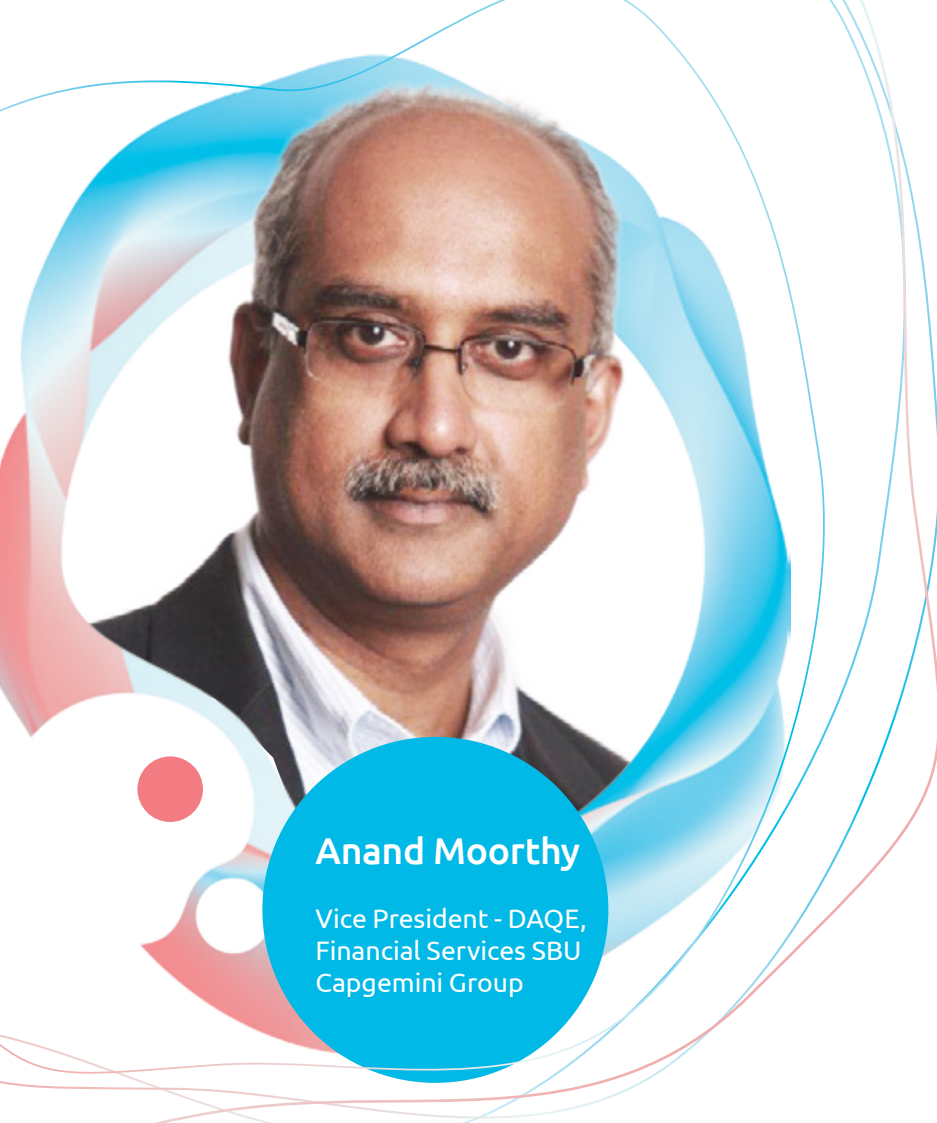
In order to meet the business demands for higher speed, IT organizations went through various transformations. From central organized IT teams to self-empowered teams. And from clearly staged software development phases to continuous delivery and continuous integration. In that transformation testing activities have also transformed from being an independent stage to a continuous process, owned and supported by all team members.

The Continuous Testing Report 2020 looks at how organizations are adapting from traditional testing to continuous testing, and what today's best practices are in this space. One clear conclusion of the report is

that achieving and implementing continuous testing is not without challenges.

The report also analyzes the status and best practices of the critical areas for continuous testing. These are shift left and shift right of testing, improving the level of in-sprint testing, orchestration and organization of quality assurance activities within and across the feature teams, test environment and test data provisioning, continuous quality monitoring in production, and the skilling and enablement of feature teams in QA and testing.

Based on survey data and complemented with hands-on examples, our experts provide in this report a concise overview of the current standards and best practices, and key recommendations that will help you to move ahead and overcome challenges. This report aims to be a guideline to help you to move ahead with continuous testing and in that way, fulfill the promise of agile and DevOps developments: a better and faster service for your business and client objectives.



Anand Moorthy

Vice President - DAQE,
Financial Services SBU
Capgemini Group

Dear readers, welcome to the second Continuous Testing Report.

Quality and testing approaches, methods, and expertise have undergone radical changes over the last few years. Every organization today aspires to deliver faster and more valuable IT solutions to business and customers. To do this, they have been leveraging agile and DevOps methodologies and using smarter automation technologies and as-a-Service solutions to deliver IT faster and with greater flexibility.

At the same time, the IT landscape has also been growing in complexity. There is an increased dependency on IT solutions today, with the integration of front-office and consumer-facing apps with back-office core systems, the leveraging of cloud and microservices and the integration and use of IoT. And, on top of that, AI is emerging to make these solutions autonomous and self learning.

All this technology is delivered by different teams, many of which may not even be part of a single company.

As we scramble to deliver innovative solutions for the newer, more complex IT landscape, there is, of course, a risk of failure. While some failures are inevitable and often provide a valuable learning opportunity (given a quick feedback loop), there are others that

we must prevent from happening. Failures in core systems that seriously disrupt the business operations of an enterprise, failures that seriously impact a large number of clients and therefore jeopardize an organization's reliability and brand perception, or failures in systems that cannot easily be rolled back all demand good testing of these systems before being deployed.

In reality, many organizations today are struggling to adapt their QA and testing processes to meet all these changes and needs. The core questions today are: how to achieve quality faster, how to do testing smarter, and what should actually be covered in a test?

The Continuous Testing Report 2020 gives you insight into what is state of the art today, a benchmark on quality approaches in Agile/DevOps, some use cases, and above all, some clear recommendations on what you can do to improve your QA and testing activities.

I wish you happy reading and many good insights that help you to achieve your continuous quality goals.



Sushil Kumar

Head of DevOps
and Continuous
Testing Business,
Broadcom

Welcome to the 2020 Continuous Testing Report from Capgemini, Sogeti and Broadcom.

In every vertical, and every company size, one thing is clear: the ability to deliver new functionality with speed and confidence is the key to competitive advantage. Failures in quality directly impact your brand and bottom line. And, visibility into the performance, value, and ROI of what you are building and delivering is vital to your organization's ongoing success.

This year, 40% of respondents said they were using business KPIs to judge the success of continuous testing. 43% say they are using production data. While other metrics indicate that these processes are largely still manual - we are definitely moving in the right direction to focus on the value and quality of what is delivered.

We have our sights set on doing more to ensure quality (e.g. security testing, using AI, sentiment analysis, chaos testing). The skills companies are looking for reflect those shifts. However, there was very little change year over year in how respondents ranked the challenges in adopting and improving continuous testing. Areas such as aligning requirements, achieving in sprint testing, and gaining visibility throughout the development lifecycle continue to be problematic.

What will move the needle? Continuous testing, like the Agile and DevOps practices it powers, requires an organizational and technology shift. It requires that everyone across the organization take responsibility for quality and have their focus on delivering business value. We can accomplish this with modern and easy-to-use tools that help democratize quality across all stakeholders - including developers, product managers and business analysts. These tools need to be flexible enough to allow teams to work the way they want to work, using technologies and interfaces they prefer. The use of data intelligence, artificial intelligence and machine learning also holds great promise to make testing more intelligent and automated, while dramatically reducing manual effort and allowing enterprises to balance innovation and risks. We can build collaboration around shared KPIs and actionable insights collected across systems, builds, and releases. With a focus on business value, we can eliminate complexity and empower our teams to shift left, and right, to move faster with confidence.

We are excited about 2020 and what this year will bring. We hope you enjoy the report.

CONTINUOUS TESTING REPORT 2020

Executive summary

This Continuous Testing Report (CTR) 2020 brings together survey data and the opinions of subject matter experts from Capgemini, Sogeti, and Broadcom to explore the ever-changing landscape of testing at every stage of the software development lifecycle.

This is the second year of our joint survey. To gauge the climate, we canvassed the opinions of 500 senior decision makers in large and enterprise-level organizations in North America and Europe. More than half (55%) of them have already adopted a continuous testing approach; the remainder plans to do so. The slow increase in maturity (compared to last year) demonstrates that the path to continuous testing is not an easy process. Teams must be ready to move from simply automating a checkbox functional test script to positioning quality and automation at the very heart of the development lifecycle.

Here are some of the key findings and inferences that we derive from the data:

Shift left

Many organizations are struggling to make shift left a reality. Designing and maintaining meaningful test cases that align with end-user expectations is challenging or extremely challenging for 68% of respondents this year. Model-based testing (MBT), BDD, TDD, and in-sprint security tests along with predictive analytics were all mentioned as focus areas.

The key to success here is the transition between quality as a “stage” in the application delivery lifecycle, and quality as a concept owned by everyone, from planning to production. This requires evolving skills and new levels of collaboration between all disciplines, namely developers, testers, security experts, and business.

Success in continuous testing is in meeting business goals and user expectations.

40% of respondents are measuring CT effectiveness via user feedback and adoption, and via business KPIs.

Test orchestration

The top three metrics in this year’s survey for the effectiveness of continuous testing focused on business results (production data, user feedback, and ROI). More than three-quarters (78%) of respondents said that “getting visibility throughout the development lifecycle” is a challenge when implementing continuous testing.

This shows that organizations are maturing, and that they are looking more closely at the delivery of overall quality and value to the business and the end customer.

Our key takeaway is that orchestration needs to be designed to bring together the entire SDLC in a single source of truth, from release management through to deployment, with integrated tooling, quality checks, and metrics, to meet business needs.

Test environments

Test environments are one of the greatest impediments to continuous testing and Agile delivery: 36% of respondents stated that they spend over 50% of their time managing them.

The survey also reveals that the ability to spin up environments dynamically will be key, changing and combining individual components at will in line with need. To that effect, infrastructure such as code practices, containerization, cloud provisioning, and service virtualization will play a significant role.

In test data management, given the current regulatory compliance needs, it is critical to have a common mechanism to self-service, virtualize, generate synthetic data, and mask production data with demand management, governance and metrics to measure and monitor the health of testing activities.

Dynamic set-up of environments is the need of the hour.

53% of respondents stated that they will use cloud provisioning.
Service virtualization: **45%**
Containerization: **37%**
Dynamic data from production: **38%**

Shift right

In general, success in shift right includes advanced correlation between technical and non-technical data, and analysis of load and data usage patterns. It also includes greater use of crowd testing, test monitoring, and the mining of user feedback data from traditional and non-traditional feedback channels (such as social media for end-user apps).

But this practice is still far from being widespread. A solid practice includes strong use of production monitoring data and tools to understand real user journeys with the flows designed through model-based testing, so as to increase the accuracy and efficiency of test cases.

Intelligence is learning from real life: shift right is slowly but steadily gaining importance.

39% of respondents said they use crowd testing.
Chaos testing: **32%**
Monitoring: **45%**
Testing in production: **45%**

Test organization

Many respondents (40%) said their testing is still mostly supported by a specialized quality engineering (QE) team, indicating that the hybrid model is here to stay, with the evolution of centralized bodies such as a quality engineering office/quality management office. These will drive innovations in practices, tooling, and methods to scale uniformity and reusability of assets across organizations.

Our recommendation is to build more rounded skillsets, which will need to include skills transformation programs in areas such as automation, data analysis, design, code analytics, machine learning, infrastructure as code, orchestration skills, and architecture.

We are no longer focused on testing, but on quality, throughout every phase of the application delivery lifecycle.

Taking stock: quality, speed, and intelligence

Delivering quality at speed is absolutely a focus for every business, and we are seeing this reflected in practices, in organizations, and in the KPIs we use to quantify our success.

It can also be seen in the adoption of new technologies such as AI. Use cases in testing where AI can be applied include predictive analytics, prescriptive analytics, code analytics, intelligent automation, self-healing in terms of data and scripts, production analytics, and, more recently, synthetic data generation. The prerequisite to all this is availability of structured data – be it test, production, or development data – which is key for success.

In short, we are no longer focused on repetitive testing, but on driving quality intelligently, throughout every phase of the application delivery lifecycle.

AI and testing: the art of balancing logic and repetition.

42% of respondents stated they will use artificial intelligence (AI) for predictive analytics
36% mentioned code coverage
39% mentioned using analytics from operations

Shift left

Shift left is growing in scope, sophistication – and importance

What is shift left?

Shift left is a well-known practice intended to identify and address defects and areas of improvement as early as possible in the software delivery lifecycle. Instead of validating quality in a time-boxed stage later in the lifecycle, this approach allows fast and reliable testing across all disciplines: unit, functional, API, performance, security, integration, and so on. The objective is to enable rigorous testing that fits into the same cycle, sprint, or iteration while allowing all stakeholders – inclusive of those traditionally located on the “right” – to stay in alignment and remain flexible.

Key challenges

While shift left brings enormous benefits, many organizations are struggling to make it a reality. From skills and methodologies to culture change, the survey unveils the trends as well as the barriers to adoption.

In this year’s survey (Fig. 1), we find that more than half (56%) of respondents stated that they find in-sprint testing very challenging:

1. Poorly defined requirements or user stories, and complexities of maintaining multiple versions of requirements lead to coding errors and testing inefficiencies.
2. Over the last two years, around two-thirds of all respondents (67% this year) said that designing and maintaining meaningful test cases that align with user expectations is a significant challenge when implementing continuous testing. Some respondents have been sharing difficulties associated with understanding and authoring test cases from text-based requirements, with the risk of accidentally missing or designing incorrect test cases.

3. Test scripts maintenance still absorbs a disproportionate share of the investment in automation.
4. Test data and environments are greatest impediments to early automation.
5. Qualified engineers are in short supply: 62% say they are having trouble finding skilled professionals to build their continuous testing strategy.
6. Embedding non-functional testing as part of the pipeline is regularly overlooked. For instance, performance testing is an important part of the test spectrum, but is still considered to be very challenging by almost two-thirds of our respondents (63%).

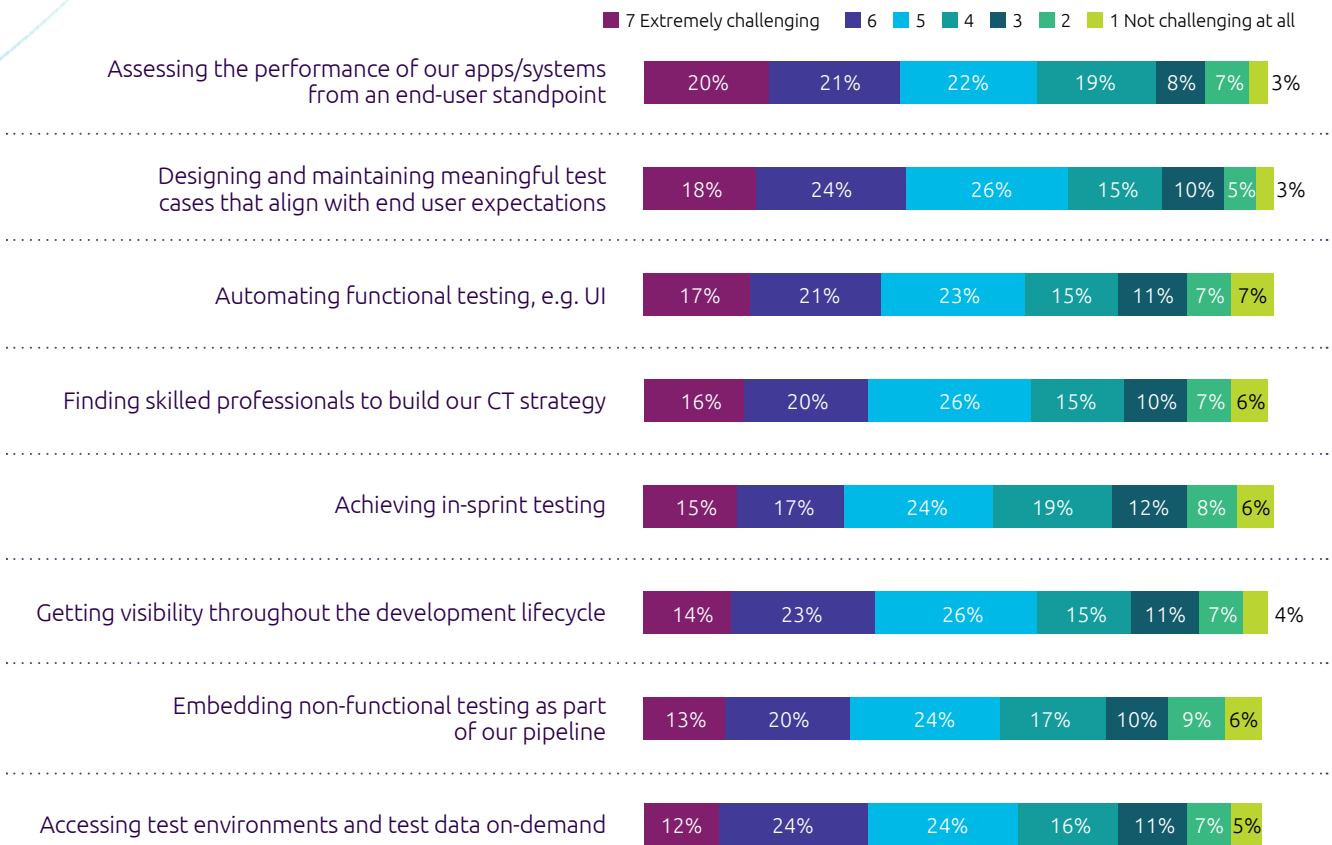
If we accept that one implicit aspect of shift left testing is the ability to recognize and accommodate the needs and expectations of end users from an early stage, our survey highlights further issues.

The rise of model-based testing (MBT)

Model-based testing (MBT) is an important part of a shift left strategy. In simple terms, it entails generation of test scenarios using system models as inputs. Examples of models include technical specifications such as flow charts, unified modeling language (UML) specifications, and process flows. With the model as a comprehensive yardstick, outcomes are quantifiable. Increasingly, MBT tools are having script engines integrated for automatic test script generation. They capture components, screen elements, and user actions in action, target, and target-type formats, and this can be passed on to the script engine for test script generation in various formats.

Because shift left takes place early in the development process, it relies heavily on the ability to understand required outcomes from the outset. This knowledge is critical for shift left in general, and for model-based testing (MBT) in particular.

Fig 1 How challenging each topic is when implementing CT - 2020



Once again, this year our survey asked about current approaches to requirements gathering, analysis and engineering. A total of 58% of respondents – the same as last year – said they pursued a full dynamic modeling approach, either with strong integration into other lifecycle tools, or simply capturing dependencies, constraints, and so forth.

We’re not sure about the extent to which this is true, and feel people aren’t doing as good a job of gathering, storing, and using test requirement and modeling information as they claim. For us, the most advanced category comprises those respondents pursuing dynamic modeling that is integrated into other parts of the lifecycle, and at 31%, this group represents less than a third of the total.

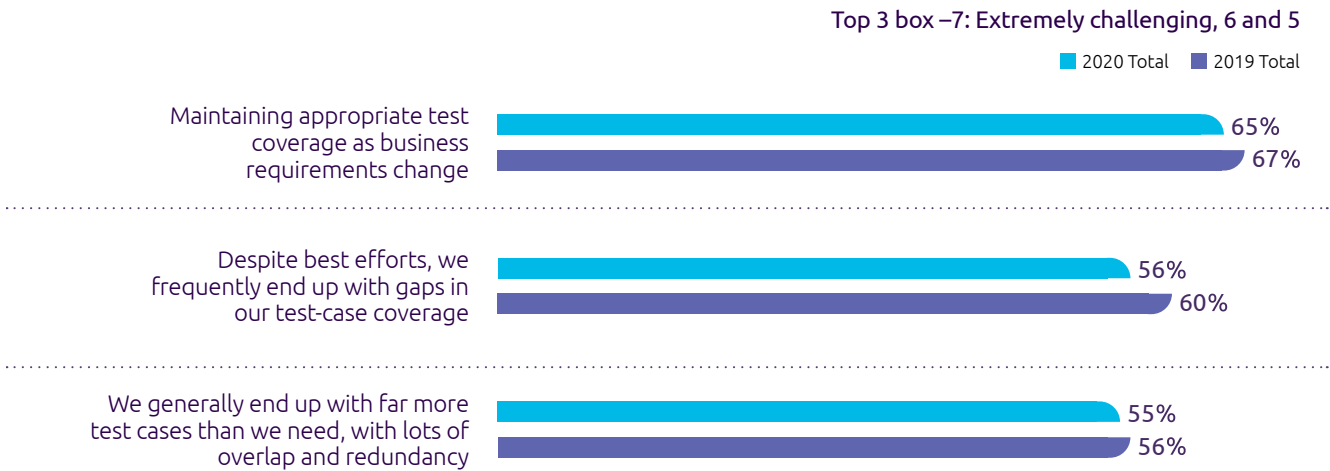
Business analysts and product managers shouldn’t necessarily write a user story, but could instead build a full, requirements-led flow of that story using models of systems. This would remove any need for interpretation: teams could test earlier, test faster, and find defects earlier. The resulting model can create full test-case requirements coverage that not only creates the test itself, but that makes maintenance easier.

We’ve expressed reservations here about the “full dynamic modeling” claim of the majority of respondents – and responses to another question do seem to corroborate our view. That question was: “How challenging is each topic when it comes to ensuring test case coverage?” ((Fig. 2).

More than half (55%) of people said they generally end up with far more test cases than they need, with lots of overlap. If they were really building models such redundancy wouldn’t happen. They seem to be modeling tests, rather than modeling based on requirements.

The same question was asked in last year’s survey, and we can see here that perceptions of this challenge and indeed of both others have dropped – but not to any statistically significant degree. What the slide seems to be telling us is that organizations are over-testing in some areas, and under-testing in others. This is why modeling is so important: it provides the test coverage that is needed, and what’s more, it knows how to write appropriate test cases, which, in turn, makes automation easier.

Fig 2 How challenging each topic is when ensuring test case coverage - 2019 and 2020 trend



The proliferation of x-driven development

In Agile, requirements were traditionally written in the form of a text user story. A user story is the user view of the system, and is typically written by the business or product owner. A system view is critical as well, and is written by developers and architects in the form of technical specifications such as UML diagrams.

Behavior-driven development (BDD) has emerged as a bridge between the user and system views. In this method, acceptance criteria are based on both the user view of the system and the behavior of the system. The concept behind BDD is defining acceptance criteria in a format, such as Gherkin, which is testable and can be automated. Test-driven development (TDD) is based on writing tests before the code is developed. Developers carry out unit testing based on these test cases, and for each problem identified, code gets refactored, hence reducing technical depth.

We see an increasing trend of writing user stories based on end user’s behavior in the survey, and 36% of respondents said they have adopted a BDD approach.

For this approach to be efficiently used, we are observing a growing emphasis on training teams in authoring user stories in the BDD format, to establish some consistency of approach, and reap speed-to-test benefits. In this year’s survey, it is difficult to tell how many follow a structured approach and how many don’t.

In addition to BDD, test-driven development (TDD) remains a favorite approach for Agile practitioners. In our view, they are complimentary approaches. TDD verifies the quality of the code the developer writes, and BDD validates the behavior of the system.

Skills becoming cross-disciplined

Shift left is taking testing out of the exclusive hands of traditional testing teams. For example, most organizations said that testing is still mostly supported by a specialized QA team, and that operational control is handled by a specific team or person, while only 33% said that testing is handled by autonomous agile teams or as a service.

But this trend seems to be moving toward shift left, because elsewhere, we see that 40% said that some testing is done by non-dedicated testers, and 20% said that almost all of it is done by them. We will look for those percentages to increase year-on-year, as shift left continues to grow in popularity. We may well see an increase in the number of specialists in full end-to-end testing that agile teams won’t do.

Building intelligence with predictive analytics

In this year’s survey, 42% of respondents stated they will use artificial intelligence (AI) for predictive analytics, and 36% mentioned code coverage. Using AI for predictive analytics is playing an increasing role in optimizing shift left activities, with organizations investing in building appropriate use cases.

In simple terms, predictive analytics is the science of drawing intelligent insights from raw sources. There are two sources of data – namely, code and application lifecycle data – that can aid in optimizing shift left activities:

1. **Code analytics** assists in determining which test cases are impacted, once code is changed. This can be achieved by mapping lines of code with test scenarios to generate a 360-degree code map.

A case study

Dynamic set-up of environments is the need of the hour.

A major financial services organization adopted a structured BDD approach by capturing user stories in Gherkin format with a cross-functional team comprising QA development and product owners. At the same time, the QA team set up a framework to automate the Gherkin format.

Previously, using a waterfall approach, testing was a bottleneck to velocity, but now, with a structured BDD approach, the organization was ready to trigger tests within the same sprint in a completely hands-free, automated manner. This resulted in improving the velocity by 30–40%.

- 2. Test optimization:** In this scenario, historical test case data for previous sprints can be analyzed by a machine learning algorithm – for example, natural language processing and random forest algorithms. These algorithms analyze test case and defect data, and assign a quantitative risk index to each test case, as well as removing duplicate test cases. This enables testers to intelligently determine what tests should be run within the sprint.

Here are some examples of how some of our financial services clients have implemented these techniques:

- When a developer checks in code, the code analytics module analyzes impacted test cases. Let us assume theoretically that there are 1,000 impacted test cases.
- These 1,000 tests cases serve as an input to the optimization machine learning module. This identifies high priority test cases and removes duplicates, optimizing the test pack to 800 test cases. This results in an optimization of approximately 20-25% in shift left activities.

As with any AI use case, the prerequisite to this is availability of relevant application lifecycle management and code data. Organizations need to invest time and effort to ensure that this data is clean and relevant prior to building their use cases.

Growing significance of in-sprint security testing

In this year's survey, and in the latest World Quality Report, security testing was ranked as highly important by respondents – and yet, despite this, it's something that still isn't mainstream in shift left. Tests are conducted separately and are then reported to developers, who then act on it. If a shift to the left is needed anywhere, it's here in particular.

The security experts who are currently providing this external support need to educate developers – and in order to do this, they themselves ought to have development skills, so they can better understand the environment in which developers work, and so they can talk their language. This will then enable developers to run security tests of their own.

Developers don't need to be security experts – but they do need to add to the skills on standards that are specific to sectors – for instance FISMA in financial services – into their repertoire, and to become instinctively more secure coders. It's incumbent on security testing experts to help them get there.

Shift left – in the future

In summary, developing a mindset of Test Early, Fail Faster, Assure Earlier is the way forward.

Model-based testing, BDD, TDD, and in-sprint security tests, along with predictive and machine learning (ML) test selection, are new ways of working in this discipline. Above all, the key to success is the development of cross-functional skills, together with collaboration between all disciplines, namely developers, testers, security experts, and business. This year's survey shows the extent to which testing is being conducted by non-dedicated testers. In years to come, it will be interesting to watch this trend, because it will indicate the extent to which shift left – and shift right too, for that matter – is being adopted.

Organizations will need to invest in skills transformation and intelligently adapt their methods and tooling ecosystems accordingly.

With inputs from

Marcus Seyfert
Managing Consultant, Sogeti

Julien Pessarossi-Langlois
Test leader, Sogeti

Komala Chandran
Transformation and Delivery Manager, Capgemini

Arunagiri Sakkaraiyalam
Senior Solution Architect, Capgemini

Stephen Feloney
Head of Products, Continuous Testing, Broadcom

Sebastien Tabarly
Head, Agile4Security, Sogeti

Test orchestration

Coordination is needed in order to ensure efficiency and transparency throughout the testing cycle

What is test orchestration?

For continuous testing, automation and orchestration go hand in hand. While test automation addresses a single task, test orchestration embeds testing into the CI/CD pipeline to tie together both automated and manual tasks and address quality as a whole. Orchestration delivers comprehensive visibility into testing, allowing organizations to understand the activities in each phase – who is responsible, areas for improvement, and track metrics over time.

Key challenges

The 2020 survey indicates that while orchestration is understood to be necessary and valuable, it brings challenges and aspirations. While 42% of respondents say they are monitoring the CT pipeline to improve releases processes and make test cycles more efficient (Fig. 3), that is not the majority, and 78% said that “getting visibility throughout the development lifecycle” is a challenge when implementing CT (Fig. 4). This percentage is virtually unchanged from last time, indicating that the issue is persistent

Fig 3 Continuous testing practices used

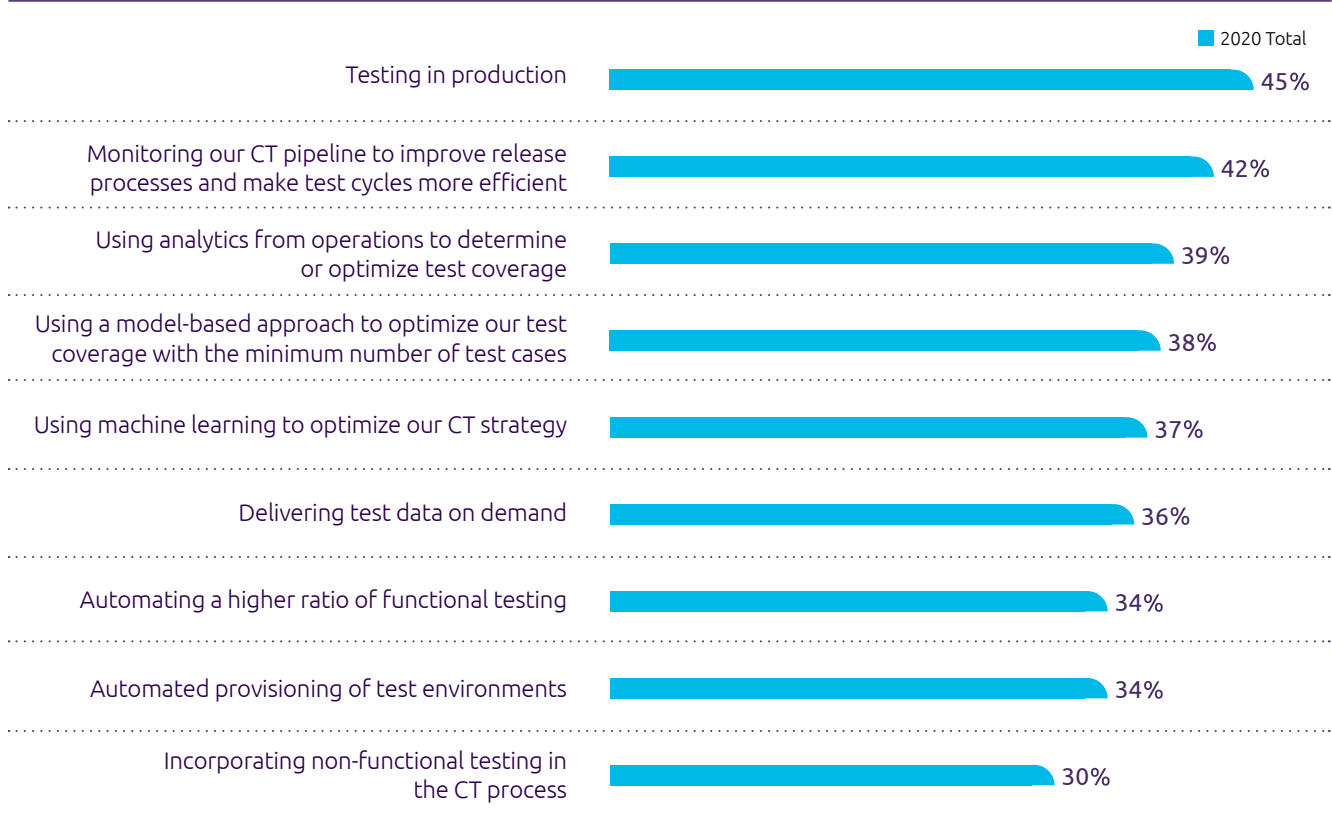
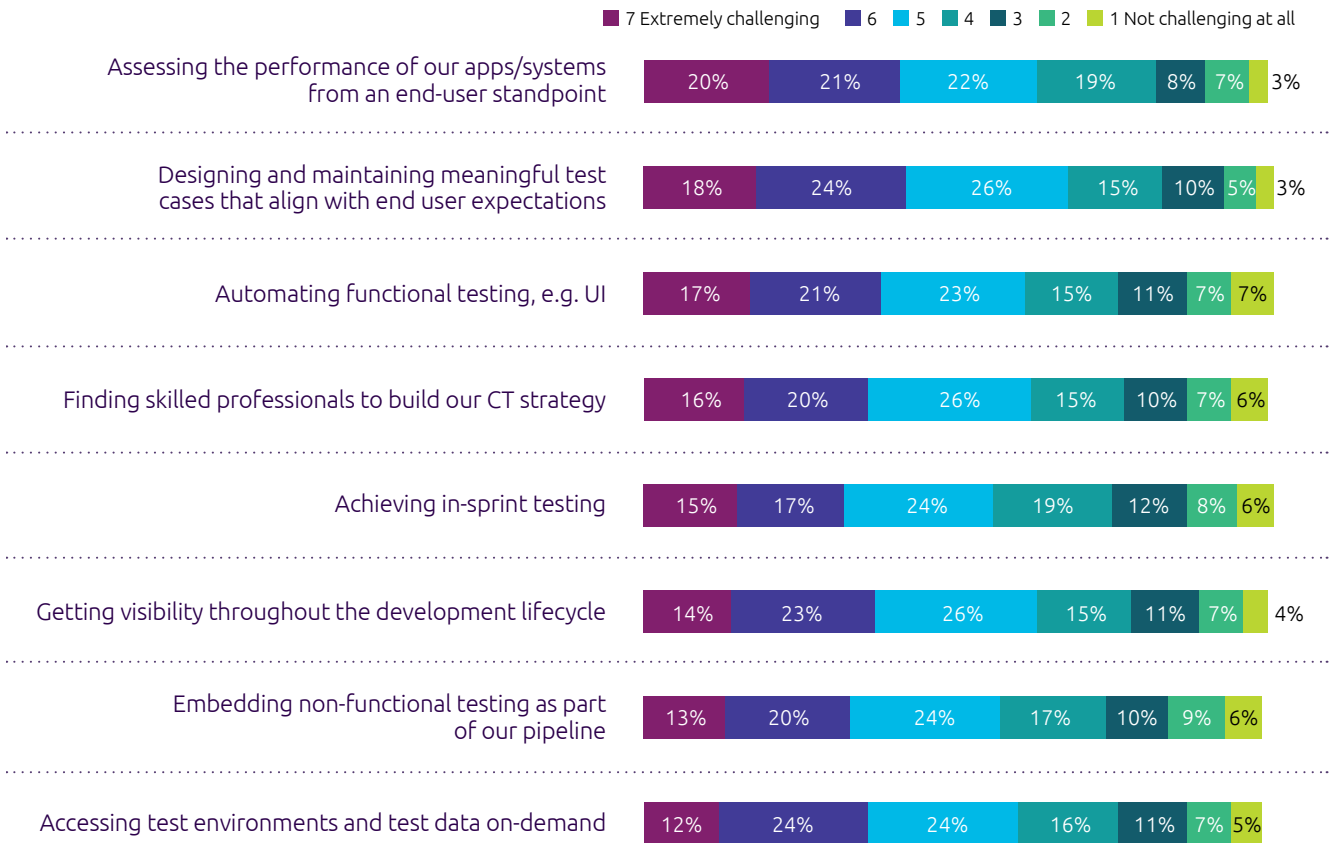


Fig 4 How challenging each topic is when implementing CT - 2020



Continuous delivery now...

Orchestration can help drive continuous delivery and continuous testing maturity. Even for teams with very manual processes, test orchestration provides an effective starting point. Since testing is so often the bottleneck in continuous delivery, the benefits delivered through test orchestration will have a direct impact on the ability to deliver quality applications at speed.

Orchestration also addresses cultural shift. Implementing an orchestrator is often an eye opener for organizations, as in many cases, there is no comprehensive understanding of who is doing what in each release, where delays and inefficiencies lie, how each team's work impacts other teams, and where to focus for notable improvements. Without orchestration, release readiness decisions are often made based on intuition rather than data-driven metrics about the specific release in the pipeline. Orchestration brings teams together in a single source of truth, with the ability to have team-based and overall application KPIs.

At the outset, releases may comprise multiple manual steps. As more processes are automated and improved, bottlenecks can be identified and fixed, and the results are seen in

the KPIs. And, because orchestration helps the process flow more smoothly, feedback is more readily available to development teams, and all teams have easy access to understand what is in the pipeline, its current status, and how delays impact the business.

... and a roadmap for the future

As orchestration practices improve, organizations will want to work toward a connected ecosystem that integrates release management, requirements, design, build, test, and deploy. An ideal ecosystem would be completely adaptive and self-reliant.

A comprehensively orchestrated system could include:

1. Continuous delivery with automated planning build, code, tests, and monitoring integrated in an assembly pipeline with 100% system availability
2. Adaptive test scripts that respond to changes in user interface
3. Real-time test data generation, where compliant test data is available on demand

- Automated code resiliency: if the release candidate fails, the code base can be rolled back to the previous version in a matter of seconds
- Continuous monitoring: uninterrupted validation and health check of services are run in an uninterrupted manner to detect issues in real time

Developer-driven releases

With effective test orchestration, testing is seamlessly embedded in the CI/CD toolchain. Code is tested as changes are made, which allows problems to be found and fixed earlier in the cycle – when they are both faster and cheaper to address. Testing orchestrated within the CI/CD toolchain also increases shift left adoption, by lowering the impact of in-sprint testing on development teams, and speeds the delivery of quality code to production.

The majority of respondents now say they are deploying new builds daily or weekly (Fig. 5), so the ability to streamline shift left testing is critical to developer productivity and sustained innovation in the organization.

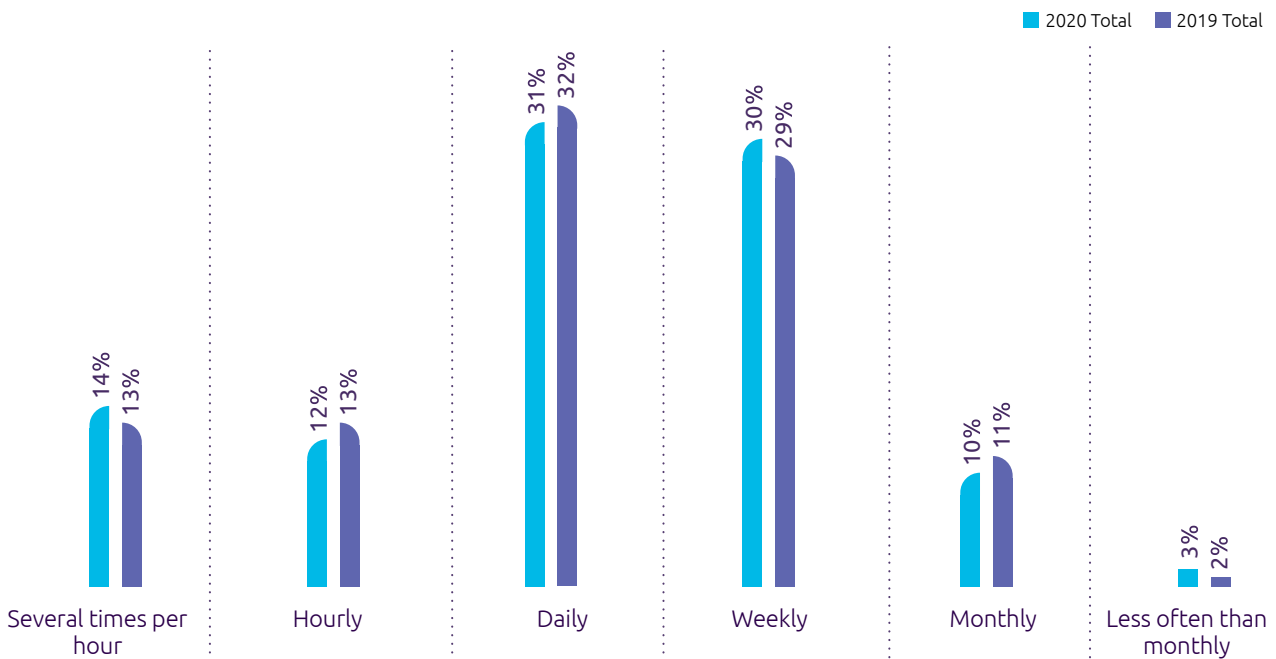
Metrics available via test orchestration allow us to:

- Have computed metrics in order to support testing activities and associated decisions
- Continuously check the evolution (quality improvement or quality degradation) of quality as a set of continuously computed metrics
- Use thresholds and compiled criteria, to act as quality gates in a continuous delivery pipeline.

When asked how they were tracking the effectiveness of continuous testing (Fig. 6), the top three responses focused on the outcome of the functionality (production data, user feedback, and ROI). This indicates that most respondents are focused on the business results rather than the process-level KPIs. However, since process/technical level KPIs are more actionable, teams will want to track those metrics that support their work and decision making.

As continuous testing matures, lower-level KPIs such as level of automation, requirements coverage, and defects found have a positive trend, and those results are seen in the top-

Fig 5 Frequency of deployment of a new build



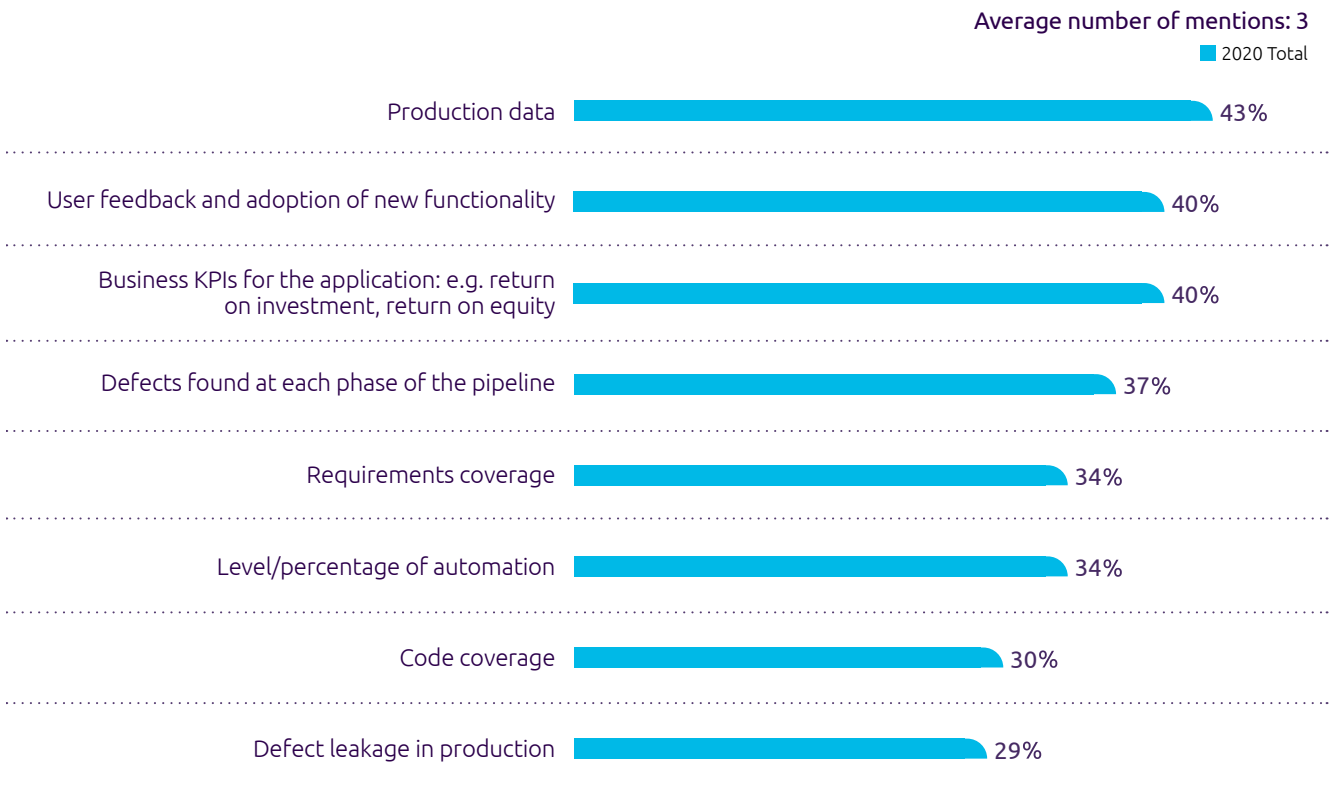
Metrics

Because orchestration connects multiple systems and processes into a single source of truth, and provides the ability to standardize those processes into a repeatable “template,” it delivers important metrics that can be used to track both technical and business level KPIs.

level business metrics that reveal overall quality and value to the business and the end customer.

There are so many possibilities for establishing metrics in testing that the key criterion both teams and organizations as a whole should set for themselves is to ask, “Does having this metric help determine whether the software will be an asset to the business?”

Fig 6 Measuring the effectiveness of continuing testing processes



Key capabilities and challenges

We see a striking similarity in how respondents ranked the challenges to test orchestration (Fig. 7). Very few found any of these challenges manageable, and a large portion found almost all of the listed factors either challenging or extremely challenging.

In the field, we often see organizations attempting to ensure applications are fully tested by “running all the tests all the time,” when understanding what needs to be tested would greatly increase accuracy while optimizing resources and test cycles. The 72% of respondents who rate “Understanding what needs to be tested” at 4 or above corroborate this (see the third bar in the chart). We also frequently see organizations facing challenges in identifying bottlenecks, as shown in the fourth bar. (Incidentally, identifying bottlenecks is an orchestration issue – but fixing them is another testing matter altogether.)

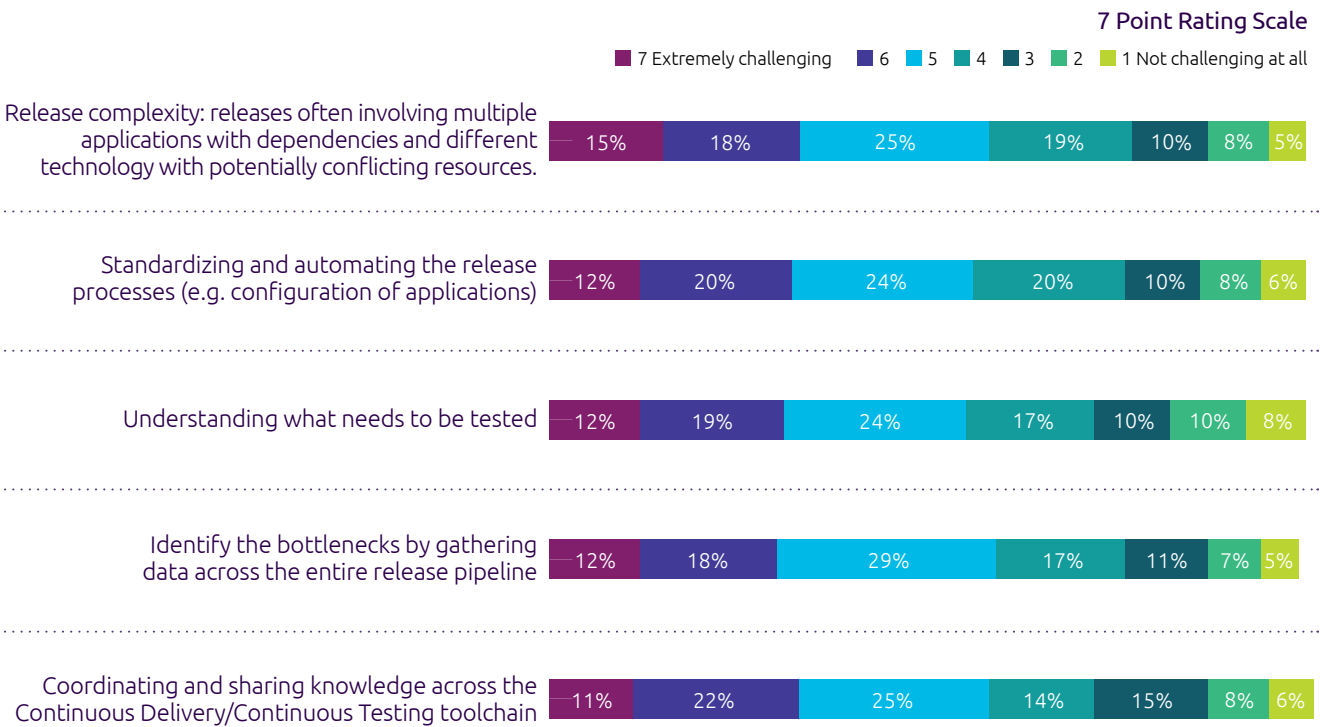
In fact, much of what we can glean from this chart relates to transparency: people are having difficulty in seeing what’s happening, which underlines the need for orchestration and the impact it can have on delivery to the business.

When asked which advanced use cases respondents are planning to address in 2020, “predictive test selection and optimization” was the number-one choice for most respondents. This helps to reduce test cycle time and ensure that testing is being conducted properly.

This choice was followed closely in our survey by the ability to correlate user behavior with requirements to define test strategy, optimizing the test schedule without sacrificing quality, and release risk prediction. These are all capabilities that are greatly augmented with the application of orchestration with AI capabilities, and we look forward to advancements and successes in these areas in the next year.

These figures show that orchestration is an important practice that will drive all the downstream activities. Simply put, this means orchestrating, automating, and building visibility in the entire release pipeline by making use of all the techniques outlined, such as standardizing and automating the release process, and identifying bottlenecks in pipelines. Without it, organizations are prevented from taking full advantage of the unique blend of tools, automated tests, and practices they have deployed.

Fig 7 Challenges to test orchestration



A case study

Velocity through orchestration

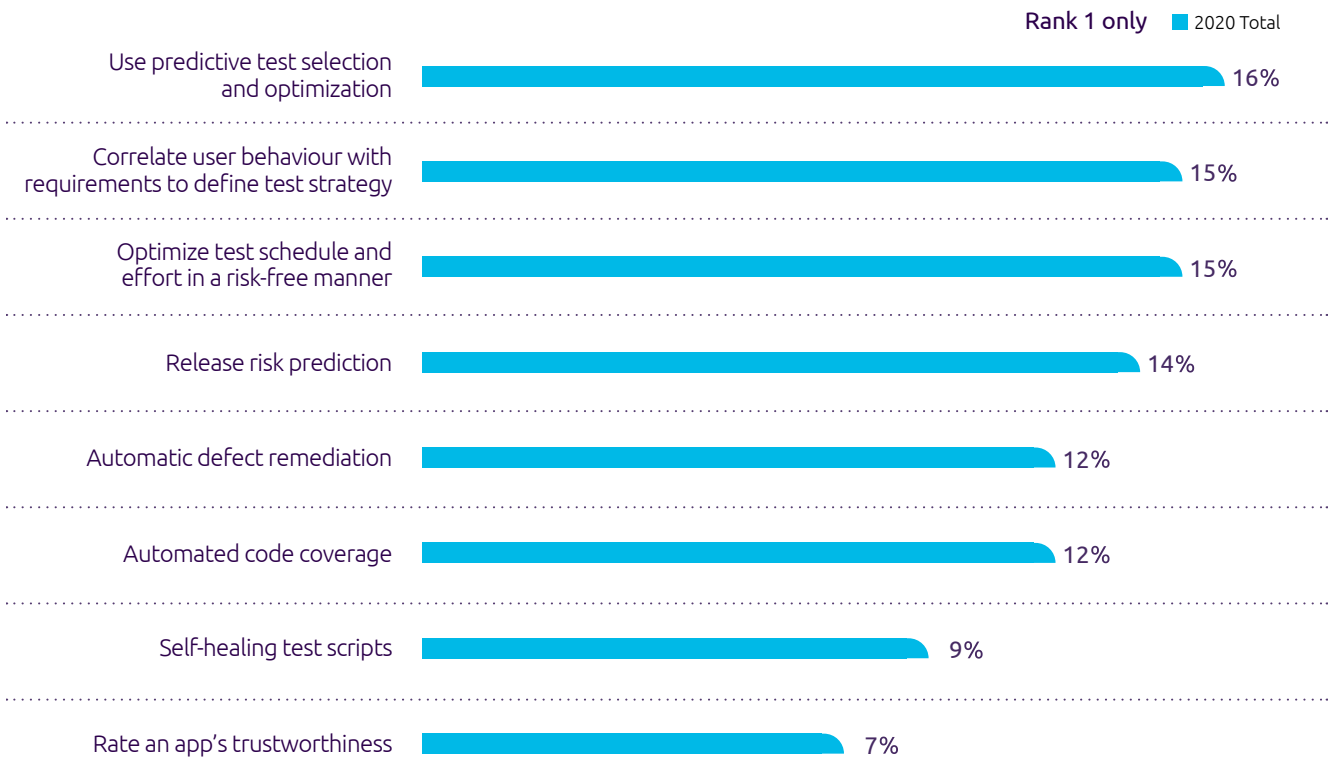
A leading global postal organization invested in test orchestration in order to streamline its processes. As a result, the release timeline was reduced from three weeks to one week, which from a velocity standpoint was the key business metric. The improvement in this respect was effectively 70%, which is highly unusual. Also, application development effort was reduced by 30%.

These results were achieved because of a sophisticated orchestration system which connects the underlying ecosystem of open source and commercial tools inclusive of release management, configuration management, testing, environment setup, and deployment tools.

When a new build is ready, the developer checks in the code and the orchestrator manages the test to deploy process. Environments are set up on the fly, unit tests are run, then API tests, then UI tests. Each test cycle is automatically promoted to the next dependent on passing quality gates in the previous cycle. If all quality gates are passed, the code can be automatically deployed to production.

The completely automated approach adds efficiency and eliminates wait times, and the templated approach and repeatable pipeline provides both visibility and metrics that deliver immediate feedback and metrics for continuous improvement.

Fig 8 Addressing advanced use cases



Key points

To summarize, orchestration is needed to make sure teams and projects properly manage test processes right through the software development lifecycle. The test strategy (or the QA guidelines) provide an approach to how to test, and to which tools to use. A test environment strategy is also needed, and should be included in the test strategy or QA guidelines.

In the short term, orchestration systems should be designed so as to bring together the entire testing process through a single source of truth with integrated tooling, quality checks, and metrics to inform release readiness and success. In the long term, intelligence will augment these processes to provide more efficient and accurate test cycles and predictive metrics for release success and business value.

With inputs from

Uri Scheiner
Continuous Delivery Product Lead, Broadcom

Mona Iversen
Test Manager, Sogeti

Albert Tort-Pugibet
CTO, Sogeti Spain

Maxime Bardou
Automation Lead ESTC -European Software Test Center, Sogeti

Prabakaran Karuppiah,
Vice President Financial Services, Capgemini

Supporting processes

Test data, environments and service virtualization

Test environment management (TEM) is a term that has sometimes been applied in different ways, so it may be useful to establish the definition we are using here. For the purposes of this report, TEM means managing a complex set of requirements related to infrastructure, data, and applications, so as to provide the ideal testing environment on demand, in order to replicate what the application will encounter in production. This, in turn, enables organizations to uphold the goals of their business in terms of time, cost, and quality.

Test environments are one of the greatest impediments to continuous testing and Agile delivery, with testing teams spending a significant amount on setting them up. This was indicated by our survey as well, where 36% of respondents stated that they spend more than half their time building and managing test environments. To respond nimbly, organizations need environments that can easily be spun up, replicated, decommissioned, and managed at scale, with practices such as cloud provisioning (mentioned by 53% of our respondents), service virtualization (45%), and containerization (37%).

In the past, test environments were treated as monolithic, static objects, and this inflexibility made it difficult to match conditions in production. In the new way of thinking, applications under test are considered in terms of individual components. This enables environments to be spun up in real time using a combination of virtualized and static components. With cloud computing platforms in particular, organizations can program the infrastructure (infrastructure as code) and apply immutable production principles, where disposable containers are configured at run time.

Over the last few years, we've seen positive trends towards the adoption of new test environment management (TEM) techniques, and this year's survey shows that more than two-thirds of respondents (68%) have a formal process for TEM that is followed by many, most, or all of their teams.

From a provisioning perspective, we have already seen the growth in prominence of cloud-based and containerized systems, which can significantly improve speeds and optimize costs. This year's survey also tells us that 46% of respondents have established or are planning to establish a scalable shared services team to support their TEM needs. We see this, too, as a positive trend: a team with designated KPIs to work towards can take ownership, share best practices, and provide visibility of TEM to senior management.

Services and data virtualization

Indeed, our survey highlights many of the challenges relating to building testing environments. They include the misalignment between production and test environments, which means test teams struggle to replicate an environment that best represents their production systems. The respondents also indicate that complex test configurations (for example, while migrating scripts to the cloud), and an inability to instantiate test environments in an automated way, are causing delays.

Like test data on demand, virtual services are a critical part of test environment management and of shifting left to achieve continuous testing. Service virtualization can provide some quick wins and allow testing to be done in parallel with development: it emulates the behavior of specific components or whole systems in API-driven applications,

cloud-based applications, service-oriented architectures, authentication, and other heterogeneous component-based applications. It gives testing teams access to dependent system components that are needed to exercise an application under test (AUT), but without the need to access the actual live components. Our survey shows that 45% of respondents said that they are exploring and implementing this approach where possible, and this is consistent with what we see in the field. With the use of service virtualization, many impediments to achieving the right test environment can be overcome.

The benefit of this approach is that any team can access any service on a virtualized basis, and test against it. It's fast, it's accurate and it will respond as though it's real, which means teams can test in parallel.

For example, a mobile payment project enabled an Australian bank to offer cardholders the ability to load their cards into their mobile app wallet, allowing both near-field communications (NFC) and in-app purchases. The greatest challenge was to integrate test environments with the app wallet. This was overcome using service virtualization for end-to-end integration testing. The entire effort was completed in eight weeks, with 767 services virtualized, 96% automation coverage, and 70% cycle time reduction. This would have been impossible without service virtualization, which has the ability to virtualize difficult backend processes that are crucial to successful testing.

Test data management

Organizations often get far in adopting and implementing an automation framework before finding out that test data undermines its speed, rigor, and stability. The feedback we have received this year on current and planned test data practices is interesting. The option that garnered the lowest response, with 30%, was the automatic generation of synthetic test data, while the ability to generate dynamic test data from production ranked fairly high, with 38%. This is somewhat surprising: in our experience, many organizations consider the use of synthetic data as a best practice, due to easier availability and data compliance implicit in regulations such as GDPR or OCC (the second-highest test data practice adopted or planned, at 41%). It also preserves all the characteristics, diversity, and density of locked-up production data.

This seems to indicate that generally, organizations are focused on providing "real-world" test data, and the only way to do that accurately is to use data from production in a compliant way. It also indicates that masking test data from production is still well entrenched with a testing team's experience and comfort level.

For example, a bank may wish to identify any geographical differences in its customer profiles, and therefore looks to segment its data so as to conduct realistic demographic analysis in individual regions, instead of across its entire area of operations. Similarly, a medical research company may wish to extrapolate from its database only those DNA

A mobile payment project enabled an Australian bank to offer cardholders the ability to load their cards into their mobile app wallet, allowing both near-field communications (NFC) and in-app purchases. The greatest challenge was to integrate test environments with the app wallet.

sequences that will be most appropriate for the cancer tests it is conducting.

This year's research also shows how much time is spent searching, managing, and generating test data. On average, our respondents told us their teams spend 44% of their time in this way. This accords with our own experience in the field: it isn't unusual to hear of waits for test data of as long as five to six weeks. These long waits can present critical problems – especially for testing teams that are looking to shift left and achieve in-sprint testing. The coding of a new feature may be complete, but without the right test data, the testing of that new feature is delayed until the right data can be obtained.

Survey respondents highlighted several core issues they encounter with test data. Top issues here all pointed to the availability of test data on demand. More than half of them (52%) said their testing teams are dependent on database administrators to get the data they need. Handoffs, prioritization, complexity, and wait times can cause significant delays in the provisioning of test data. Almost as many (49%) said their data is typically spread across multiple databases, and any test data must be consistent across data sources. Similarly, 47% mentioned the challenge of having to maintain the right test data set versions with different test versions. This can be an issue in regression testing, where there is a need to keep test data both accurate and current.

The incorporation of test data management within model-based testing as a standard capability is absolutely essential. Compliant and realistic test data should be accessed, aggregated, and created regardless of both the data source and the model. By being available on-demand to everyone across the lifecycle, test data positively impacts the ability for organizations to achieve continuous testing.

Our results also point toward a need for a simplified solution to overcome this common barrier.

AI systems and test data management – preparing for the challenge ahead

Building AI systems is hard, and validating them is harder still. Traditional testing techniques are based on fixed data inputs. Testers are hard-wired to believe, that given inputs x and y, the output will be z, and that this will be constant until the application undergoes changes. This is not true in AI systems: the output is not fixed. It will change over time, as we know more, and as the model on which the machine learning system is built evolves, as it is fed more data. This will necessitate the adoption of test data management strategies that are very different from traditional TDM techniques. It will mean building synthetic test data from training data sets and continuously refining based on algorithmic precisions defined by the machine learning algorithm.

Test environments – a checklist

In conclusion, in order for real progress to be made in TEM and TDM, we feel the following points should be addressed:

- To drive scale and best practice, test environment and test data management are best treated as shared services.
- Test environments should be regarded not as unchanging entities to be nurtured, but as functions in the test cycle to be spun up or destroyed at will, with virtualization, infrastructure as code practices, containerization, services, and test data.
- In test data management, it is critical to have a common mechanism to self-service, virtualize and mask data with demand management, governance, and metrics to measure and monitor the health of testing activities.
- In particular, it is essential to have robust sub-setting procedures to address regulatory compliance issues with masking procedures using a combination of custom framework and standard TDM tools with customized automated frameworks to maintain integrity of data.
- Last but not least, build a strategy and workforce to design data to test AI systems now.

With inputs from

Anish Behanan

Senior Director, Financial Services, Capgemini

Jeffery Hughes

Product Marketing, Continuous Testing, Broadcom

Keith Puzey

Client Services Consultant, Continuous Testing Broadcom

Shift right

The more an organization learns from experience, the better it will test

The whole premise of continuous testing is that software quality is not something that happens at a fixed point in the software development lifecycle – and that means that even when new capabilities are entered into production, quality assurance isn't over. Shift right involves using the depth and richness of production information and users' behavior, feeding it back, learning from it, and building that knowledge into new iterations. It allows continuous testing to be extended, so as to create a more holistic approach in the development lifecycle, increasing the accuracy of the validation process.

Shift right is revealed in many of the main continuous testing practices used by our respondents. The most popular responses fall overtly into this category: testing in production; monitoring the continuous testing (CT) pipeline to improve release processes and make test cycles more efficient; and using analytics from operations to determine or optimize test coverage. Similarly, the advanced use cases mentioned by respondents as areas they are addressing or planning to address in 2020 include correlating user behavior with test requirements to define the test strategy (40% of respondents).

Shift right testing spans incorporating analytics-gathering capabilities into the applications, monitoring the health and performance of applications and services in production, and aggregating users' sentiments through cognitive services.

While the concept of shift right isn't new, the challenges organizations find in continuous testing are as applicable to shift right as they are to any other field.

Shift right improves the accuracy of testing via analytics

The challenge of the end-user standpoint raises a fundamental question about the accuracy of testing as a whole. Indeed, designing and maintaining meaningful test cases that align with end-user expectations remains a key challenge, as mentioned by 68% of respondents this year.

Even when projects apply static analysis steps and tools, and even if we combine exploratory testing with effective test automation, the quality validation will still miss some of

what is useful and relevant for end-to-end quality assurance purposes. To minimize such gaps and optimize their test coverage, 39% of respondents are using embedded libraries and other types of analytics. This is encouraging, yet insufficient.

This year's survey shows that 63% of respondents face challenges in getting visibility throughout the development lifecycle. In our experience, many teams are nonetheless benefiting from the insights they gain from events and data analytics. Here are a few real-life examples:

- User flows are established alongside aspects of learnability and task completion. Real user journeys are correlated with the flows designed through model-based testing, increasing the accuracy of the test cases
- GUI-related analytics, inclusive of heatmaps, are used to make informed decisions during A/B testing. This technique compares metrics and user behavior around different variants of a variable, by deploying it to different sets of users to gauge effectiveness. A/B testing is a well-established practice that is becoming easier now, and for two reasons. First, releases are coming out in smaller 'chunks', which simplifies identifying the impact of a single variant. Second, A/B testing tools are getting more mature in their delivery mechanism and rolling out the production environments associated to each variant. To enable consistent experimentation and rapid change, A/B testing technology must indeed be integrated into a delivery pipeline capable of rolling out new features on-demand, with monitoring and metrics, and delivering that actionable data back to development
- Meta-data on end users' context and settings are used to create more realistic automated scripts and ease the reproduction of crashes. In particular, mobile analytics are leveraged to understand which devices are used most (and should be tested first). Real-world traffic is used to identify API load on the backend, and overall data is used to increase the volume and depth of feedback during beta testing phases

The missed opportunity in Shift right: customer sentiment

User experience is emotional, and is driven by how users think, perceive, and feel. Intimacy, immediacy, and privacy are key factors. Users' perceptions matter tremendously: when they enjoy using an app, they will keep using it, they will tell their friends, and they will encourage others also to use the app. When things go poorly, the downside impact to an application, and a brand, is even more notable. While more teams are using analytics to "correlate user behaviour with requirements to define test strategy", this technology is still not mainstream in monitoring end users' sentiment and emotions.

Similarly, continuous testing practitioners are still not really considering social media or trustworthiness feedback as an integral part of their process. Reviews, in particular, can complement monitoring by filling the gaps between what is being monitored and what is important to the users.

As artificial intelligence and unsupervised machine learning algorithms evolve, analytics of unstructured feedback will improve, enabling organizations to align test cases more closely to customer journeys and customer expectations.

Towards more realistic performance testing

Numerous studies have already demonstrated the severe impact of poor application performance on the business. Performance impacts conversation rates, average order value, and brand perception. It becomes even more relevant to measure both the perceived performance and the actual performance, across every step of a transaction journey (front-end, network, and back-end).

To get the most complete picture of overall performance, many forward-thinking organizations are taking on the challenge of enhancing their test scenarios by adding deep inspection capabilities provided by monitoring tools, which can isolate the exact issue causing the performance degradation. Such issues include the call chain for a slow

method, a database request that's holding things up, and a dependent service that's misbehaving. Using application performance management (APM) tools was noted to generate positive cross-team dynamics. Respondents also reported that they are looking at auto-remediation and self-healing test scripts.

Other Shift right techniques

- 39% of respondents said they use **crowd testing** from sources such as social media feeds. This approach uses a widely dispersed group of pre-approved third-party software professionals who test software on their own devices. Good tools are available to simplify the process. Its real-user perspective is a distinct advantage, and useful analytics tools are now available to assess factors such as demographics
- **32% said they perform chaos testing.** This practice randomly disables production instances to assess its overall resilience and make sure it does not impact end users. By running chaos testing in a carefully monitored environment, we are able to gauge its ability to withstand any conditions (see below)
- **Test monitoring:** this approach evaluates and provides feedback on software currently in production, either at designated intervals or continuously. Indeed, relevant CT practices highlighted by our respondents included testing in production (45%), monitoring the CT pipeline to improve efficiency, and using analytics from operations to optimize test coverage. All this monitored data is used to make data management more dynamic and productive. Also, monitored production data can be used to create virtual services based on real user behavior. In addition, monitoring can be conducted in test as well as in production, and comparisons can be made to identify and address any differences

Key recommendations for Shift right

To shift right, organizations might consider:

- Making better use of production monitoring data and tools to understand real user journeys, load patterns, data usage patterns, and end-to-end performance that includes client-side performance
- Running these scenarios in right-context pre-production, leveraging techniques such as service virtualization as well as scale-model testing
- Running test scenarios in production via synthetic monitors to continuously validate and update these scenarios
- Identifying missing test cases by turning on monitoring in test environments and comparing test environment logs with production logs. This will clearly identify missing test and data scenarios
- Mining user feedback data from traditional and non-traditional feedback channels (such as social media

A case study

Shift right

A bank had developed a wealth management app, and was unable to perform enough in-sprint testing because current testing processes could not keep pace with the two-week sprint schedule. The bank therefore had to proceed with a short-term bespoke approach, using resilient scripts that were monitored in production to generate findings that were fed back in real time. This allowed the bank to react instantly to application performance and 'get out in front' of any issues, resulting in a 40% improvement in velocity.

for end-user apps) to better understand customer challenges, engage with them, and make the appropriate changes

- Employing Robotic Process Automation (RPA) for User Acceptance Tests – using RPA to simulate a manual tester in both pre- and post-production may be helpful to automate some difficult-to-automate testing scenarios (e.g. payment processing)
- Better collaboration between testers and site reliability engineers (SRE) and classic operations teams to facilitate access to production data, as well as to jointly drive better reliability engineering practices

Five steps towards chaos testing

Chaos testing is the discipline of experimenting on a distributed system to build confidence in the system's capability to withstand turbulent conditions in production. This idea, which has proven to be very successful for Netflix, is now being adopted across industries. To design tests that fail and validate recovery requires that the test professional understands the architecture, design, and infrastructure of systems. Mentioned below are five steps to achieve this :

1. Conduct a failure mode analysis by reviewing the design of the system. In simple terms, this means identifying all the components, internal and external interfaces, and identifying potential failures at every point. Once failure points have been identified, organizations should establish that there are indeed alternatives to failure. For example, let us say we have a service-based architecture, and if the application depends on a single critical instance of service, it can create a single point of failure. In this scenario, organizations should verify that if there is a request time/out, then an alternative is available
2. Validate data resiliency. i.e. there is a mechanism for data to be available to applications, even though the system that originally hosted the data fails. Organizations should verify that the data backup process is either documented or automated. If it is automated, then it should be confirmed that the automated script backs up data correctly, maintaining integrity and schema
3. From an infrastructure standpoint, configure and test health probes for load balancing and traffic management. These ensure that the system is not limited to a single region for deployment in case of latency issues
4. From an application standpoint, fault injection tests should be conducted for every application in the system. Scenarios include shutting down interfacing systems, deleting certificates, consuming system resources, and deleting data sources
5. Conduct critical tests in production with well-planned canary deployments. Organizations should verify that there is an automated rollback mechanism for code in production in case of failure

A case study

Shift right

From the Google Play console, the development team of a mobile app witnessed that feedback on stability and uninstall had a major drag on the rating. The team was able to use the information provided in these reviews to discover, consider, replay and prioritize the problems end-users were reporting. They also addressed several misunderstandings about what the app provides, clarified in-app purchases, and simplified a few critical flows.

This helped improve the ratings and ranking of their app.

Above all, the key to testing resiliency is continuous learning of the design architecture and infrastructure of systems – because the more an organization learns, the more it will understand points of failure, and the better it will test.

Yet another evolution of skills

One key challenge that needs to be addressed is the issue of the skills gap in shift right initiatives. Since production data is typically voluminous, and almost impossible to process manually, testers now need to develop skills in data analytics, data mining and machine learning.

Elsewhere in this year's survey, we see that production data is the principal measure for the effectiveness of continuous testing, and yet most testers today don't know how to deal with it. There is much progress to be made, as the goal of shift right is also to reduce waste so as to optimize the entire delivery pipeline.

With inputs from

Jacqueline Ike
QA Consultant, Capgemini

Sudhakar Jharbade
Senior Manager, Financial Services, Capgemini

Charuta Deshpande
Senior Manager, Digital Assurance and Quality Engineering, Capgemini

Isaac Alvarez
Innovation Lead, Sogeti

Shamim Ahmed
CTO, Continuous Delivery, Broadcom

Test organization

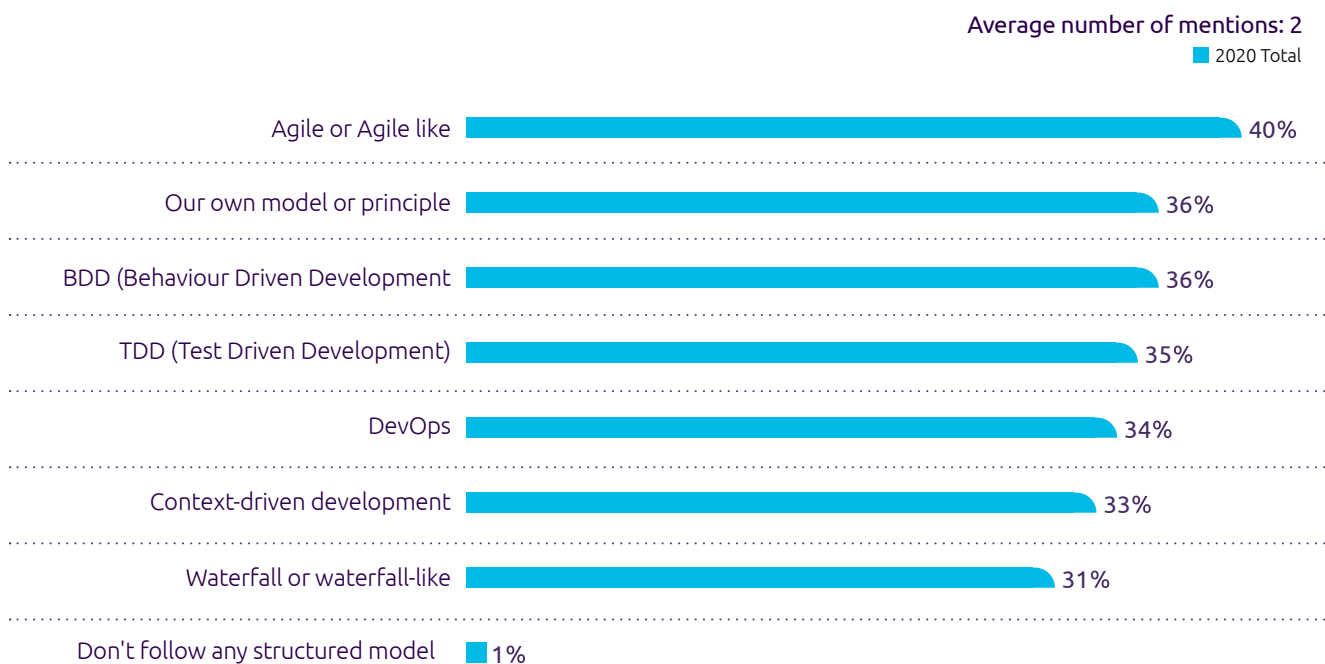
Gravity is shifting between testing in the center and testing in the field – and roles are being transformed

In a world in which most organizations are moving to Agile and DevOps, it's useful to take stock of how they are organizing their approach to testing. To what extent are they centralizing – or indeed, decentralizing? What testing models are they currently using? What are the skills implications? What future trends, if any, can we identify? And now that testing is being democratized, with shift left approaches such as behavior-driven development (BDD), as well as with shift right practices, how much testing is now being done by non-dedicated testers?

In this year's survey we asked about the development and testing models that organizations have adopted (Fig. 9). The size and shape of the responses we see in the graph below do not surprise us. In recent years, time-to-market has shortened: businesses and, by extension, apps, are more customer-driven, so there is a need for greater speed – and for greater flexibility, too.

Current test organizations vary. Many of our respondents (40%) told us their testing is still mostly supported by a

Fig 9 Development and testing models adopted



specialized quality engineering (QE) team (Fig. 10), although there is a clear trend in the data of this year's report that shows organizations are shifting left: this is borne out by the fact that 33% of respondents reported the direct handling of testing by their autonomous Agile teams. Here, testing is embedded into the application development lifecycle as a core activity. It is no longer the law enforcement of quality, but an enabler for all stakeholders to make informed decisions at all times. Agile quality engineers are natural members of the Agile team and help define the in-sprint test strategies, automate test scenarios, monitor quality levels, and execute specialized test activities. To help shape culture toward early testing, they participate in the definition of "done."

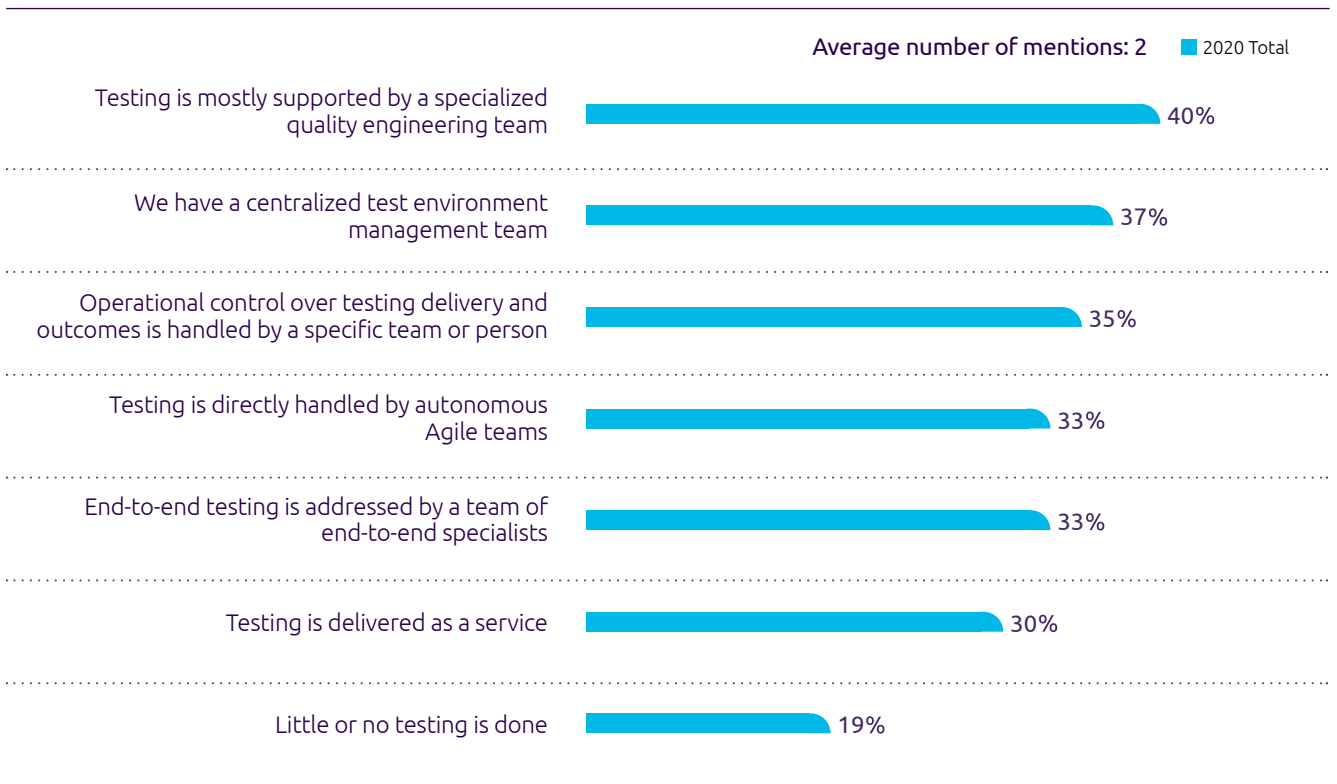
We do find, though, that teams are shifting left at different speeds, and in different ways, to address their own specific bottlenecks. Indeed, we're witnessing what may be termed a hybrid approach, where organizations are retaining centralized teams that cover areas including test environment management, operational control, service virtualization, and security management – but at the same time, they are delegating test functions to decentralized teams that bring together a range of development and test skills.

This hybrid approach may, for some, become a norm: for organizational and cultural reasons, it simply may not suit them to shift left entirely.

Within an Agile-at-scale organization, the level of dependencies and overall complexity of the released system as a whole can grow exponentially. It can be hard to integrate in an Agile development context, because the level and types of testing can vary – functional end-to-end testing, for example, or technical end-to-end testing, or security. The challenge is to protect the end-to-end customer journey validation across the various squads, e.g. via covering integration risks. One third of respondents have implemented an end-to-end team to ensure the various quality attributes are assessed with a unified perspective. Some respondents mentioned they have gathered a cross-functional team, with people drawn from various squads, and that they have implemented a feature test matrix to facilitate such an end-to-end view on quality.

Over a third of respondents (35%) said they are maintaining operational control over testing delivery within a specific team. As long as they avoid falling in the top-down overcontrol trap, this proves extremely beneficial in driving continuous QA innovation in terms of processes, people and technology. Such a team maintains the enterprise QA guidelines, the standard test automation framework, and facilitates test data and test environment provisioning. In some cases, this team is also responsible for implementing an enterprise-wide culture of KPIs, and gives visibility into the productivity and quality of the system under test.

Fig 10 Current test organization

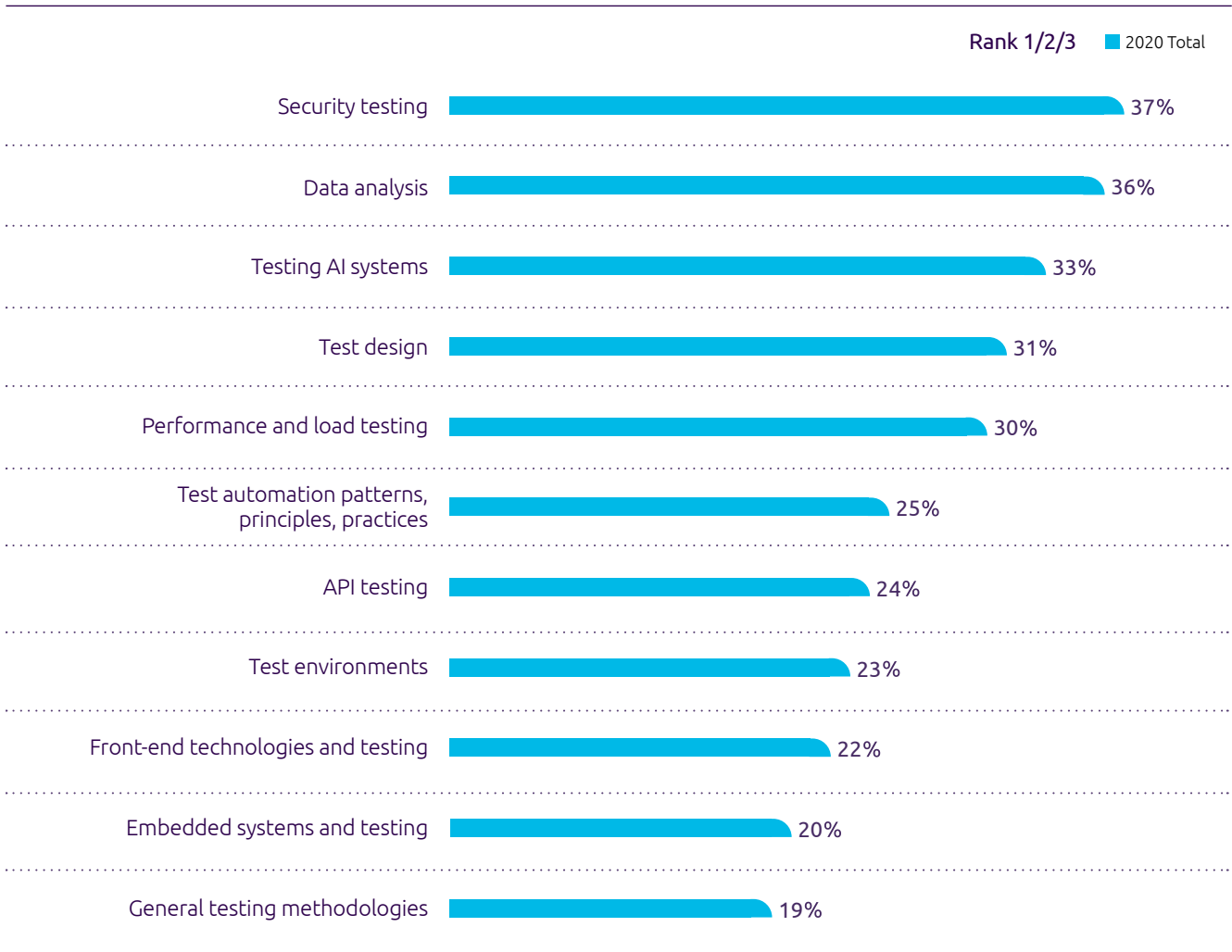


Skills

To differing degrees, test organizations require all the skills highlighted in this year's survey (Fig. 11). The lines between teams are blurring, and companies are looking to develop teams with a greater breadth of skills – skills that extend beyond the conventional testing remit. Workforces are

important by 37% of respondents, but its place in the software development lifecycle is, in our view, evolving. Rather than forming the end-stage of the cycle, security testing is increasingly regarded as an integral part of the continuous testing process – and that, in turn, creates a need for developers themselves to become not only more security-

Fig 11 Skills important in the coming year



being transformed across the SDLC, as new recruits join straight out of college, bringing with them competencies in architecture, in business, and beyond, as well as in technology.

In terms of organizational priorities, test automation is still the core skill that centralized testing organizations need to make shift left a reality. Some specialized skills ranked above this core skill in this year's data, most likely because skills in this area are increasingly being seen as an understood, ongoing need.

It's understandable to see security ranked as highly

aware, but better able to test for security themselves at each and every stage.

The need to develop skills in testing AI systems was ranked highly by a third of respondents (33%). It is indeed difficult to test decision-making routines, particularly when the data on which those decisions are being made is itself constantly updating and changing, as it does in many AI systems. This is just one driver for data analysis skills being ranked highly by 36% of respondents. However, the high ranking given to AI testing skills doesn't tally with current levels of maturity. It seems more aspirational than actual.

The evolving role of specialists

The extent to which testing is now being conducted by non-dedicated testers (24% for all or almost all testing, and a further 40% for some testing) suggests that organizations are increasingly mature in their continuous testing practices, where everyone is responsible for quality.

The presence of dedicated specialist testers is likely to be an indicator of a continuing center-of-excellence model. However, the role of that model is changing. The highly integrated nature of modern systems, combined with the increasing number of apps and routines, means that testing is growing both in scale and complexity. The ripple effects of this are being felt far beyond any traditional center, right across the entire software development lifecycle (SDLC).

This evolving central testing function is transforming into more of a consultancy role than that of a centralized testing provider. It can arbitrate: for instance, it can decide that a problem is not a testing issue but an architectural matter, and can provide appropriate advice and assistance. In Agile and DevOps environments in particular, we're seeing these central testing teams trying to upskill themselves in domains such as DevOps/CI/CD, performance testing, service virtualization, test environments, test data, and security. Their objective is to specialize in as many capabilities as they can so they can contribute to the overall quality of the user experience. The more capabilities they master, the more they can embed quality gates throughout the process and teach/enable the Agile teams so they can leverage those capabilities by themselves.

In short, this center of excellence or enablement will act beyond testing, and across development and deployment, addressing bottlenecks, and becoming highly outcome-driven. Indeed, we might say that traditional testing resources are now being asked to apply their expertise beyond the traditional testing remit – and the organization as a whole will be all the better for it.

With inputs from

Dhiraj Sinha

Vice President, Financial Services, Capgemini

Albert Tort-Pugibet

CTO, Sogeti Spain

Gustava Matus

Senior Consultant, Sogeti

Alex Martins

Head of Solutions Engineering, Continuous Testing,
Broadcom

Amir Zabetian

Senior Manager, Sogeti

Continuous Testing: the road ahead

The future for continuous testing – and how to get there

Continuous testing isn't merely a process conducted for its own sake. Its aim is to ensure quality and to mitigate against risk throughout the software development lifecycle. It doesn't simply meet technology objectives; it is grounded in business and enables organizations to respond to market conditions quickly and effectively, to achieve competitive advantage, to increase revenue, and to improve operational efficiency – all with no risk to quality.

How is continuous testing evolving?

The status of continuous testing that we can glean from this year's report is a fair reflection of the current climate as we perceive it across the global markets we collectively serve. Maturity varies widely across geographies, sectors, and individual organizations, with a variety of bottlenecks.

In the alignment of quality with business goals, we are seeing, first, a more widespread use of quality gates, measuring against KPIs of many kinds – business as well as technical – so as to satisfy all stakeholders. Second, we are seeing an increasing need to develop more rounded skillsets, inclusive of development, design, and operations. Time and time again in this report, it has been noted that the evolution of both business and technology is creating new demands of people, in areas including data analytics, artificial intelligence, and cross-functional testing practices. Effecting transformations of these kinds in the workforce can prove more difficult than transforming an organization's technology.

- We see shift left practices maturing with behavior-driven development (BDD), model-based testing, and service virtualization techniques. Security takes center stage, and there is an increasing importance in bringing security testing into the orchestrated testing pipeline
- From an organization standpoint, we see the industry coming to terms with a hybrid model, with decentralized teams aligned to scrum teams and a centralized community of practices for specialized skills
- We are also seeing a shift with continuous testing becoming an integral part of CI/CD pipeline orchestration. The intent is to drive velocity without impacting cost. There is also a greater maturity in tools, with platforms being developed to integrate both

open source and shift left and shift right tools to drive adoption across teams, to allow quicker time to value, and to lower toolchain maintenance of complex, integrated quality capabilities

- In some areas, there has been little movement – for example, in approaches to release orchestration and in the development of test data and test environments. Test data and environments continue to be the biggest impediment to continuous delivery. There are encouraging signs of companies investigating the lifecycle of data (and not just the provisioning step) a little more seriously
- In other areas, the enthusiasm seems to be aspirational rather than actual. For example, while more teams are using AI and analytics to select and prioritize test cases, and while products are available, the ability to monitor end users' sentiment and emotions is still in its early days.

What are the actionable takeaways?

Given how little change is reflected in this year's report, there is a clear need to focus on areas in which most progress can be made. In a sense, the "where next?" question should be posed not merely about continuous testing, but holistically – across requirements, design, development, operations, and indeed the entire software development lifecycle.

In our view, areas of focus should include:

Strategy

- The embedding of testing throughout the lifecycle, with functional and non-functional tests such as security and performance testing taking place at each stage, and not simply as a final rubber stamp.
- Zero-touch testing: using development, design, and production data to create functional test scripts automatically. Production data provides a good pointer – and, when coupled with emerging analytics technology, it will increasingly be able to generate actionable cognitive insight.
- The whole-lifecycle approach means a new mindset is needed – an attitude in which quality is not merely a factor in development, but is an intrinsic part of it. It is only when quality is an imperative, and when companies

move both left and right to facilitate it, that they can move fast with confidence.

Shift left

- New automation techniques will continue to evolve in shift left, including model-based testing, BDD, TDD, and in-sprint security tests, along with further application of machine learning algorithms for predictive, prescriptive, and intelligent automation. Intelligence will be built into automation design practices with a focus on self-healing techniques, and resiliency built into scripts and data to drive speed, particularly in the case of in-sprint automation.
- Additionally, AI/ML will be used for defect prediction to pro-actively understand potential sources of defects and release risk.

Organization and skills

- Workforce transformation: as noted above, new, more rounded skillsets are needed for the future. The road ahead will need to include skills transformation programs in areas such as data analysis, design, code analytics, machine learning, infrastructure as code, orchestration skills, and architecture.
- In line with this, from an organization standpoint we see hybrid models maturing, encompassing decentralized and centralized organization teams to drive economies of scale. Changing culture is as important as changing technical skills, and we see a top-down effort in several organizations to drive this cultural change. We also see a trend of centralized bodies such as a quality engineering office/quality management office evolving to drive innovations in practices, tooling and methods to scale uniformity and reusability of assets across organizations. The expectation is that the role of a test professional should also be elevated from an operational role to that of a strategist, with quality engineering consulting roles also gaining importance.

Test orchestration

- Continuous orchestration systems will be increasing, designed so as to bring together the entire release process in a single source of truth, with actionable insights brought together by AI algorithms gathering data across tools and across this pipeline. This requires integrated tooling, quality checks, and metrics to inform release readiness and success. This will also necessitate robust engineering practices, strong release management and deployment practices.
- A comprehensive approach is needed to govern and measure the health of testing activities with appropriate metrics, from requirements management, test assets, execution, environment and data, so as to drive tactical and strategic decision-making.

Supporting processes

- Addressing test environments and test data blocks is necessary to achieve maturity in continuous testing. Test

data lifecycle management will be key for compliance in the road ahead. These include practices such as centralized demand management and governance with self-service features to generate, search, mask, and reserve data using a combination of automated techniques and virtualization.

- From a test environment standpoint, the ability to spin up environments dynamically will be key. To that effect, infrastructure such as code practices, containerization, and virtualization will play a significant role.
- There is a great deal of potential in the application of AI to drive efficiencies in testing across the cycle. New use cases will emerge in areas including test data, predictive analytics, prescriptive analytics, and dashboards. Organizations will also need to look at how to test AI applications, and build intelligence into the automation practices by using machine learning algorithms to determine which tests to automate, self-healing and test script resiliency – so those are areas to look forward to as well.

Shift right

- As noted above, automation will also continue to develop in shift right, with greater use of analytics applied to production data, more testing in production, more crowd testing and chaos testing, and with greater exploration of user journeys and of the effects of customer sentiment. This exploration is likely to include the mining of user feedback from traditional and non-traditional channels.

With inputs from

Maheshwar Kanitkar

Senior Director, Digital Assurance and Quality Engineering, Sogeti

Michael Buening

Director, Sogeti

Marco Venzelaar

Managing Consultant & Lead Technologist, Sogeti

Alex Martins

Head of Solutions Engineering, Continuous Testing, Broadcom

Shamim Ahmed

CTO, Continuous Delivery, Broadcom

Stephen Feloney,

Head of Products - Continuous Testing, Broadcom

About the study



Continuous Testing Report 2020

The Continuous Testing Report is based on research findings from 500 interviews carried out during October 2019 - January 2020 using an online survey approach with some telephone interviews where required and preferred by respondents. The average length of each interview was approximately 20 minutes and the interviewees were all senior decision-makers in corporate IT management functions, working for companies and public sector organizations across eight countries.

The interviews were based on a questionnaire of 24 questions. Quality measures were put in place to ensure that the questionnaire was understood, answered accurately, and completed in a timely manner by the interviewee.

For this research, we selected only organizations with more than 500 employees (in the respondent's national market).

Research participants were selected to ensure sufficient coverage of different regions and vertical markets to provide industry-specific insight into the testing and data extraction practices within each sector.

To ensure a robust and substantive market research study, the recruited sample must be statistically representative of the population in terms of its size and demographic profile.

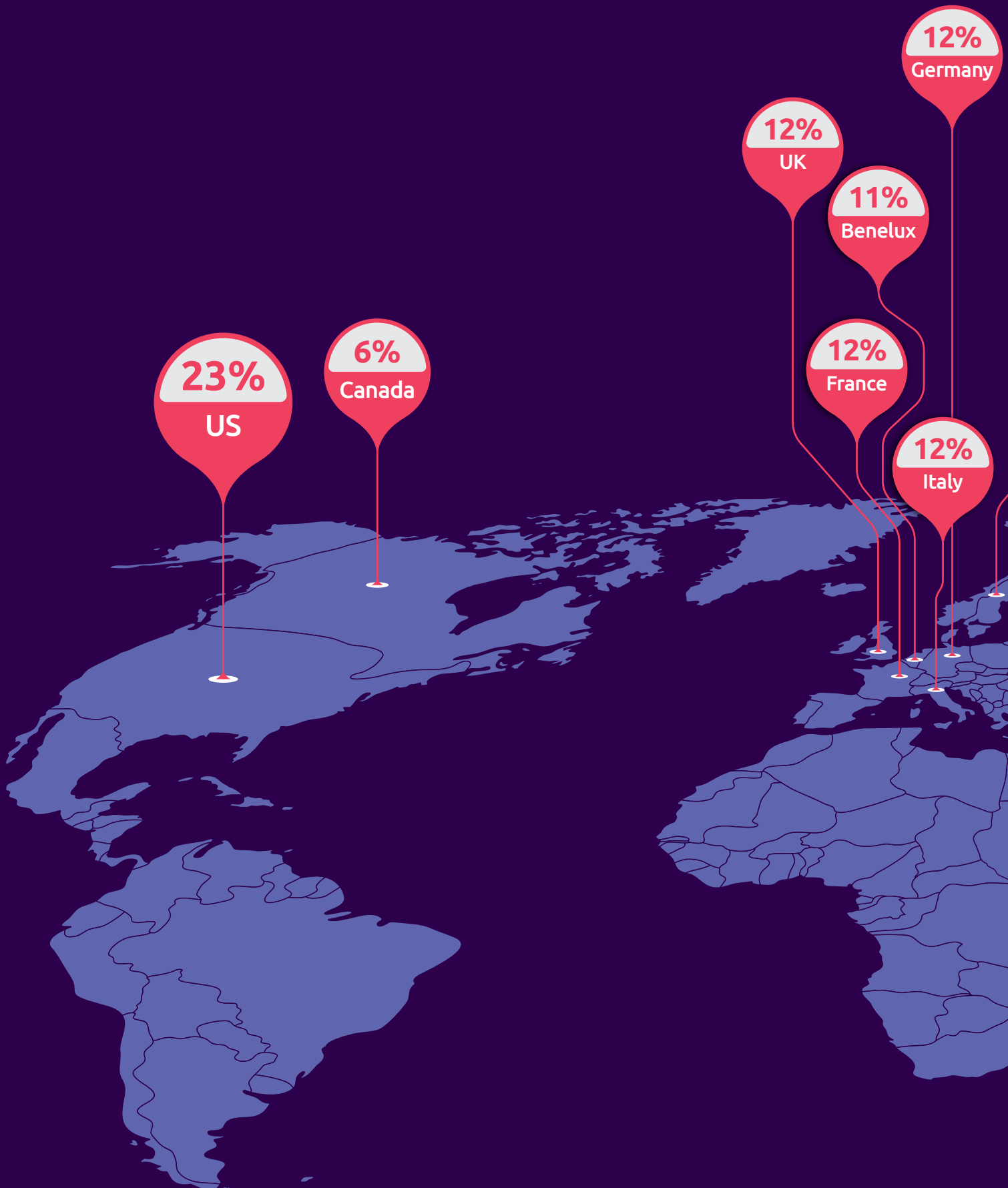
The required sample size varies depending on the population it represents – usually expressed as a ratio or incidence rate. In a business-to-business (B2B) market research study, the average recommended sample size is 100 companies. This is lower than the average sample size used for business-to-consumer (B2C) market research because whole organizations are being researched, rather than individuals.

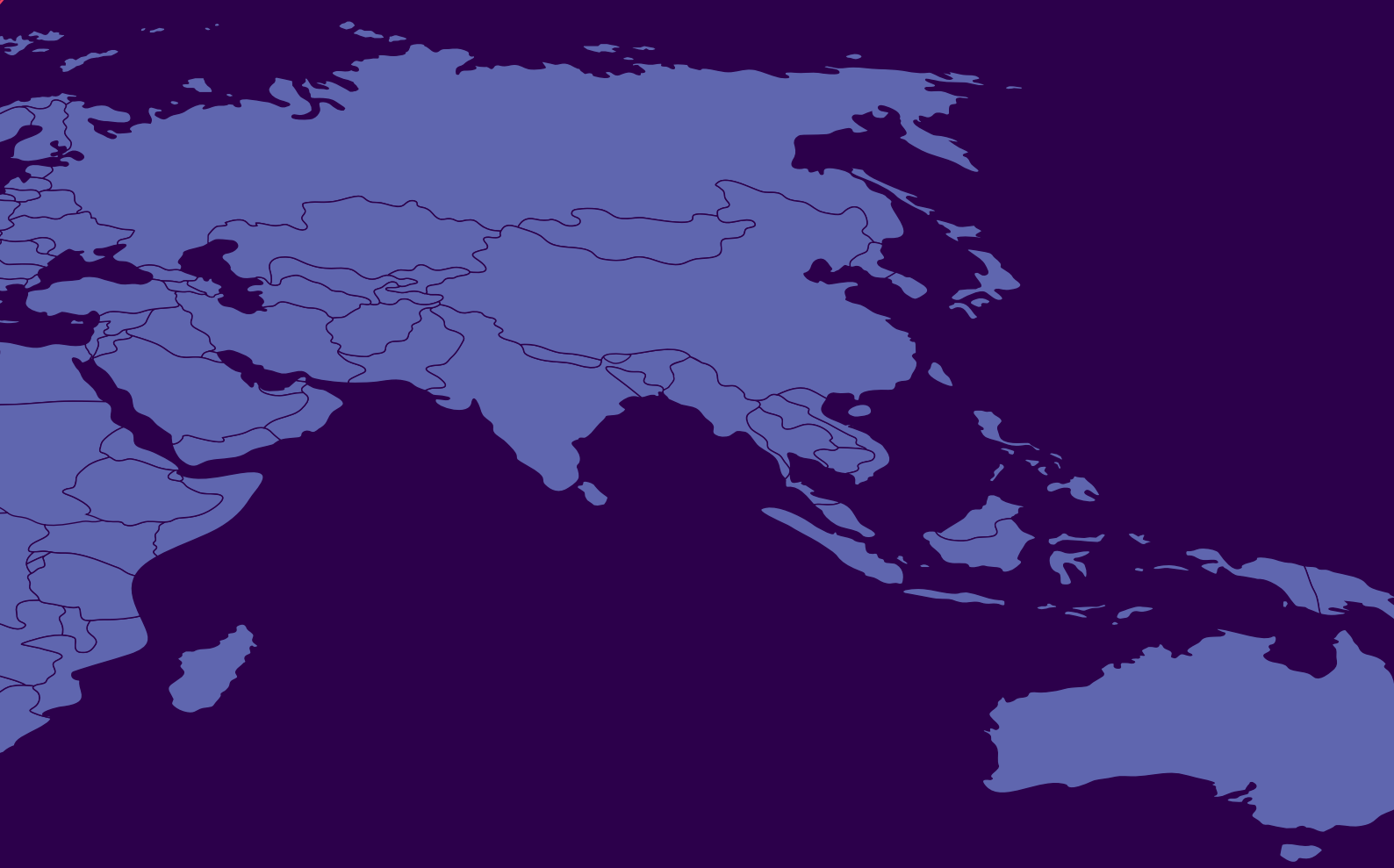
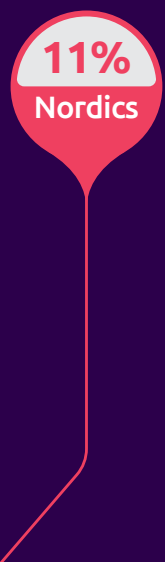
As mentioned above, the B2B market research conducted for the Continuous Testing Report is based on a sample of 500 interviews from enterprises with more than 500 employees (30%), organizations with more than 1,000 employees (27%), organizations with more than 5,000 employees (27%) and companies with more than 10,000 employees (16%).

During the interviews, the research questions asked of each participant were linked to the respondent's job title and the answers he/she provided to previous questions where applicable. For this reason, the base number of respondents for each survey question shown in the graphs is not always the full 500 sample size.

The survey questionnaire was devised by QA and testing experts in Capgemini, in consultation with Coleman Parkes Research. The 24-question survey covered a range of testing and data extraction subjects.

Interviews by region





Methodology

Online survey and some telephone interviews

.....

Audience profile

Senior decision makers from large, enterprise organizations globally

.....

Number of respondents

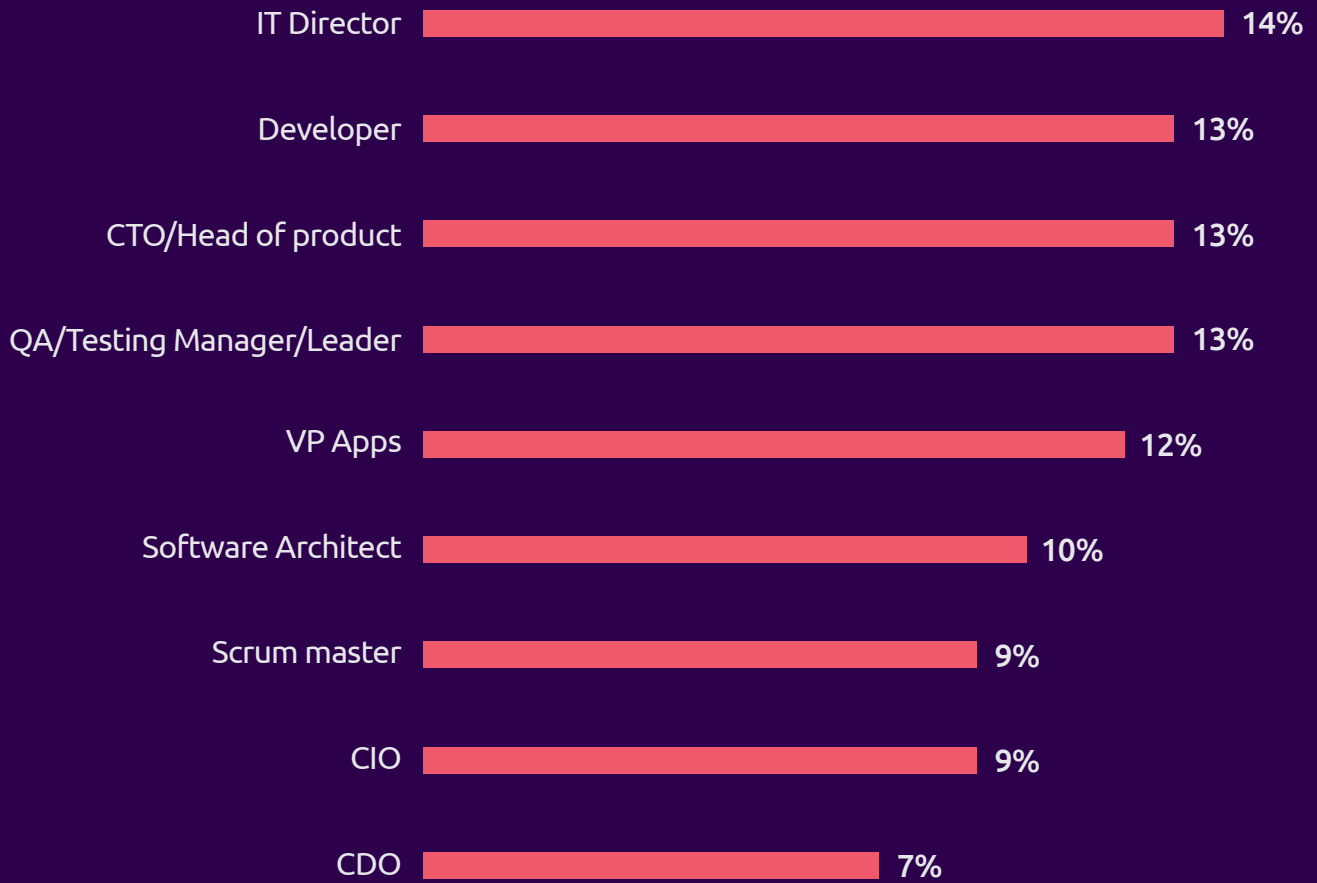
N = 500

.....

Fieldwork dates

October 2019 to Jan 2020

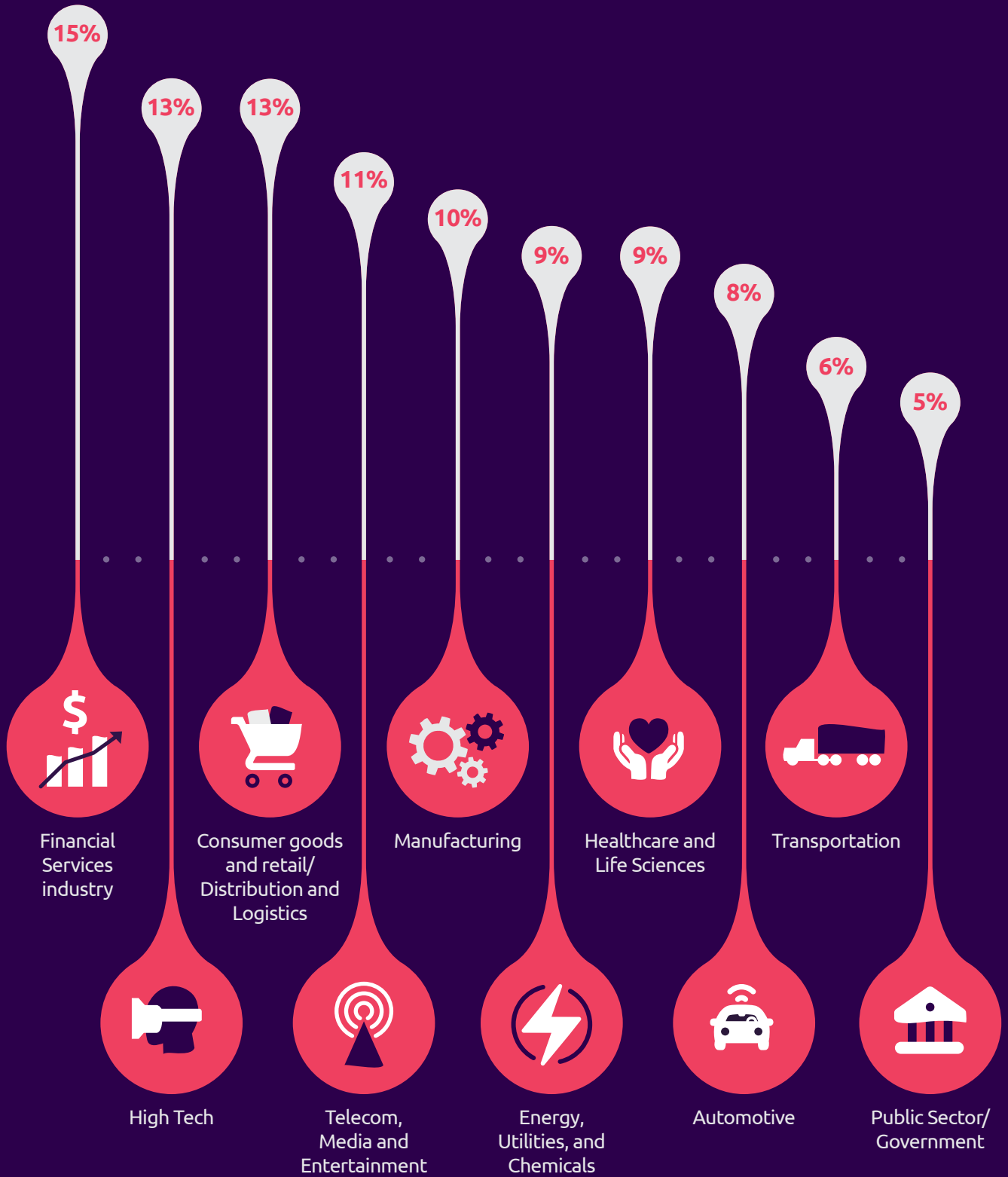
Interviews by job title



Interviews by number of employees



Interviews by sector



Thank you

Capgemini, Sogeti, and Broadcom would like to thank

The 500 IT executives who took part in the research study this year for their time and contribution to the report. In accordance with the UK Market Research Society (MRS) Code of Conduct (under which this survey was carried out) the identity of the participants in the research study and their responses remain confidential and are not available to the sponsors.

All the business leaders and subject matter experts who provided valuable insight into their respective areas of expertise and market experience, including subject-matter experts from Capgemini, Sogeti and Broadcom.

Main Report Authors

Christine Bensten
Deepika Mamnani
Antoine Aymer

Writer

Robert Fenner

Creative Design

Manas Kar

Global Marketers

Archit Revandkar
Anubhav Gulani

Partner Management

Marianne Kantor

Content Proof Reading

Monica Kwiecinski

Market Research

Stephen Saw
Sian Burnage
Ian Parkes
(Coleman Parkes Research) *

*Ian Parkes, CEO and co-founder of Coleman Parkes Research, is a full member of the Market Research Society. All research carried out by Coleman Parkes Research is conducted in compliance with the Code of Conduct and guidelines set out by the MRS in the UK, as well as the legal obligations under the Data Protection Act 1998.

Capgemini, Sogeti and Broadcom, and their respective marks and logos used herein, are trademarks or registered trademarks of their respective companies. All other company, product and service names mentioned are the trademarks of their respective owners and are used herein with no intention of trademark infringement. Rightshore® is a trademark belonging to Capgemini. TMap®, TMap NEXT®, TPI®, and TPI NEXT® are registered trademarks of Sogeti, part of the Capgemini Group.

No part of this document may be reproduced or copied in any form or by any means without written permission from Capgemini and Broadcom.

About the sponsors

About Capgemini and Sogeti

A global leader in consulting, technology services and digital transformation, 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 organizations 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. It is a multicultural company of almost 220,000 team members in more than 40 countries. The Group reported 2019 global revenues of EUR 14.1 billion.

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 organizations implement their digital journeys at speed

Visit us at

www.capgemini.com

www.sogeti.com

About Broadcom

Broadcom delivers Continuous Testing solutions for the global enterprise. Broadcom's Continuous Testing portfolio helps leading companies shift testing left, and right, for continuous quality and superior business outcomes. Release new capabilities with confidence with Continuous Testing from Broadcom. For more information, go to www.ContinuousTesting.com or www.Broadcom.com

Broadcom Inc. (NASDAQ: AVGO) is a global technology leader that designs, develops and supplies a broad range of semiconductor and infrastructure software solutions. Broadcom's category-leading product portfolio serves critical markets including data center, networking, enterprise software, broadband, wireless, storage and industrial. Our solutions include data center networking and storage, enterprise and mainframe software focused on automation, monitoring and security, smartphone components, telecoms and factory automation.



Broadcom

Christine Bensten

Head of Product Marketing,
DevOps and Continuous Testing
christine.bentsen@broadcom.com

Sogeti

Mark Buenen

Global Leader,
Digital Assurance and Quality Engineering,
Capgemini Group
mark.buenen@sogeti.com

Sathish Natarajan

Group Vice President,
Head of Digital Assurance and
Quality Engineering,
Capgemini North America
sathish.natarajan@us.sogeti.com

Antoine Aymer

CTO,
Digital Assurance and Quality Engineering,
Sogeti
antoine.aymer@sogeti.com

Capgemini

Anand Moorthy

Vice President,
Digital Assurance and Quality Engineering,
Financial Services
anand.moorthy@capgemini.com

Sanjeev Deshmukh

Vice President,
Digital Assurance and Quality Engineering,
North America
sanjeev.deshmukh@capgemini.com

Deepika Mamnani

Senior Director,
Digital Assurance and Quality Engineering,
Financial Services
deepika.mamnani@capgemini.com

In association with

