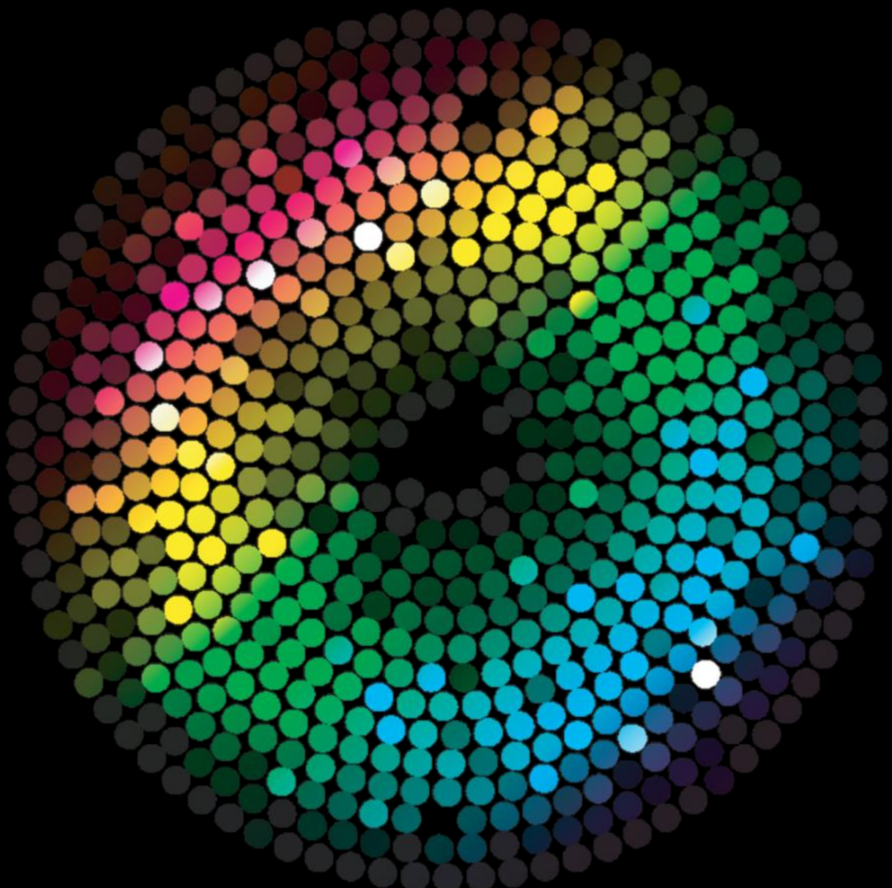
Deloitte.



2021 Quality Engineering Trends Report

Contents

I. Introduction

The pace of new technology adoption in areas like Internet of Things (IoT), blockchain, and artificial intelligence (AI) is driving exciting change in software development while also bringing unprecedented complexity. IT and product leaders eager to capitalize on these new technologies face a slew of quality delivery challenges.

Rolling out new solutions quickly in this environment requires advanced upskilling, tighter coordination within and among teams, and a firm grip on the latest tools and methods available to the quality engineering function – all areas that most organizations are still working on. The good news is that many of the same technologies complicating the software development lifecycle are bringing opportunities for drastic improvement to testing and quality engineering practices.

In this, our inaugural Quality Engineering Trends Report, we look at the key dynamics shaping the future of quality engineering and describe how organizations can better position themselves to capitalize on each one. To do this, we conducted a survey of business leaders across industries – asking them what's top-of-mind for their current quality engineering practices, the trends they are excited about, investments they are making, and the challenges they are facing.

We address the key concerns and considerations that our survey respondents elevated and break down the opportunities within each to improve your quality engineering engine. We are confident that you will find clear, actionable takeaways for your organization.

Rohit Pereira

Quality Engineering Practice Leader

Avneet Chatha

Quality Engineering Delivery Leader

Chaithanya Kolar

Quality Engineering Innovations and Assets Leader

II. Executive Summary

These are unprecedented times. Organizations have been scrambling to understand how to maintain a competitive edge and get their capabilities out to market at the same speed. One thing that has not changed in this environment is the emphasis on quality and risk management.

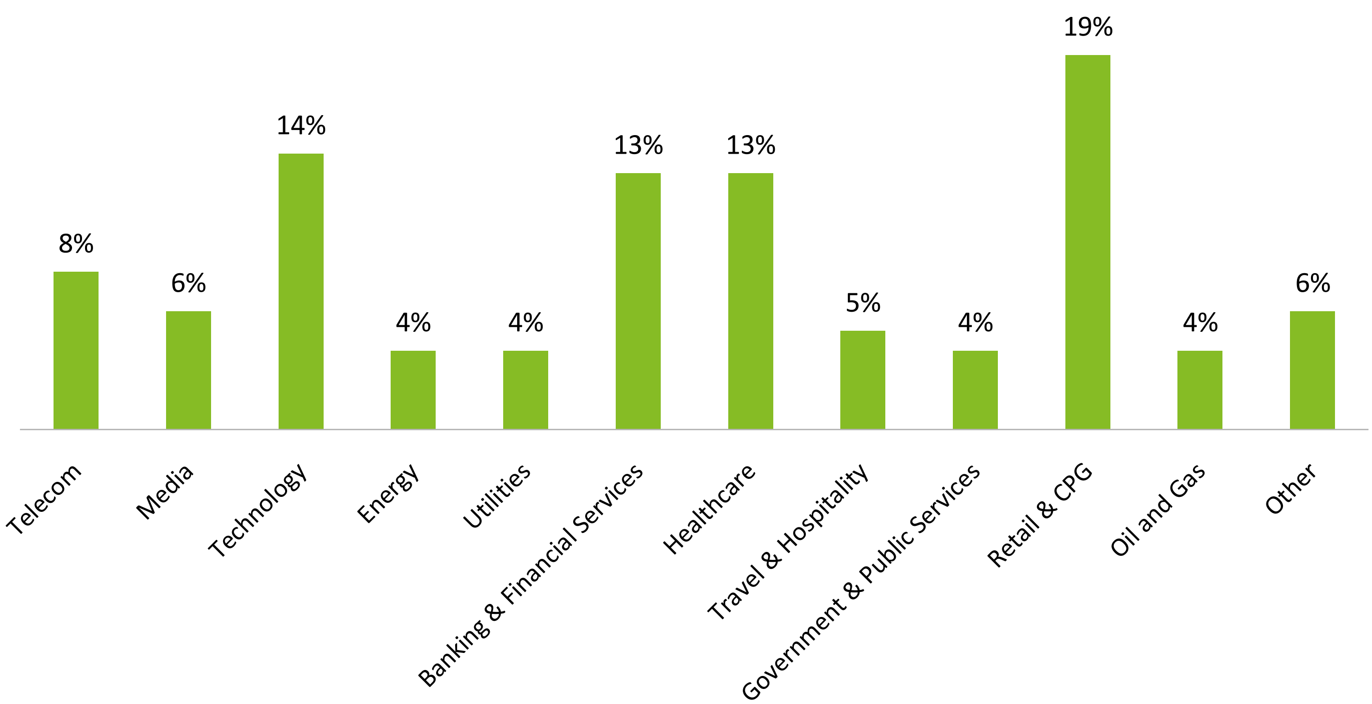
To capture the current mindsets of organizations and their quality engineering approaches, we surveyed 224 senior executives with IT leadership roles across industries (Figure 1). Results from the survey, our experience in the market, and our thought leadership form the recommendations in this report.

Who we interviewed

All of our respondents held senior leadership positions. Among them, 43% were in IT senior leadership, 26% were in the C-suite, 24% were senior directors or directors, and 8% held vice president roles.

Respondents represented organizations with at least $500M in annual revenue – 42% from organizations with $500M to $4.9B in annual revenue, and 58% from organizations with greater than $5B in annual revenue.

Figure 1: Primary industry breakdown of survey respondents



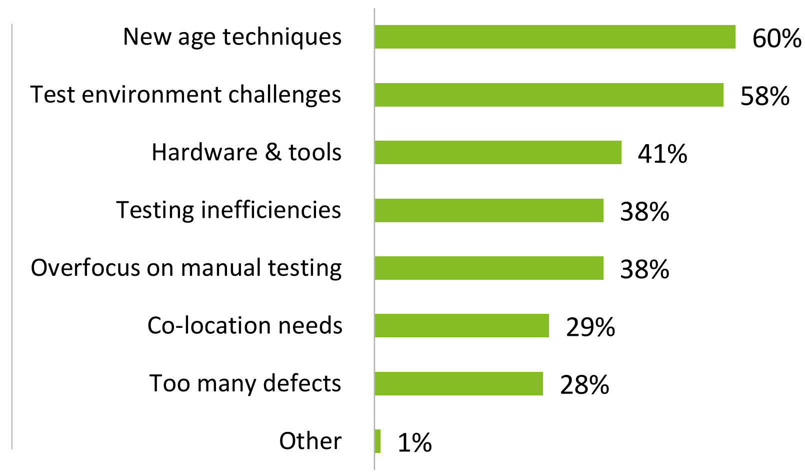
What we heard

While respondents cited a diverse mix of considerations driving their overall testing spend, emerging technologies (60%), mounting test environment challenges (58%), and hardware and tooling needs (41%) rose to the top (Figure 2). A combination of these themes reverberates throughout our findings and recommendations across trend areas.

A picture of the overall ecosystem testing challenges cited by respondents tells a complementary story (Figure 3). Many leaders cited multiple challenges and selections were diverse, but limited automation (56%), competing priorities and processes (51%) and maturity of test processes (50%) were most common. It is apparent that organizations want to take a step back, rethink, and streamline their testing practices in order to make the most of rapidly emerging software development opportunities and their latest investments.

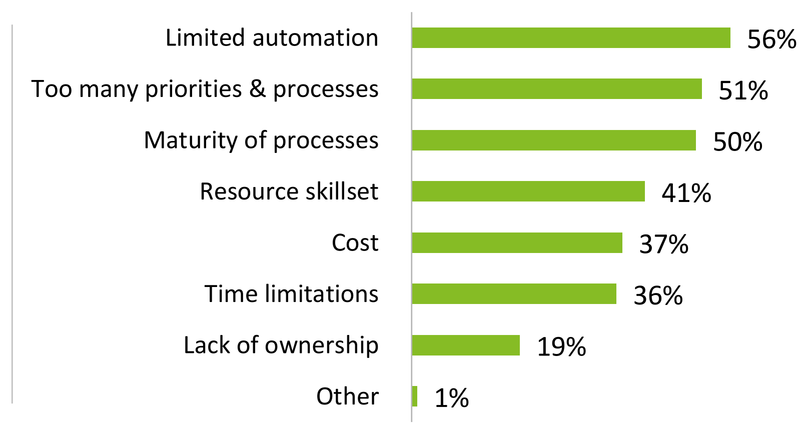
So how can this be achieved? With a clearer picture of the needs of today's IT organization serving as context, we break down the key opportunities within each major trend area impacting the quality engineering function.

Figure 2: Testing spend drivers within the organization



Note: Respondents could select multiple answers.

Figure 3: Key testing challenges in the organization's ecosystem



Note: Respondents could select multiple answers.

What we believe

Our perspectives are divided into 11 chapters – one for our overarching outlook on the future of quality engineering and one for each of the 10 trend areas we see making an outsized impact on quality delivery. In each chapter, we surface key signals from our survey and break them down with a detailed perspective on where things are headed and how opportunities can be seized to make drastic improvements to your quality engineering practices.

Summary Perspective	Chapter	
Overall, **the future of quality engineering** will center around scaling **intelligent automation,** perfecting **continuous testing, securing the enterprise,** and **generating meaningful test data.**	III.	Future of Quality Engineering
Intelligent automation demands some programming knowledge, but **will free up humans to conduct more meaningful and strategic work.**	IV.	Intelligent Automation Has a Human Touch
Internal culture and org structures still need to catch up as organizations scale their Agile, DevOps, and continuous integration and continuous delivery (CI/CD) practices.	V.	New Normals in Agile, DevOps, and CI/CD
Security testing is a top priority and a top challenge for effective cloud migrations and cloud-native development, where choices around service and delivery models are paramount.	VI.	Choose Your Cloud, Test Your Cloud
Chaos engineering can drastically reduce costs associated with downtime but **requires a methodical framework to prioritize failure scenarios.**	VII.	Chaos is a Good Thing
Effective **test data and test environment strategies combine a multitude of emerging technologies and concepts**, and together define an optimally run testing function.	VIII.	Strategic Approaches to TDM and TEM
Resource skills and effective tooling remain significant areas of need in implementing and making the most out of AI and ML solutions.	IX.	Know Your AI Testing Needs
While security, testing infrastructure, and testing automation are atop a long list of challenges for testing IoT applications, **going back to the basics is the best place to start in honing an IoT testing approach.**	X.	IoT Starts with Basics
Crowd testing with public crowds has become an integral part of many organizations' testing practices, but **vendor selection can make all the difference.**	XI.	Wisdom of Crowd Testing
Testing is being reimagined for 5G and multi-access edge computing (MEC) applications, where security, scale, and interoperability create as many development opportunities as they do challenges.	XII.	MEC, 5G, and Possibilities
Permissioned blockchains are a transformational opportunity but **face security testing challenges and skill shortages.**	XIII.	Permissioned Granted, Blockchain

III. Future of Quality Engineering

III. Future of Quality Engineering

In recent years, we have seen the rise of emerging macro technological forces such as core modernization, digital experience, analytics, blockchain, and the business of IT. In order to capitalize on their full potential, organizations should understand how these forces interact and intersect with other emerging technologies rather than viewing them in siloes.

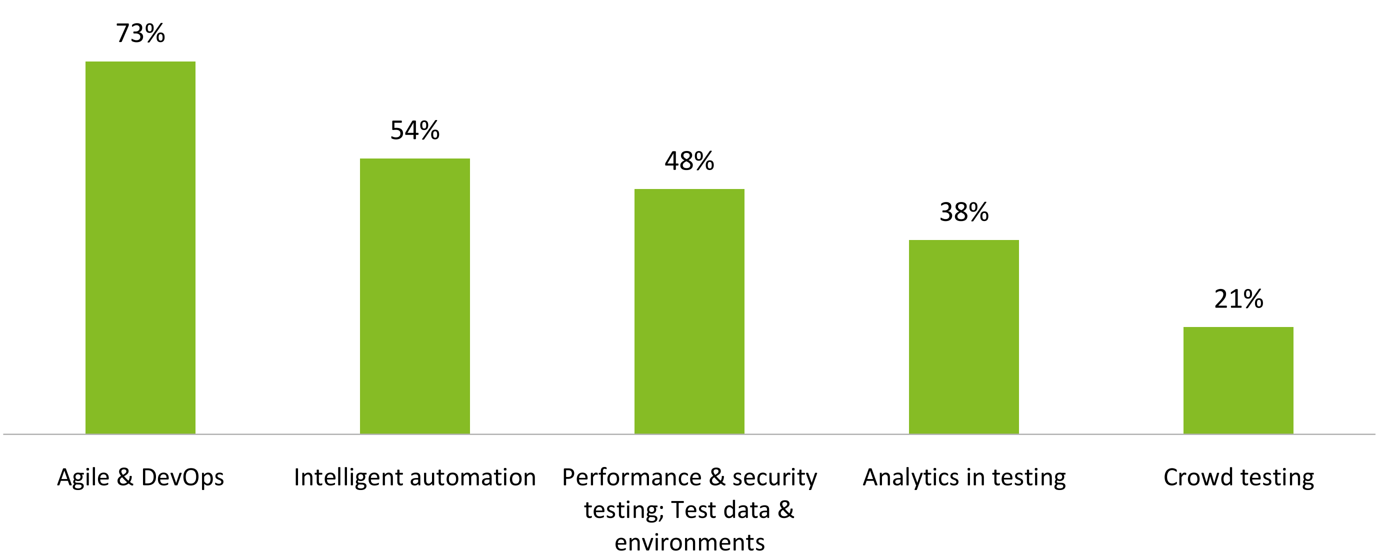
The advent of 5G, for instance, is bringing more highly connected ecosystems to the universe of Internet of Things (IoT) devices. Organizations are also gaining increased access to artificial intelligence (AI) and machine learning (ML), which is encouraging the use of more sophisticated approaches like human-centered design. Organizations are now looking at how human emotions can be read and processed.

Survey results in this year's Quality Engineering Trends Report reflect the breadth of options (and decisions) facing IT and business leaders when it comes to investments in quality and testing trends. Respondents largely indicated they were planning to invest in more than one trend area over the next 1-2 years (Figure 4). Agile and DevOps led the way as a selection from over 70% of respondents, followed by intelligent automation (54%) and performance and security testing, test data and environments (48%) each checked off by roughly half.

Organizations are also exploring non-traditional testing approaches not limited to the confines of a specific team established to validate critical functions. Crowd testing, for instance, differs from traditional methods in that it is carried out by multiple testers dispersed geographically. It is gaining traction through a variety of third-party platforms.

Organizations are exploring how all these trends intersect to create more value, enable new ways to manage technology, and impact the quality engineering function specifically. As organizations continue their path through digital transformation, quality needs to evolve from a supporting to an

Figure 4: Trend areas organizations are likely to invest in over the next 1-2 years



Note: Respondents could select multiple answers.

enabling function by shifting the paradigm away from "*cost of quality*" to "*value of quality.*"

Quality engineering has already evolved considerably and is now considered an integral part of software delivery rather than an afterthought. The late 90's saw the advent of various methodologies like agile testing and exploratory testing when testing was manually done. Then, the new millennium surfaced approaches that shifted focus to automation-driven test execution.

In recent years, we have started to see a gradual mindset change in organizations to shift-left their test planning and execution activities. This has involved engaging testing from early stages of development to identify and mitigate challenges. But automation has still largely been performed in pockets. While test execution has shifted to rely more heavily on automation, other areas such as requirements, design, and even reporting are frequently more manual.

As the external environment gets more complicated, IT leaders are looking for solutions to their environment issues, data challenges, technology upgrade requirements, and increasing need for business process knowledge. They are looking for advanced testing mechanisms to handle evolving technologies like cloud, IoT, blockchain and analytics. But there are significant technological, socio-economic, and financial factors at play. Several are particularly important for organizations to consider as they update approaches in quality engineering.

Rise of machines

While AI and ML have been around for a while, they have largely been untapped until recently. As organizations have developed a better ability to harvest data, so to have they been able to take better advantage of AI and ML to analyze patterns and generate more meaningful insights. Investment in big data has also enabled organizations to access AI and ML algorithms that are deterministic in nature. The value of data has elevated the need for more realistic approaches in identifying and extracting the right data.

As a result, there is a growing need for data scientists within testing teams. We need to position quality engineers to be catalysts for speed, agility, and business performance.

Establishing digital twins

Digital transformation and emerging technologies are continuing to impact quality engineering in every industry. Processes are evolving faster, a wide array of smarter testing tools are emerging, and newer architectures are being built.

In a well-connected ecosystem, digital entities and human testers will work in tandem to achieve elevated results.

Digital twins, or digital replicas of products and processes, hold promise to help human testers keep up and will radically alter the testing function.

In a well-connected ecosystem, digital entities and human testers will work in tandem to achieve elevated results. Digital twins will also be applied beyond the areas of test execution that are traditionally automated to include requirements, design and even reporting.

Oh user, where art thou?

Consumer expectations for what their products should do are more demanding and varying than ever. Technology advancements have turned what was once fantasy into reality. It has become critical, then, for organizations to consider the finer emotional drivers of a user at every touchpoint and micro-interaction with a product. Sophisticated processes that help isolate, analyze, and contextualize specific user actions will go a long way in evolving the user experience to keep up.

Optimizing the user experience also requires ongoing finetuning to accommodate shifting needs. This has spurred the importance of more intelligent quality engineering techniques that include production-driven data considerations accounting for seasonal and cyclical variations.

Evolution of automation

Approaches to test automation are also changing. While the shift from manual to automated testing took off several years ago, there are still gaps in automation driven by limitations of existing tools and processes. To best seize the automation opportunity, organizations will need to move beyond traditional techniques and toward new approaches.

Intelligent automation will have an outsized impact, bringing tools with self-healing capabilities that eliminate one of the major pain-points in automation – script maintenance (Figure 5). Touchless automation is also gaining traction as it enables both testers and business users to automate test cases without worrying about coding. It drives faster results and reduces the time required to understand underlying code.

It's no surprise that over half of survey respondents highlighted intelligent automation as a key area to invest in over the next 1-2 years.

As organizations revisit their investment allocations, they are slowly transitioning towards a more seamless and connected testing lifecycle approach using digital entities that link one phase to another from requirements management through reporting.

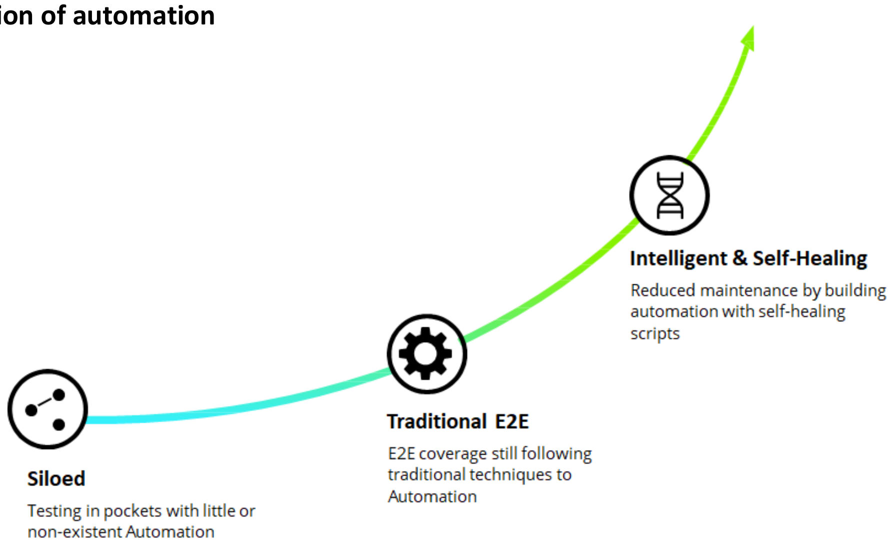
Rinse 'n' repeat – continuous testing and integration

In today's Agile environment, there is greater emphasis on in-sprint development and execution followed by continuous integration with the broader pipeline. All aspects of software development, testing, and deployment need to adapt by moving away from traditional waterfall or hybrid approaches.

As release cycles get shorter and shorter to keep up with evolving user needs, new DevOps practices, tools, and processes are becoming an essential part of the software development lifecycle. Continuous testing is one DevOps trend that has become a must for development organizations looking to win the continuous delivery game.

Gartner describes the aim of continuous testing as to "obtain immediate feedback on the business risks associated with a software release candidate."[1] This is quite a change from the traditional approach to testing

Figure 5: The evolution of automation



Source: Deloitte analysis.

and QA. Many organizations now strive for continuous testing as table stakes and are breaking away from the old days of performing testing at the end of the development cycle.

All aspects of software development, testing, and deployment need to adapt by moving away from traditional waterfall or hybrid approaches.

Securing the enterprise

For many organizations, cloud's takeover of the enterprise is well underway and for some it is complete. Over 90% of organizations have at least a portion of their IT environment in the cloud today, and they aren't putting on the brakes.[2] In fact, spending on public cloud and private cloud infrastructure increased roughly 50% and 7%, respectively, in 2020, and cloud investments are expected to continue increasing as proportion of overall IT spend.[3]

Cloud has forced the reimagining of some tried-and-true roles within testing organizations, particularly around security testing. As applications and data move off premises to be hosted elsewhere, data security has become a vital component of testing on the cloud. Data security testing evaluates a system's ability to protect data and resources from vulnerabilities like hackers, viruses, unauthorized login attempts and more.

Security test coverage needs to be clearly defined, with expectations for differences in testing data at rest vs. testing data in motion made explicit. Cybersecurity testing should also be conducted to secure code reviews identifying weaknesses that may not typically be detected.

Manage and maintain "the new oil"

The generation of clean test data is critical to maximizing the value of testing. However, this is an area where many organizations are lacking. Struggles are driven by two key factors:

- **Limited portability of test data** across different test environments complicates the effort and limits enthusiasm for investing additional time and resources.
- **Lack of permanent test environments** for testing teams results in more disparate ad-hoc efforts as more organizations tend to perform testing in shared environments.

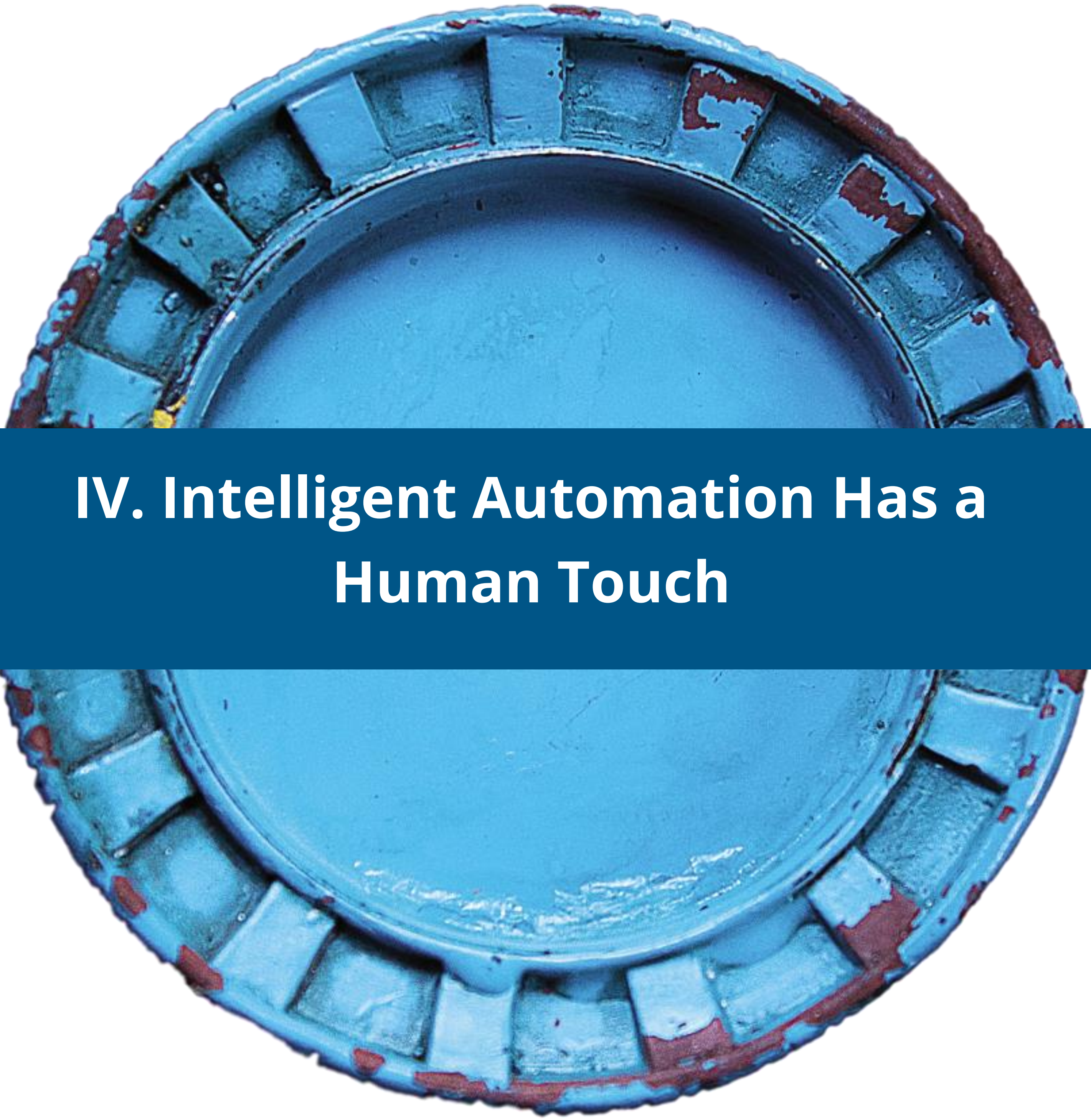
Despite these challenges, organizations are realizing that meaningful test data is crucial for effective validation of critical business scenarios prior to go-live. With challenges involved in combing through millions of records to identify data subsets for testing, effective methods for obtaining the right data sets are at a premium.

Organizations should explore a variety of methods to improve their data generation and porting approaches, including data as a service (DaaS) options.

The takeaway

The advent of disruptive forces like **5G and IoT have unlocked myriad use cases for AI and ML**, propelling them well beyond mere buzzwords. To harness and keep up with emerging opportunities, **quality needs to evolve from a supporting to an enabling function by shifting the paradigm from "cost of quality" to "value of quality."** Growth opportunities exist in **digital twins**, **intelligent automation**, **CI/CD**, **cloud security**, and **test data generation**. As organizations race to apply new techniques and stay ahead of the curve, they should be guarded against both internal and external forces.

IV. Intelligent Automation Has a Human Touch



IV. Intelligent Automation Has a Human Touch

We have touched on how trends in areas like mobile, cloud, microservices, AI and ML, and blockchain have necessitated an update to traditional testing methods. But the adoption of Agile has also accelerated the software development lifecycle, making it even more challenging for testing to keep pace. And most organizations still rely on manual testing. Continuous development requires continuous testing, which can only be achieved through automation.

Traditionally, over half of the testing effort goes towards test execution. Going back a decade, organizations focused primarily on automating test execution cycles. There was little focus on automating other phases like requirement analysis or test design. Now, advancements in AI and ML unlock opportunities to automate the other phases. It's time that organizations adopt digital testers, or bots, powered by AI and ML to automate a majority of their test activities.

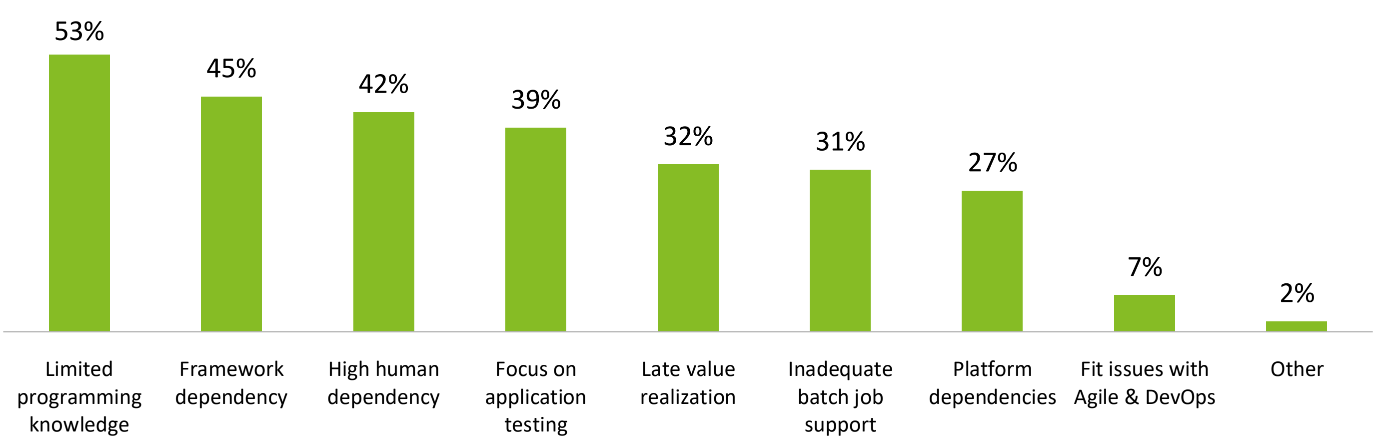
Test process evolution and its challenges

The evolution of testing over the past couple decades helps illuminate some of the key challenges that have emerged.

Let's take a look at some of the most significant:

- **Limited automation in the test life cycle and higher maintenance effort**: Test automation has been around for 20+ years and has been limited primarily to performing regression testing of applications. This has matured over the years, but still involves higher effort in continuous maintenance.
- **Evolving technology**: Constantly evolving digital technologies create a need for continuous integration and continuous delivery (CI/CD) and faster testing feedback. Test processes need to change and shift-left with the help of APIs, service virtualization, and faster automation.
- **Speed and time to market**: With the need for products to be launched faster, testing always lags behind the speed of development. There is opportunity to automate repeatable testing tasks.
- **Test governance**: Most organizations today have varying governance and reporting mechanisms, without consistent and uniform metrics/KPIs across business units. Current processes lack real-time and insightful reporting which would enable improvements in test automation.

Figure 6: Challenges organizations see with current test automation tools and solutions



Note: Respondents could select multiple answers.

Challenges involved with evolving technology on the testing front were echoed by our survey respondents. The need for programing knowledge (53%), framework dependency (45%), and a high level of human intervention required for automation execution (42%) were the top three challenges identified with current test automation tools and solutions (Figure 6).

Enter, digital testers

There are a variety of repetitive and manual tasks across the testing lifecycle that eat up time from human testers – including requirement analysis and test design, maintenance of automated test scripts, and test reporting. Many of these tasks can now be automated with the help of digital testers powered by AI and ML. Digital testers are an intelligent automation technology that aims to reduce human intervention in the software testing lifecycle.

We are likely to see the future quality engineering workforce heavily supported by digital testers, where bots carry out a majority of the activities in the testing lifecycle and strategic tasks are handled by human testers. Digital testers are part of the broader influence AI and ML are having on test automation.

Gartner predicts that by 2022, 40% of application development projects will use AI-enabled test set optimizers for building, optimizing, and running test assets.[4]

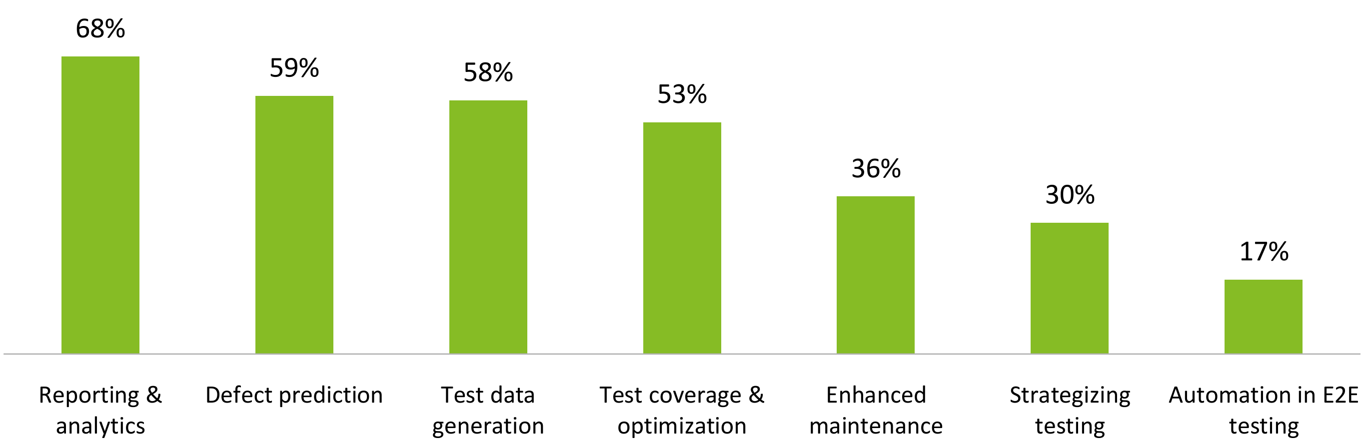
We are likely to see the future quality engineering workforce heavily supported by digital testers.

Survey respondents indicated a wide variety of expectations for the impact of cognitive technologies on quality engineering practices (Figure 7). Better reporting & analytics (68%), defect prediction (59%), test data generation (58%), and test coverage (53%) were all popular expectations. Digital testers can be leveraged for many of these use cases and can be embedded throughout the various stages of the lifecycle. Let's look at how in each stage.

Requirement analysis

Requirement analysis and test design with AI techniques and context awareness algorithms can be

Figure 7: Where organizations envision the largest impact of cognitive technologies on quality engineering



Note: Respondents could select multiple answers.

embedded in digital testers. This process can become customer-focused and insight-driven:

- **User stories and requirements**: Develop automation that integrates with test management tools to extract requirements, analyze them, and design manual test scenarios and cases.
- **End-user sentiment analysis**: Use automation to conduct reviews of end-user feedback survey data, production tickets, and social media feedback with ML/NLP techniques to identify test cases suite.
- **Intelligent test planning**: Build workplans and tasks autonomously based on requirements and predefined tasks while reporting progress to a visual dashboard.

Test design

- **Intelligent regression coverage**: Based on the requirements in scope, use AI and ML to analyze the existing test suite and identify impacted test cases, gaps for creating new test cases, and test cases that are no longer needed. It can also carry out a code coverage analysis and identify impacted areas that need to be regressed further.
- **Autonomous execution and self-healing framework**: Using cognitive abilities and ML, identify changing web object properties upfront and auto correct the test scripts in runtime for uninterrupted test execution.
- **Automated API test generation and service virtualization**: Monitor and analyze background traffic to discover patterns and identify relationships for building comprehensive automated APIs.

Test data, execution, and defect management

Such AI- and ML-infused automation can scan through an entire test set to be executed and identify data requirements to execute test cases while deidentifying PHI and PII for client confidentiality purposes.

Once deployed, it can also actively listen to changes in the application such as environment configurations, deployments, and defects. It can also identify and prioritize the set of test cases for execution, create test data, and run autonomously to provide test results. As the test set executes, defects identified can be logged directly in a defect management tool and assigned to the right developer triggered through an email.

Through continuous integration powered with AI and ML, digital testers can learn from past experiences and get smarter over time.

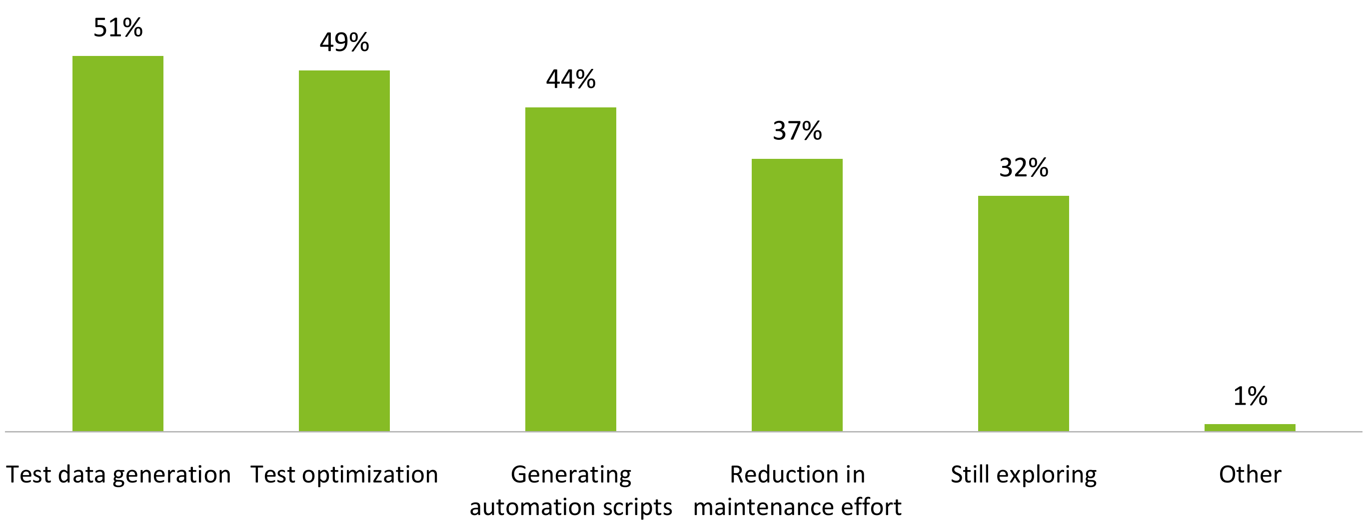
Test reporting and metrics

An out-of-the-box reporting dashboard coupled with automated reporting is another key capability of intelligent automation. Real-time dashboard reporting empowers quality engineering organizations by providing insights, making recommendations, and standardizing and optimizing reporting. Key capabilities include:

- **Integrated reporting hub and customized reporting**: Apart from providing inbuilt reports and dashboards, we should be able to automate the ability to create custom reports and charts through a real-time test report generator.
- **Application failure insights**: We can continuously analyze test reporting and other parameters, providing meaningful insights such as usage and failure patterns, typical system behavior, and anomalies.
- **Chatbot-driven status updates**: Automated status reports can free up time for test leads to focus on test strategy, architecture, and management activities while increasing transparency and real-time reporting.

Among respondents already using cognitive technologies in quality engineering, more than half are using them to generate test data (Figure 8). A lower proportion of them (37%) are using them to reduce the maintenance effort, suggesting there is room to grow here. Moreover, 32% expressed they were still in exploratory stages.

Figure 8: Where organizations are applying cognitive technologies into quality engineering & software testing

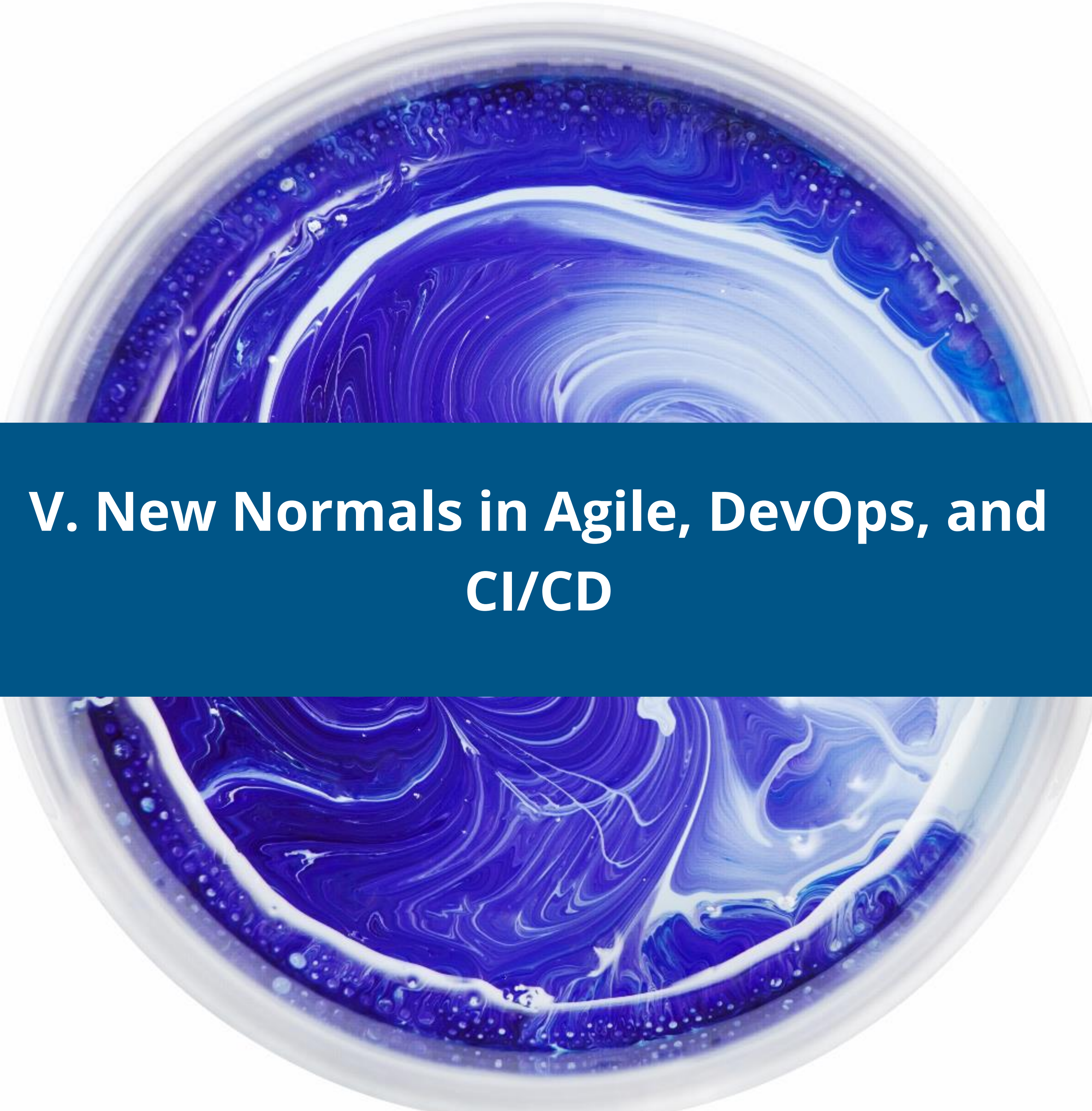


Note: Respondents could select multiple answers.

The takeaway

A rapidly shifting digital landscape, widespread adoption of Agile as the primary development methodology, and the need for faster time-to-market have forced software development organizations to think beyond traditional testing methodologies. **Many repetitive tasks in the testing lifecycle** – including those within requirement analysis, test design, and execution – **can now be automated** with the help of digital testers driven by AI and ML. We are likely to see **the future quality engineering workforce heavily supported by digital testers, where bots carry out a majority of testing lifecycle activities and strategic tasks are handled by human testers**.

V. New Normals in Agile, DevOps, and CI/CD



V. New Normals in Agile, DevOps, and CI/CD

Agile, DevOps, and CI/CD are the key pillars of modern software development processes. Organizations of all sizes looking to adapt their software delivery to thrive in today's fast-changing environment should have these tenets at their core.

"Inspection to improve quality is too late, ineffective, costly. Quality comes not from inspection, but from the improvement of the production process."

– Dr. W. Edwards Deming[5]

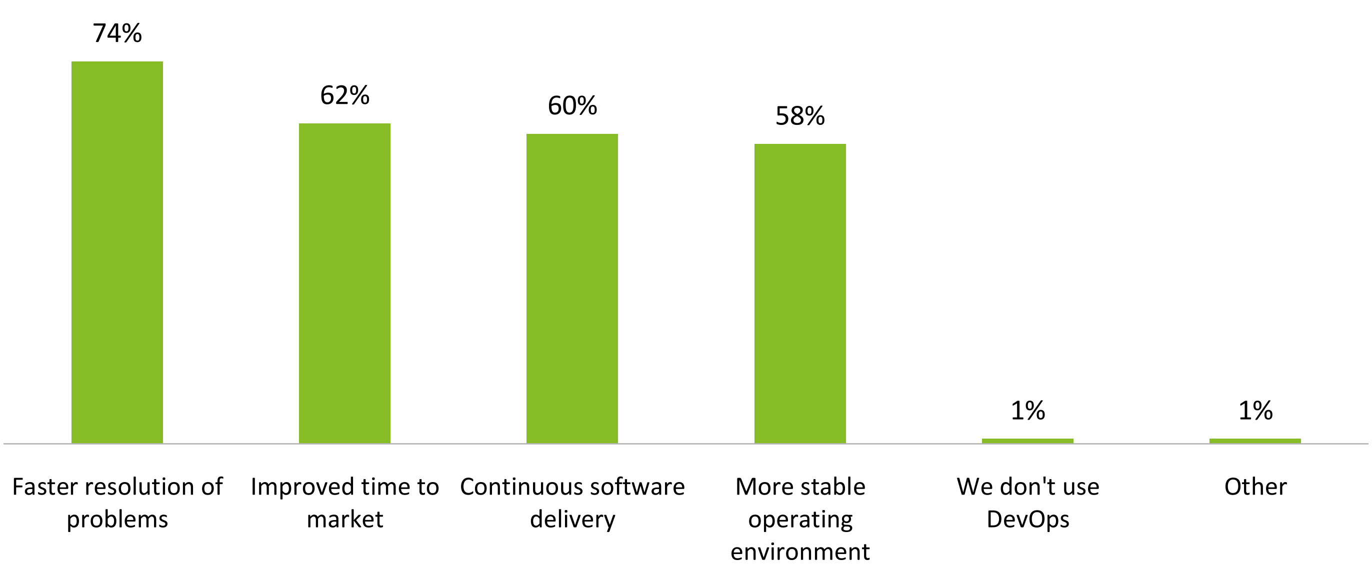
As organizations adopt new methodologies, it is also vital to fully integrate quality engineering into the development life cycle. It should be embedded into Agile and DevOps processes in order to improve value sooner and achieve greater customer satisfaction.

Case for change

Ubiquitous success stories of Agile, DevOps and CI/CD implementations across industries are driving the appetite to implement these disciplines. Gartner predicts by 2023, 75% of organizations will customize Agile practices for product development.[6] It's also telling that our survey respondents indicated Agile and DevOps as the top quality engineering trend of investment focus over the next 1-2 years (Figure 4).

There is clear indication of a shift in favor of Agile and DevOps together with the need for quality engineering practices such as shift-left approaches and automated continuous testing. It is imperative that organizations reassess their Agile and DevOps processes to maximize testing effectiveness and refine their automation strategies to enable true continuous testing.

Figure 9: Benefits organizations have realized through the implementation of DevOps



Note: Respondents could select multiple answers.

Agile, DevOps, and CI/CD are evolving and expanding fast

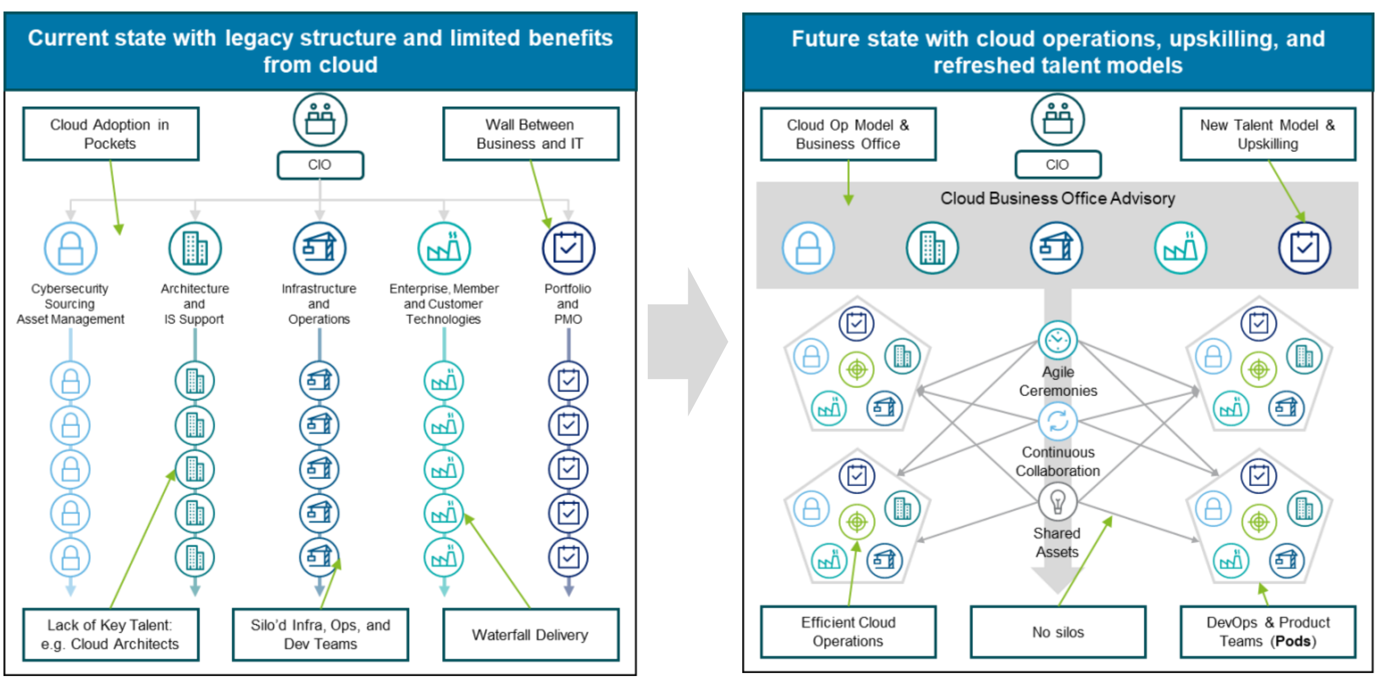
Agile and DevOps unite business leaders, developers, and IT operations staff in cross-functional teams to transform software delivery by streamlining high levels of technology automation and collaboration. Our survey respondents indicated a balanced stable of benefits realized from DevOps (Figure 9), with faster resolution of problems leading the way (74%). Improved time to market (62%), continuous software delivery (60%), and a more stable operating environment (58%) also surfaced highly, and 99% of respondents indicated they were using DevOps. At scale, DevOps at helps companies significantly reduce software defects and get products to market faster. This translates into sustainable competitive advantage.

20 years since the creation of the Agile Manifesto, Agile practices are continuing to evolve and proliferate throughout functions and industries. The delivery model has expanded beyond the software development and IT departments to areas like Agile auditing, Agile budgeting, and Agile human resources. Industries from manufacturing to retail to energy are investing actively.

CI/CD is now more than just an approach to developing and delivering software. It has become an integral and inseparable component of deploying cloud-native applications, which become more robust when their development is integrated with CI/CD methods. Companies like Amazon and Netflix, for instance, are using DevOps, CI/CD processes, and automation at every stage to deploy code thousands of times a day.

Traditional software delivery methods are not well equipped to deliver at the speed of cloud. Organizations now want scalable applications that run on private, public, and hybrid cloud infrastructure utilizing containers, microservices, service meshes, and virtual machines. Upskilling and refreshing talent models optimized for Agile, DevOps, and CI/CD are a critical step (Figure 10).

Figure 10: Transition from legacy state to a cloud-ready organization



Source: Deloitte analysis.

Pods are one model gaining traction across organizations (Figure 10). Pods are self-contained, self-organizing teams equipped with a variety of competencies that completely own and manage the iterative delivery of a product. When staffed with people with specialized and complementary skills, pods can greatly enhance decision making, improve product focus, and build, test, and release a product faster.

Quality engineering and the art of being Agile

Quality engineering teams are constantly challenged to complete testing activities in less time with less human resources. When project schedules crash, the scope and duration of testing activities are often the first to be curtailed, which can impact quality. Yet expectations for high-performing, scalable, secure, and functional systems make high quality non-negotiable. Agile and DevOps practices can help alleviate this tension but seeing their upside doesn't mean it's easy to get there. Our survey respondents indicated they still face a slew of testing-related challenges in trying to embrace Agile and DevOps (Figure 11), with organizational culture (56%), the need for technically competent resources (46%), and challenges with continuous testing approaches (46%) leading the way.

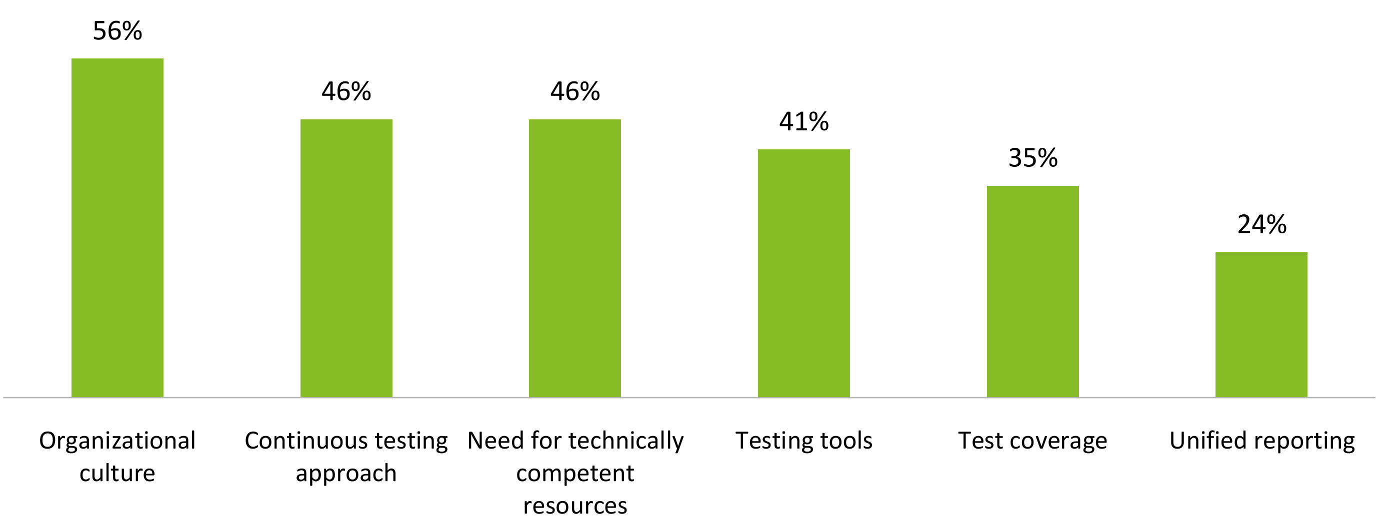
There are a variety of techniques that can help in better managing Agile testing objectives, test schedule constraints, and stringent performance requirements. Let's take a look at them.

Key techniques for Agile testing

In-sprint automation is an integral part of the Agile software development process that can help with short development cycles, frequent changes to requirements and the need to plan and execute tests within a small window of time. The objective here is to develop and execute automation test scripts in parallel to application development within the same sprint. Upon completion of the sprint, automated tests then become a part of the regression test suite.

The TestOps (shift-right) methodology promotes more testing in pre-release and post-release phases of the application life cycle (i.e., testing in production). The idea is for enterprises to define acceptable levels of quality to ensure faster time to market, and then leverage fast remediation

Figure 11: Testing challenges organizations have faced with DevOps



Note: Respondents could select multiple answers.

approaches to address defects or rollback changes when problems are detected.

Test-driven development (TDD) is an evolutionary approach to development that involves writing unit tests before the development of actual software. This approach helps developers refactor the code in order to pass the test, resulting in minimal development effort primarily focused on fulfilling the test requirements. As tests are developed upfront, time allocated for regression testing is also significantly reduced.

Behavior-driven development (BDD) encourages upfront communication between project stakeholders, so all members understand each feature prior to the development process. In BDD, the testers, developers, and product owners create "scenarios," which facilitate communication focused on examples. BDD helps foster strong collaboration between business and technical teams and improves visibility of a project's progression. Using BDD, stakeholders can eliminate ambiguity throughout the product development lifecycle beginning from user stories creation through successful testing and deployment of software.

A/B testing (also referred to as split testing) involves showing two variants of the same web page or mobile screen to different segments of website visitors simultaneously, then comparing which variant drives more conversions. Typically, in A/B testing, the variant that yields higher conversions wins, and that variant can help optimize applications for better results. A/B testing enables early feedback to deliver minimum viable products that end-users prefer. This testing method enables organizations to construct hypotheses, then learn and refine their deployment strategy for certain user experience elements as they go.

API testing helps validate the functionality, reliability, performance, and security of APIs. Instead of relying on standard user inputs and outputs to test application logic, API tests are performed at the message layer without the use of a Graphical User Interface (GUI). This helps accomplish application logic validation in scenarios where GUI testing is not possible due to frequent changes or short release cycles. API Testing covers many aspects of testing including functional testing, security testing, and load testing.

Dogfooding is internal beta testing a product to iterate and refine it before launch. The idea is to put product creators and employees in users' shoes for a first-hand experience of the issues or pain points a customer would face in the real-world. This technique improves quality and helps prevent unfavorable market reaction to a pre-maturely released product. It has become common practice across organizations involved in software development, manufacturing, and other industries. It helps increase overall UX awareness across an organization and can save big on costs (e.g., potential warranties).

The takeaway

Agile, DevOps, and CI/CD methodologies foster collaboration between development, testing, and business teams. Their core objective is to **accelerate software development, improve product quality, and automate the software delivery process**. This, in turn, helps organizations respond better to rapidly changing market conditions, attain competitive advantage, and drive growth. Specific Agile techniques to consider in **support of these goals** include **in-sprint automation**, **TestOps (shift-right)**, **test-driven development (TDD)**, **behavior-driven development (BDD)**, **A/B testing**, **API testing**, and **dogfooding**.

VI. Choose Your Cloud, Test Your Cloud

VI. Choose Your Cloud, Test Your Cloud

Organizational excellence requires a balanced approach of maintaining high availability, business continuity, business agility, and scalability. Cloud supports all of these pillars while being cost effective.

The growth in cloud adoption will continue to be dramatic. Gartner forecasts that worldwide end-user spending on cloud services will grow to $305B in 2021, up 18.4% from 2020. Furthermore, the proportion of global IT spending on cloud transformation is expected to encompass 14.2% of total enterprise IT spending by 2024, compared to 9.1% in 2020.[7]

We asked our survey respondents about their cloud migration plans (Figure 12). 60% of respondents already have at least 11% of their applications in the cloud, with 25% having at least 50% in the cloud. Overall, 98% of respondents are planning to use the cloud. With this overwhelming enthusiasm to adopt cloud technologies, the question becomes, "how can we effectively move to the cloud while mitigating business risk?"

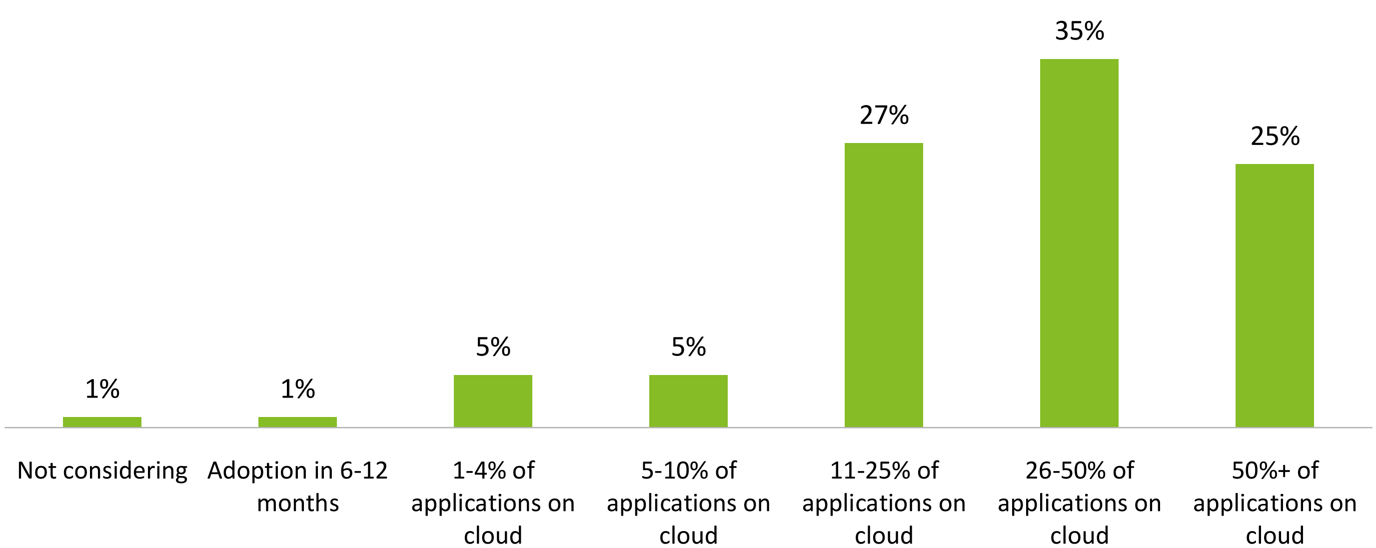
Quality engineering is one the most pivotal factors in making the most of cloud investments and achieving cloud excellence. Cloud has helped adapt and accelerate quality engineering capabilities through tools that allow for more efficient and continuous testing. But with an array of diverse cloud deployment models, quality engineering in the cloud is a complex process requiring well-designed approaches. Integrating existing tools with CloudOps and automation techniques will play a key role.

Becoming cloud-ready requires a well-planned migration strategy coupled with an intelligent test strategy. To guide your cloud transformation, we examine here how to create robust testing strategies catering to various cloud adoption methodologies.

Cloud transformation models

Various service and deployment models have emerged to meet specific needs of users (Figure 13). A cloud deployment model is a configuration of environment declaring how cloud services are available to users. A

Figure 12: Extent to which organizations are moving to cloud



cloud service model represents a specific pre-packaged combination of various IT resources from a cloud provider. Different combinations of these provide varying levels of control, flexibility, and management.

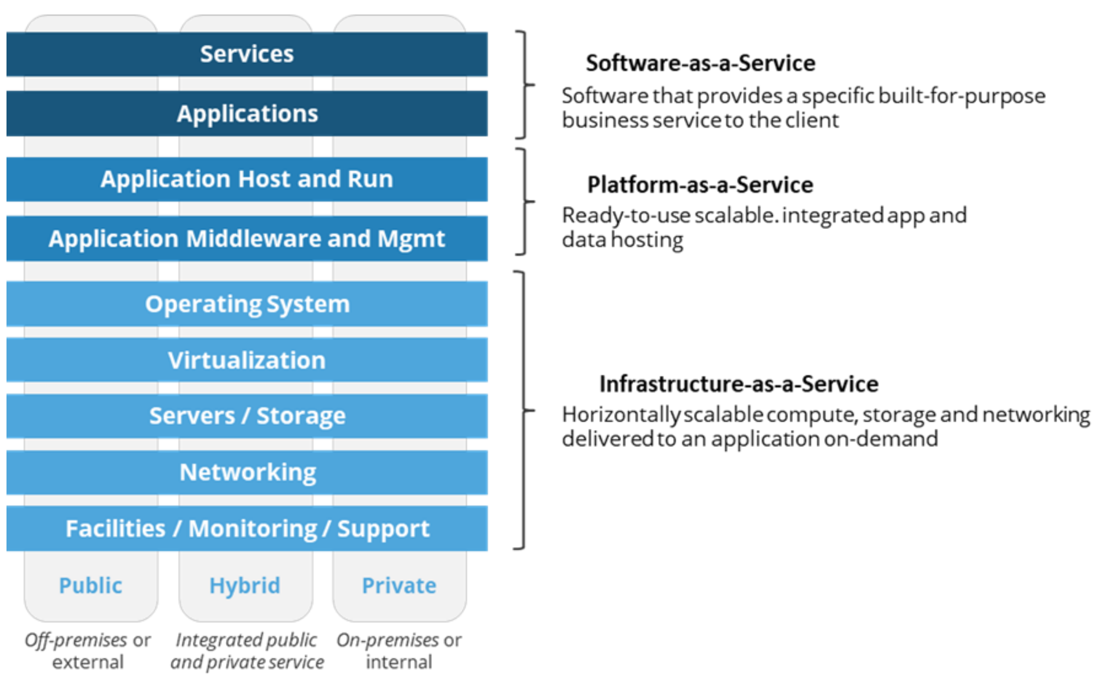
Service models (usually provided through subscription):

- **Infrastructure as a service (IaaS)** enables the customer to install traditional applications on servers offered by cloud providers expecting the customers to manage all aspects of the application software and hardware.
- **Platform as a service (PaaS)** enables the vendor to manage the hardware deployment, software installation, and configuration while the customers manage, tune, and optimize the software.
- **Software as a service (SaaS)** allows users to access the applications that are installed on the remote devices via APIs or web.

Deployment models:

- **Public cloud** provides computing and storage resources to customers over the internet. It is the most widely used cloud service eliminating the cost involved in hardware built and maintenance.
- **Private cloud** provides superior levels of security and control to organizations. This model is specifically dedicated for a single organization to be able to configure and manage the environment based on their specific business needs.
- **Hybrid cloud** helps to combine both public and private cloud models, hence allowing two platforms to seamlessly interact. This model is highly recommended for critical businesses balancing big data analytics with strict data privacy constraints.
- **Community cloud** is a collaborative, multi-tenant platform that can be used by several unique organizations, sharing the same applications.

Figure 13: Cloud service and deployment models



Source: Deloitte analysis.

Cloud's impact on quality engineering

Organizations need to build and maintain in-house testing facilities that can mimic the real-time environments and help manage the business risk associated with migrating live applications into the cloud. Testing in the cloud leverages cloud computing infrastructure, reducing the unit cost of computing while increasing testing effectiveness.

Cloud testing has emerged as a game-changer in the field of quality engineering – leveraging the combination of emerging technologies such as CI/CD and continuous testing, test automation accelerators, and intelligent test strategies. The advancement of cloud computing and AI and ML together has accelerated the quality engineering space.

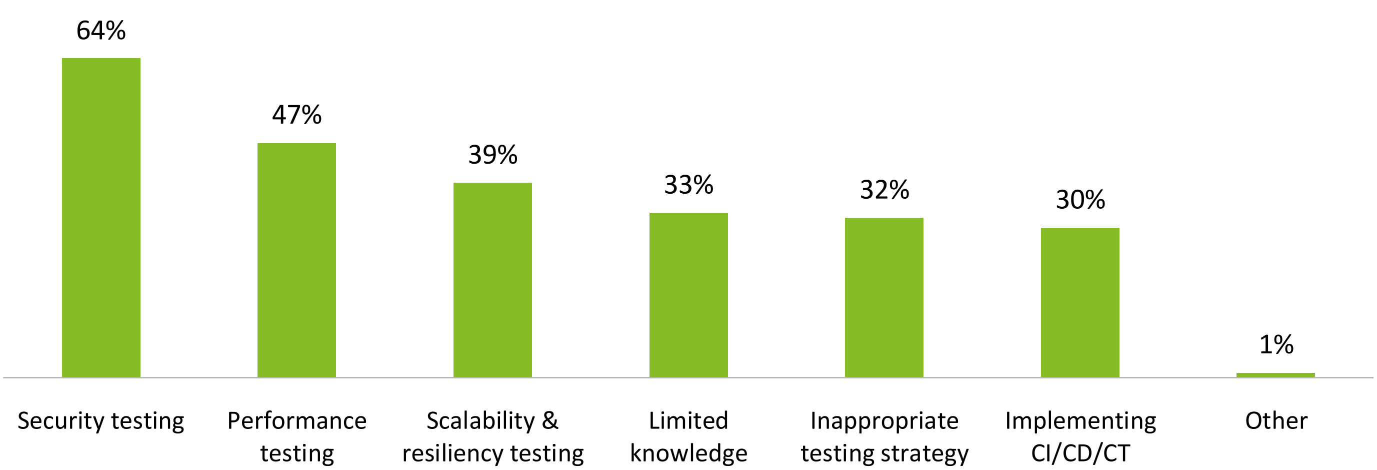
In moving to the cloud, it's important to apply the latest quality engineering practices to cloud infrastructure testing. But effective testing is not easy. Our survey respondents highlighted security testing as the top challenge by far (64%) when it comes to testing applications that have cloud infrastructure (Figure 14). Performance testing (47%) and scalability and resiliency testing (39%) were other top challenges.

Effectively overcoming these testing challenges requires tailoring different testing types to be cloud centric. High-volume load and stress testing are critical. Application latency and scalability also need to be tested rigorously – monitoring data integrity, user privacy, and all security constraints. Re-structuring the reliability and resiliency approach with chaos engineering concepts will also help in addressing system challenges in a control environment.

Continuous testing is critical for ensuring high application quality in the cloud. It is often implemented as a set of automated regression, performance, and other tests executed with CI/CD DevOps practices and container services. Transforming legacy test data pipeline transformation to cloud can also be supported by AI and ML for synthetic test data generation.

Automation has been playing a key role in effective quality engineering. Integrating existing tools with CloudOps and other technologies can help make the transformation journey even more seamless.

Figure 14: Challenges organizations typically see in testing applications on cloud infrastructure



Note: Respondents could select multiple answers.

Importance of testing strategies for cloud transformation

A well-defined cloud testing strategy can help organizations boost its competitiveness and helps reduce the cost of testing without adversely impacting business critical applications. It is important for organizations to think bi-directionally while designing test approaches, both for migration of live applications as well as building new cloud native applications using cloud infrastructure.

Testing strategies for cloud migration

When migrating to a cloud environment, organizations should map out existing on-premise environments, understanding the possibilities offered by their cloud providers and considering what shape each application should take in the cloud. A successful migration strategy will maximize value from cloud infrastructure while minimizing migration time, effort, cost, and risk.

Organizations can leverage a mix of 6 fundamental migration strategies – rehost, replatform, refactor, repurchase, retire, and retain (Figure 16).

Testing strategies for cloud-native applications

Cloud-native technology enables organizations to tap into the elasticity and agility needed to gain competitive advantage. Cloud-native applications fully exploit the advantages of the cloud computing model, often leveraging techniques including multi-cloud, PaaS, Agile, micro-services, containers, DevOps, and CI/CD. An effective QA strategy is dependent on their core features, and testing is crucial. There are five key areas to consider here – resiliency and reliability, scalability, robust automation, dynamic environments, and security (Figure 15).

A micro-services-based architecture combined with CI/CD methods can accelerate deployments of cloud native applications. Effective testing on top of that ensures data persistency across various services,

Figure 15: Key areas to consider in testing cloud-native applications

Resiliency and Reliability

Forcing it to fail is one way to test how well it recovers. It's not only important to design for failure for also test for recovery. Testing should consider traffic routing during overloading and how does load balancing effect the performance. Validating how users are handled when some features are not available is also critical.

Scalability

Auto-Scaling is a very important feature of cloud computing services. The native apps must be aware of the scalability requirements due to dynamic load patterns and include them in the test strategy.

Robust Automation

For cloud native apps, automation is the key lever that brings the most impact in terms of speed at which software is deployed.

Testing must simultaneously test the deployments, perform a selected set of functional tasks and undertake negative task- based scenarios.

Dynamic Environments

As the workload on cloud is dynamic with fluctuating demand, the native apps are designed by spinning a new environment and defining infrastructure as code which helps to reuse the code, as it can be leveraged across the environments.

Security

A foundational piece of security applications in the cloud is application security testing. Dynamic testing techniques can be leveraged at various stages of the SDLC to identify vulnerabilities in applications.

Source: Deloitte analysis.

Figure 16: Key testing strategies for cloud migration

Migration Strategy	Definition	Testing Approach
Rehost	Moving existing applications to a solution based on Infrastructure as a Service (IAAS) without changes.	• Validation of end-to-end business workflows ensuring all **pipelines and integration** points are working as expected. • Latency and **scalability** test to diagnose issues related to system response time.
Replatform	Making few cloud optimizations to achieve a tangible benefit, without changing the core architecture of the application.	• Perform **integrated/usability** testing across multiple business processes mainly focusing on replatformed features. • Risk-based testing on targeted functionality with end-to-end automated regression suite.
Refactor	Rearchitecting and redesigning of application and infrastructure, using cloud norms like microservices and containerization.	• Expanding coverage for functional and non-functional tests. Include **disaster recovery** and **golden replication** testing in scope. • Ensure **data generation**/massaging and **penetration testing** for cloud-native.
Repurchase	Moving away from bespoke applications or customized application to COTS application. Using industry standard tools to help reduce technical debts and optimize work processes.	• Focus on key business functionalities to compare application functions to the application hosted on-premise. • Detailed end-to-end validation of Legacy or **multi-cloud** integration and data conversions.
Retire	Applications are either removed or decommissioned that are no longer needed.	• Minimal testing needed for applications to ensure usual business is continued and all processes are archived. Some integration testing can be performed to check data flow and pipelines in the E2E application.
Retain	Develop high-level CX roadmap and identify success measures to guide transition of the business from launch to steady state.	• This strategy is termed as "passive" – it also doesn't involve migrating an application to the cloud, because: – Legacy operating systems and applications are not supported by cloud environments. – The business is heavily invested in the on-premise application and may currently have active development projects.

Source: Deloitte analysis.

integration, and seamless coupling of multiple micro-services for the end customer. Container services such as Docker package everything (e.g., code, configuration, OS, test data) into containers, making deployment and testing easier.

For testing, Kubernetes clusters or on-demand clusters created on a container-capable CI worker node can be deployed. These clusters can provide the automation and observability necessary to manage applications at scale and with high velocity. The cache process approach for stateless requests should also be considered along with the disaster recovery process. Kubernetes can also help with performing rolling restarts, graceful shutdown, and releasing client threads.

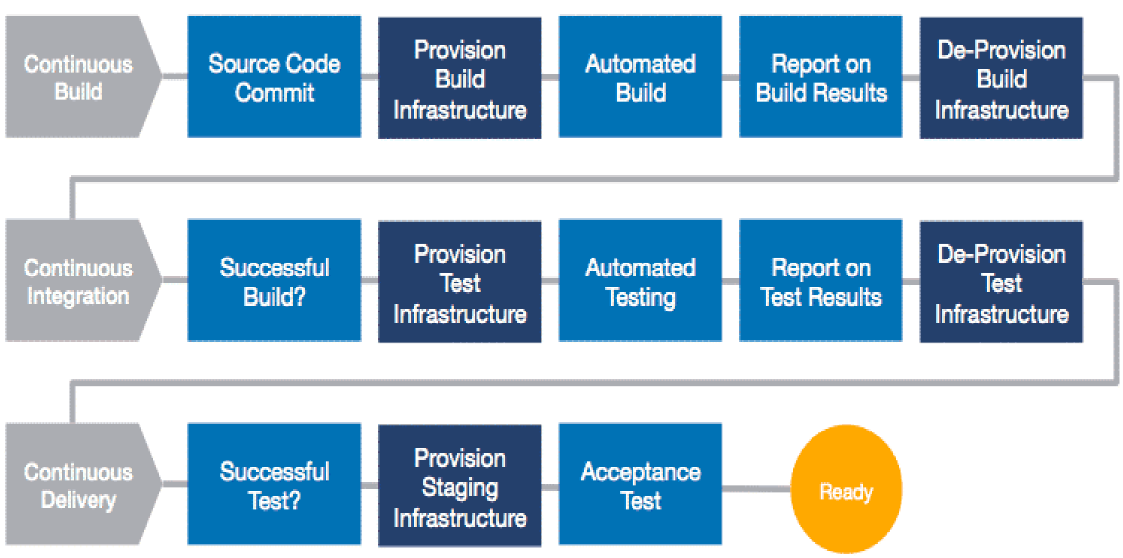
It is critical to ensure graceful system degradation to user requests on the unavailability of legacy systems. Apart from measuring relevant QA metrics, a focus on ensuring application achievability and the ability to troubleshoot during downtime is crucial.

Continuous testing in cloud

Continuous testing is a set of capabilities that drive higher software delivery and organizational performance. When implemented diligently, continuous testing enables flawless continuity in the delivery cycle. Integrating a fully automated continuous testing process into the software development lifecycle is the most effective solution for a successful CD process.

To accelerate the application lifecycle through CI/CD (Figure 17), organizations need to leverage infrastructure resources like virtual machines, storage, and networking to automate the build, integration, and testing of code under development. Companies are increasingly turning to cloud-based options, both public and private, to supply resources for CI/CD processes. They provide the necessary flexibility, performance, and availability needed to ensure quality.

Figure 17: Testing lifecycle through continuous build, continuous integration (CI), continuous delivery (CD)



Source: Deloitte analysis.

Continuous testing leveraging cloud provides benefits for DevOps teams including:

- Testing across multiple versions, devices, and carriers.
- The ability to test on new devices even on the same day they are released.
- On-demand parallel execution.
- Testing across geographical barriers.
- The power of analytics.

Built-in analytics are a significant benefit of many cloud test automation solutions. Cloud-based analytics are built to handle a high-volume of daily test data and include test reporting to give teams visibility into what went wrong. Fast feedback helps minimize potential delays and get defects fixed faster.

The takeaway

Cloud infrastructure can boost test automation coverage, scalability, and security while supporting the business roadmap and other planned activities. **Cloud-based testing**, specifically, **can accelerate the testing process and reduce overhead costs** associated with maintaining in-house infrastructure. It can **also help organizations acquire the required tools, software licenses, and infrastructure at minimal cost** while promoting maximum utilization. Organizations should **consider leveraging a cloud-native digital tester** that can test autonomously alongside humans to further increase testing accuracy and speed.

VII. Chaos is a Good Thing

VII. Chaos is a Good Thing

System downtime adversely affects brand reputation and revenue, particularly with today's customer expectations. And while it can vary by organization, the sheer cost of downtime can also be dramatic. Gartner published a widely referenced study estimating the average cost of downtime to be $5,600 per minute.[8] Yet downtime is inevitable at some point and is often directly proportional to system complexity. It doesn't help that expectations for limited downtime have increased as systems have become more complex – with organizations moving to the cloud, for instance, distributed systems are becoming the norm.

So how do you reduce failure rates and minimize the impact of such failures? Well, one method is to fail the system on purpose. Failing the system in test environments can expose weaknesses and allow for fixes to be implemented and tested before identical scenarios occur in production.

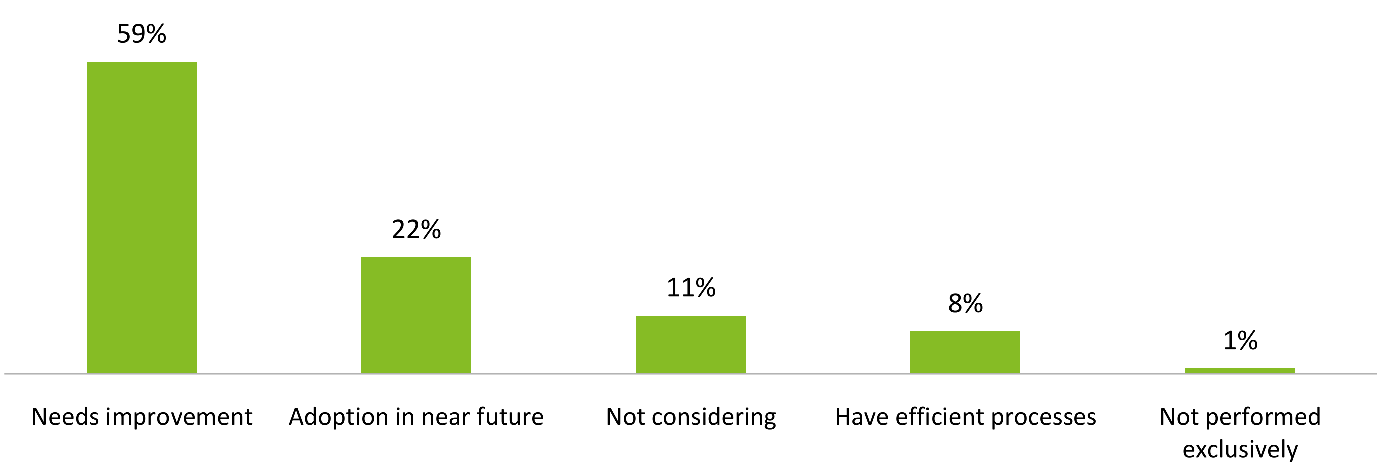
Intentional failure is the thought process behind chaos engineering. Pioneered by Netflix to ensure high systems availability in the face of extremely impatient video-streamers, the concept has become widely adopted across industries including financial services, communications, retail, and hospitality.

Why chaos engineering

Chaos engineering helps us understand what happens to the system as a whole if any of its moving parts are impacted adversely – either through complete unavailability or partial unavailability returning errors and exceptions. Chaos engineering aims to bring an organization's worst nightmares to life in order to avoid them, helping expose opportunities to improve overall system design to withstand the impact of failures. It is not a panacea, however, for all externalities beyond the reach of mimicry. Some of these external failures may include power outages and cloud provider downtime.

With such high upside to the practice, we turned to our survey to understand how organizations were adopting chaos engineering (Figure 18). Over half of respondents said they have a chaos engineering

Figure 18: Extent to which organizations have a testing strategy for resiliency and chaos engineering



strategy in place but are looking to improve it. So, how can you improve or implement a chaos engineering strategy?

Chaos engineering implementation

Consider you have an application hosted on the cloud that serves your customers vital account information and allows them to make bill payments. Your application is implemented using microservices and is talking to a NoSQL DB as well as a Redis cache (to improve performance). Your application is hosted in multiple regions to ensure high availability. On paper everything looks good until you are actually hit with multiple production incidents that bring your system down for long periods of time.

To get ahead of these issues, you could implement chaos engineering through a few key steps:

1. **Understand your application to determine its key points of failure**. You can do this in two key ways:
    - Look at your application implementation in production to identify current availability metrics and understand the biggest drivers of system downtime.
    - Look at the implementation architecture to determine single points of failure.
2. **Build your chaos framework**. This is where the magic happens. It involves mimicking different types of failures across different components. For instance, mimicking availability failures on your dependent backend services and components – Couchbase or Redis from our example above. This might include:
    - Induced latency across your different application components.
    - Induced software failures – returning exception responses or exception status codes and seeing how your system handles erratic backend behavior.
3. **Run your scenarios**. Start with the scenarios which have the smallest impact and then move up the ladder. This way you can start small – it's easier to control and curb the impact of less complex scenarios.
4. **Analyze your results**. A pat on your back for successfully completing your chaos experiments. Results, good or bad, should be celebrated since they put you one step closer to building more resilient systems.
5. **Back to the drawing board**. Based on your observations, look at what can be changed and how your system can be made more resilient. This may involve steps ranging from a simple code change to implementing resilient design patterns to a more complex application architecture redesign.

Advantages and measuring success

How does chaos engineering differ from other types of testing such as failover or failback? Other types of testing only test for certain pre-determined configurations or parameters; they are predictive and less comprehensive. Chaos engineering delves deeper into your system and provides confidence not only to your development team, but also your production support team. The randomness in failure will surface vulnerabilities in monitoring and alerting mechanisms while helping the operations team prepare to quickly respond to unanticipated failures and resolve issues.

Chaos engineering provides confidence not only to your development team, but also your production support team.

How do you know if your chaos experiments have been successful? The answer lies in your application metrics. For one, you should see an uptick in your application availability. Additionally, metrics such as

MTTR (mean time to recover) and MTTD (mean time to detect) should reduce while MTBF (mean time between failure) increases.

Future of chaos engineering

As chaos testing becomes more mainstream, it will become increasingly commonplace for organizations to embrace it. Expect AI, ML, and automation to play a greater role, in two key ways:

- **"Smarter" chaos engineering**: AI and ML models will be used to predict the point of "chaos injection" for a given application and automated chaos frameworks will work their magic to induce scenarios.
- **Automated chaos frameworks**: In addition to commercial frameworks, open-source frameworks will emerge that enable automated chaos testing and better integration into a DevOps pipeline. These frameworks will not only execute tests, but also help you define parameters to be monitored and documented for further review. Integration with AI and ML will also enable the delivery of better remediation measures.

The takeaway

Stability philosophies have evolved from never-fail to fail-fast and recover-fast, defining a new resiliency engineering school of thought. **Chaos engineering** is becoming imperative for testing and enabling resilient, fault tolerant applications and systems. Effective implementation requires five steps of **identifying critical points of failure**, **building a chaos framework**, **running scenarios**, **analyzing the results**, then **adjusting based on those results**. While still nascent, **new tools and frameworks are emerging – including opensource options** that are proving effective. We can **expect an evolution toward "smarter" chaos engineering with predictive AI and ML as well as automated frameworks** that help define parameters and execute tests.

VIII. Strategic Approaches to TDM and TEM

VIII. Strategic Approaches to TDM and TEM

Test data management (TDM)

Test data management (TDM) is a functional discipline that helps to manage proper identification, creation, verification, masking (where needed) and management of test data across all test phases – in alignment with testing requirements and information management policies and guidelines.

TDM is becoming a crucial function for the many organizations looking to reduce time spent on identifying and creating test data. The TDM market is expected to grow to $1.6B by 2027, up from $721M in 2020 (11.9% CAGR).[9]

Benefits of improved TDM

Efficient TDM helps maximize testing ROI and complements testing efforts for maximum performance and coverage levels. It can help significantly reduce delays in the testing and development process.

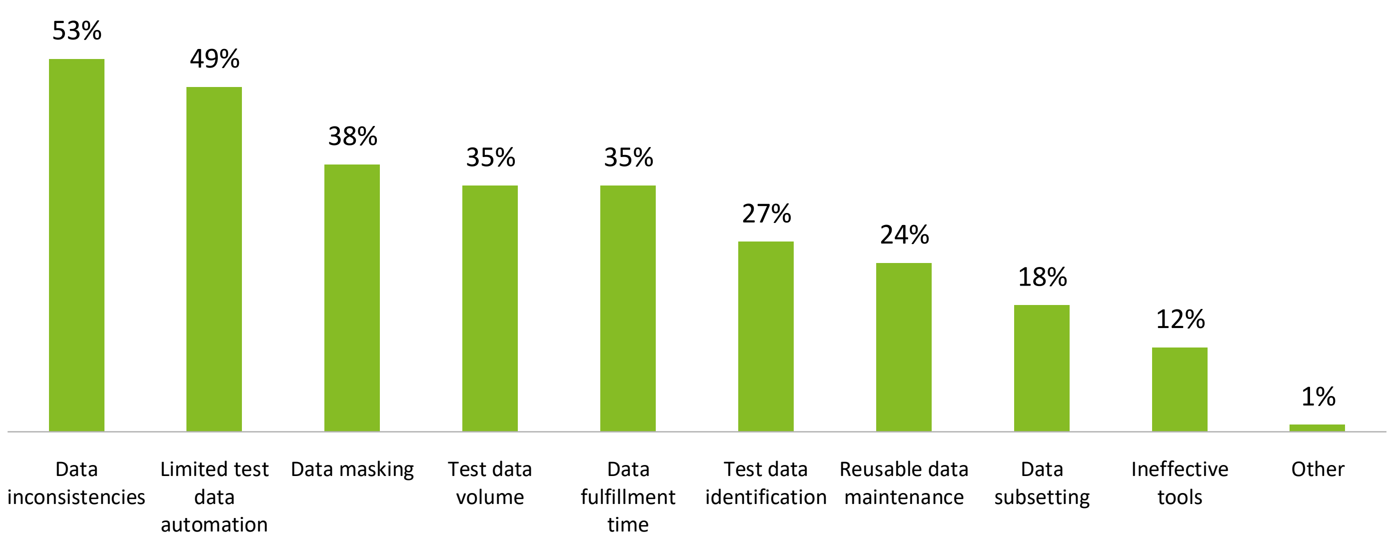
TDM is gaining popularity due to its use of a structured engineering methodology to evaluate data specifications for all potential business scenarios and the extent to which it can help make structured and well-segmented data available.

Primary TDM operations include data masking, data refresh, sub-setting, extract/transform/load (ETL), and synthetic development. TDM helps ensure test data-related issues and vulnerabilities are detected and addressed before production, enabling timely implementation of quality applications.

Challenges of TDM

We turned to our survey to look at how organizations today view challenges associated with TDM (Figure 19). Data inconsistencies (53%) was the top challenge cited, with lack of test data automation (49%) a close second. Data masking (38%), test data volume (35%), and data fulfillment time (35%) were other top friction points. Deploying cutting edge TDM processes can help reduce the effects of these challenges, increase test coverage, and decrease time-to-market. An effective TDM strategy is essential.

Figure 19: Challenges organizations are seeing in test data management (TDM)



Note: Respondents could select multiple answers.

Future of TDM

The TDM function is moving toward a self-service era. In this era, the end user of the test data will request specifics through a portal, filling in details of testing needs while data is created in the background through automation scripts. Automation scripts will be configured in a way that requires minimal human intervention to procure the necessary test data. In the event any of the automation scripts fail, the test data team will be notified to make necessary corrections. This self-service methodology minimizes the need for a dedicated TDM team through the year. Instead, once the self-service portal is set up, only a minimal support team will be required for maintenance or new test data requirements.

AI and ML are becoming an integral part of the TDM function and will greatly enhance how we test in the future. Self-healing scripts for test data creation will greatly reduce script maintenance time by automatically identifying and correcting scripts without human intervention. The next major leap for AI and ML in TDM is to automatically identify test data requirements based on written test cases. Based on a test case, the algorithm will identify test data needs, check whether the data is already available, and – if necessary – create or procure the data from production after all necessary de-identification procedures. This will make test data development as touchless as possible, supporting a multitude of industry-specific test data requirements.

Test environment management (TEM)

Test environment management (TEM) is a critical strategic function in the software delivery process. It helps manage, control, and build non-production environments with appropriate governance and services during the testing phase to ensure that environments are available for IT programs in the right size. Proper management of test environments enables project teams to effectively validate systems and applications before deploying to production.

Why is TEM important?

Test environments are a combination of the hardware, software, data, and configurations required to execute planned test scenarios. They mimic production environments for various testing needs. With the advent of Agile, every software release has multiple nuclear teams requiring a higher number of test environments for varying testing types and phases such as integration, regression, UAT, and performance. This requires effectively forecasting the need, planning the need, designing a process to raise the request for it, and collaborating with vendors (e.g., cloud and/or on-premise) to provision the request ahead of time.

Organizations need to be able to provision the right environment, control changes, and provision the required infrastructure support.

Successfully validating business applications often requires navigating the complexity of multiple user channels and distributed systems. Organizations need to be able to provision the right environment, control changes, and provision the required infrastructure support.

What TEM challenges do organizations face?

Our survey surfaces how organizations today classify the challenges faced in their test environments. We pointed out in the Executive Summary that 58% of respondents highlighted increased test environment challenges as a driver of their overall testing spend (Figure 2). As a close second to new age technologies (60%), it is a glaring driver of overall costs.

We also asked about the specific challenges organizations see in their test environments. The results tell a similar story – 53% of respondents highlighted cost of maintaining test environments, making it the top challenge overall (Figure 20). Lack of on-demand provisioning (51%) was a close second. Improvements in both areas go hand-in-hand.

Sentiment from our survey respondents aligns to more specific complexities we've seen organizations face in the market. Siloed visibility, lack of information on project metrics, no single view of the portfolio, and timing impediments from project and test environments are key points of friction in project delivery efficiency.

So how can some of these challenges be addressed?

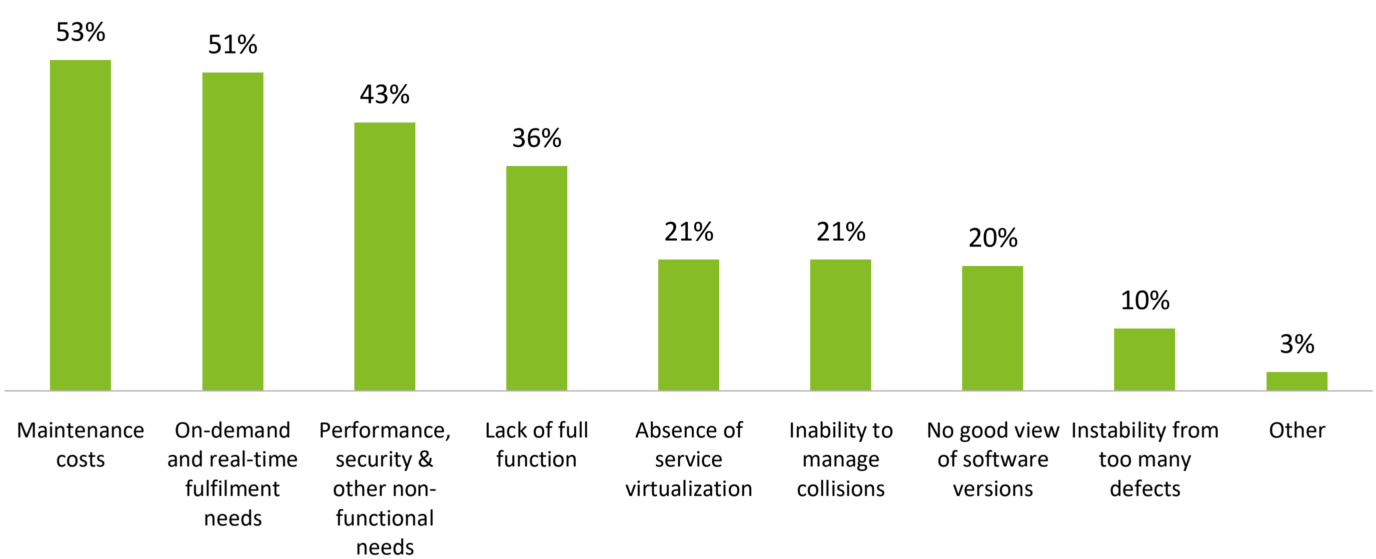
Effective TEM strategies

A robust TEM strategy contains several key features. Paired with a workflow-based solution for pre-production environments, it can help in planned provisioning, ensuring availability of production-like test environments, and giving internal and external teams one place to collaborate on and view environment availability, configurations, and conflicts. Key features include:

- **Automated or on-demand builds**: Triggering builds on-demand to expedite the hand-off of new code from dev to test.
- **Environments management**: Scheduling environments as reserve requests based on availability and configuration to match release plans.
- **Change management**: Scheduling environments as change requests for code and configuration updates to match release plans.
- **CI/CD pipeline overview**: Maintaining visibility of fast-moving CD pipelines – viewing where and when new code has been deployed and what tests have been scheduled and run.
- **Tools integration**: Integrating with operational toolchains already in use such as ServiceOne and Remedy for change and incident management. Building automation tools such as Jenkins to integrate with the test environment build process.
- **Lean environment inventory**: Re-provisioning environments that are not currently in use by teams in order to reduce infrastructure costs.

TEM maximizes organizational efficiency and quality and supports shift-left approaches. The TEM capability process enables project teams to simplify complexities

Figure 20: Challenges that organizations are seeing in test environments



Note: Respondents could select multiple answers.

with industry-leading tools and an integrated approach (Figure 21).

Benefits of TEM adoption

An effective TEM strategy can help improve software release speed and reduce dev/test cycle time by providing a single source for tracking environments, related configurations, and availability schedules to make environment bookings for both internal and externals teams. It can drive increased usage of environments due to on-time provisioning of the environments that have code as well as effective infrastructure and configuration changes.

TEM can also help eliminate environment contention. Collisions can be prevented by providing clear visibility of available environments for functional and non-functional testing teams.

Overall, advanced TEM capabilities lower environment set-up and maintenance costs by enabling the provisioning of environments on demand, the reuse of environments, and the efficient use of infrastructure.

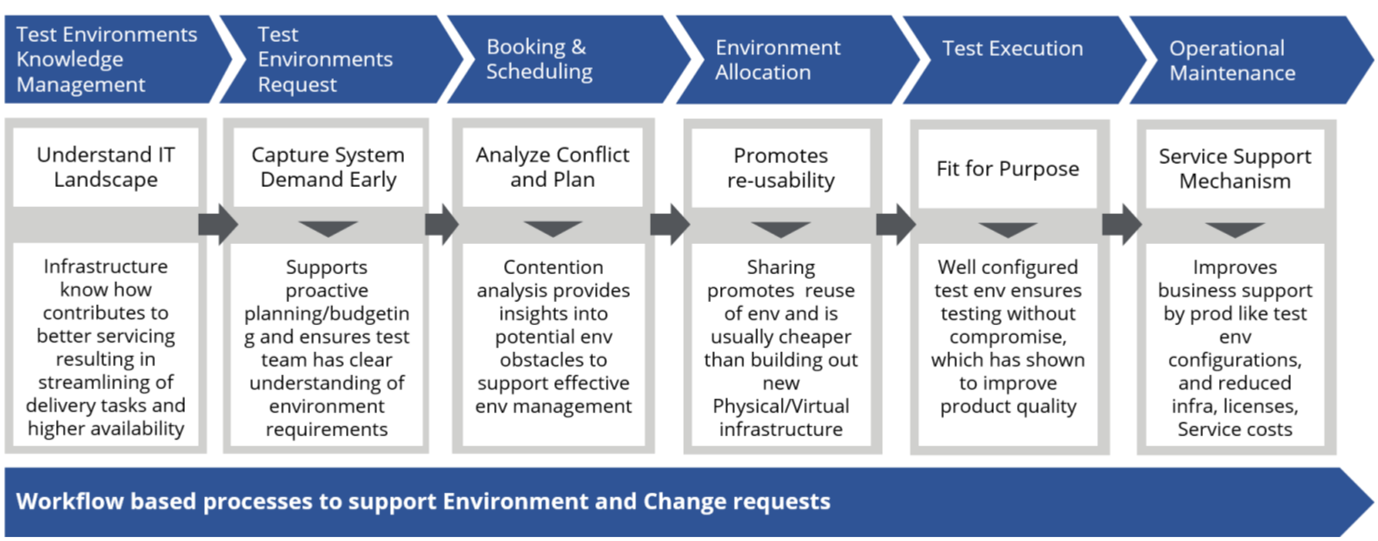
What does the future look like?

Agile and DevOps are bringing significant changes in toolsets used by operations, development, and test teams. Application delivery and deployment are no longer limited by silos. IT and infrastructure leaders should reexamine their strategies to address evolving digital needs and improve the speed and quality of application delivery.

A combination of the latest toolchains for Agile and DevOps platforms will completely transform the end-to-end testing lifecycle. An RPA solution will also serve as an orchestration engine to automate repeatable testing processes, including:

- Provisioning of test environments based on project needs.
- Updating a test environment's patch levels to be consistent with the production instance.
- On-demand spinning up and spinning off of test environments for each release/project.
- Managing test environment conflicts.

Figure 21: An effective TEM capability process



Source: Deloitte analysis.

- Data provisioning through TDM tools for data masking and replication before execution.
- Generation of insights and dashboards (e.g., using chatbot engines) for test results.

Organizations today demonstrate growing interest in tools that encompass both development and operations capabilities. Specialized teams have traditionally met the needs of each individual phase of the application delivery value stream through specialized tools. But the lines will continue to blur as these delivery platforms provide a more unified application delivery pipeline.

The takeaway

TDM is a crucial function for organizations to reduce time spent on identifying and creating test data by managing proper identification, creation, verification, masking (where needed) and management of test data across all test phases. It can help organizations **drastically accelerate time to market**, **reduce infrastructure costs**, and **reduce defect rates**.

Effective **TEM helps organizations forecast and plan for environment needs, design processes to raise requests for environments, and collaborate with vendors to provision the requests on time**. It can help organizations drastically reduce test cycle time, eliminate environment contention, and optimize infrastructure spend.

IX. Know Your AI Testing Needs

IX. Know Your AI Testing Needs

AI is one of the fastest-growing and pervasive data-driven technologies. From bringing in new customers to achieving operational synergies, organizations are applying AI models to a variety of use cases. Evaluating testing needs for AI systems requires an understanding of the algorithms, training data, and statistics applied to create them.

Effectively implementing AI systems requires teams with the right combination of industry, automation, and AI technology expertise to bring a hybrid solution tailored to the context of an individual business. Hybrid solutions help achieve better model accuracy, less adjudications, high true positives, and low false negatives.

When we asked organizations about the biggest challenges they face in implementing AI, responses were wide-ranging (Figure 22). The most common was a challenge with resource skills (55%), underscoring the difficulties in upskilling fast enough. A lack of access to tools (50%) and challenges in accessing data (39%) were next. To address and overcome some of these challenges, it's best to start by evaluating needs.

Evaluating AI model testing needs

AI systems are non-deterministic; therefore, the predictability of outcomes depends on data variations and the specific model development approach. A Forrester survey found that over 30% of respondents see testing their AI systems as either "Challenging" or "Very Challenging," while 32% think they are "Improving."[10] Clearly, there is work to be done to improve AI systems testing.

Before implementing an AI system, establish some baseline answers to the following questions:

- What qualifies as a bug in the case of AI and how can it be fixed?
- How can the right level of confidence be achieved?
- Are the results fair to all concerned?
- Are current implementations, requirements, opinions, and values reflected in the algorithm?
- What are the biggest risks with AI and how can they be dealt with from a testing point of view?

Figure 22: Challenges organizations see in implementing AI and ML

Note: Respondents could select multiple answers.

- How can certainty that the system behaves correctly with a wide variety of input and users be maximized?

In order to answer these questions and ensure that proper testing is done, it is necessary to evaluate the testing need of the AI model. An AI system validation approach is equally as important.

Key factors in determining an AI system validation approach

AI systems testing is often an unexplored territory and the testing approaches, skills required, and level of complexity will vary according to the technology being tested. For instance, testing vision, voice, or language understanding in a conversational system is going to be different from testing a decisioning or recommendations engine. There are several key factors to keep in mind when determining an approach:

- Requirements and expectations from the AI system.
- Expected model accuracy.
- Data validation.
- System architecture and the model's integration with upstream/downstream applications.
- Performance, security, and compliance expectations.

Let's look at each of these in a little more detail.

Understanding requirements

Teams should also have a deep understanding of the model behavior in order to understand the AI system requirements that need to be validated. Let's look at an example – a chatbot application. The different elements of model behavior for a chatbot to consider would include:

- How the model will analyze and respond to a conversation (e.g., suggesting what has happened historically, a likely outcome based on inputs, or an action that needs to take place).
- Methods of data extraction from the conversation.
- Expected responses and exceptions within a conversation.
- Accuracy levels of responses provided.
- Downstream and upstream system integration and UI requirements.
- Various user roles and personas.
- Model monitoring and retraining approaches.

Achieving expected model accuracy

Model accuracy plays a key role in an AI system to reduce data noise, increase recommendation accuracy, and minimize user adjudication. There are a few key ways to help determine model accuracy:

- Increase or decrease thresholds for each data element and intake type based upon performance reports.
- Adjust sensitivity and analyze ROC curve to validate efficacy of the model.
- Model F1 score to determine model efficacy.
- Use a confusion matrix to validate the performance of machine learning (true-positive, true-negative, false-positive, false-negative).

Selecting the right data

Model training data are the nerve of an AI system. It is important to identify a test data set appropriate for the volume of training data being used. Testing will be based upon input and the quality of data and this data should be isolated from the training data set. Testing datasets should consist of data that validates:

- Model boundary conditions and variations.

- Data engineering logic.
- Transformation of model prediction to UI and downstream applications.
- Overall data set submissions such as analysis reports, risk assessments and predictions, and regulatory submissions.

Integrating the AI model with other apps

The aim of integration is to ensure that the AI system can seamlessly provide results without any information loss. It is important for AI systems implementation to identify and validate the following:

- Impacted applications and business processes aligned to the end-to-end value chain.
- Interaction between different models, APIs, data integration layers, UI, and other elements.

Safeguarding non-functional requirements

AI systems must often predict results based on massive datasets and a heavy inflow of data. Performance assessments are critical here. Many predictions are made using data that includes PHI/PII data, which brings high risk and significant security compliance requirements. Security, vigilance, threat assessment, and risk mitigation are equally critical.

Vision on validating AI systems

Validating AI models and making critical decisions do not need to occur in a linear fashion. Organizations should leverage domain understanding, experience with different AI technology stacks, and quality engineering best practices to come up with tailored AI hybrid solutions. This may involve focusing on automation testing through A/B testing, large data corpus testing, API testing, or integration data engineering testing.

Clearly, as our survey captured (Figure 22), a diverse set of challenges in implementing AI persist. To address tooling and data access challenges, specifically, organizations should invest in repeatable assets and accelerators that can generate large volumes of data comparable to model training data sets. They should also focus on model accuracy as a key performance indicator, which can help reduce data noise.

The takeaway

AI systems have become pervasive and are being integrated with end-consumer applications everywhere. Because AI systems are programmed to learn and can behave differently based on what they learn, it's **crucial they function properly from the beginning to avoid hurting the user experience and violating user trust**. Organizations should evaluate AI model testing needs upfront and define a strategy that focuses on **understanding requirements**, **maximizing expected model accuracy**, **selecting the right data**, **integrating the model with other apps**, and **safeguarding non-functional requirements**.

X. IoT Starts with Basics

X. IoT Starts with Basics

IoT has an accelerating growth trajectory due to the advancement of a number of complementary technologies. 5G's arrival is driving a more seamless end-user experience for a greater number of connected devices. Blockchain is making it easier to track IoT devices. AI and ML are more accessible and deployable, unlocking a greater range of device functionality throughout the connected ecosystem.

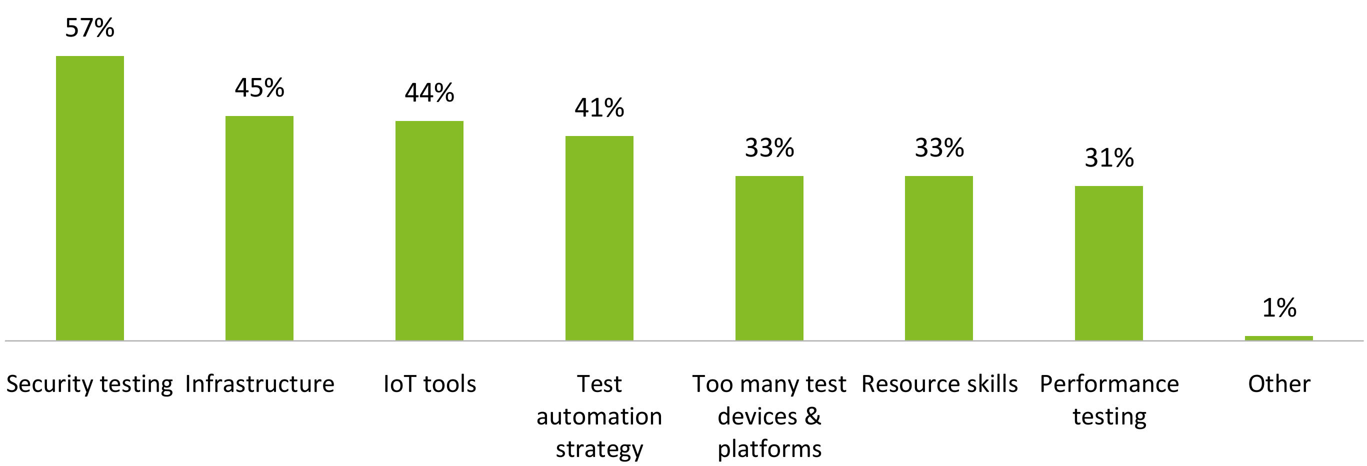
There are a wide variety of IoT use cases across industries from financial services to healthcare to retail. In healthcare, for instance, wearable devices can help continuously monitor patients without needing to keep them in a hospital. In financial services, users can complete a multitude of tasks from their smartwatches including payments, checking credit history, and making balance transfers. Organizations are realizing that IoT is foundational to creating a well-connected digital business. IDC predicts global spending on IoT will hit $1.1 trillion by 2023, a 52% increase from 2019 spend.[11]

With the sheer volume of connected devices, networks are vulnerable to hackers gaining access through one link in the chain. Data theft is a risk without effective countermeasures. Some sectors are at higher risk than others – for instance, PII and PHI within the financial services and healthcare sectors is often sought after for sale on the dark web. For organizations who want to be leaders in adoption of disruptive technologies, quality is a critical component of ensuring data security.

Testing IoT applications: key challenges and how to address them

We found in our survey that security testing (57%) was the most challenging aspect of testing IoT applications cited by respondents (Figure 23). Indeed, validating data security is critical before go-live, and can be highly complex. A balanced spread of challenges seemed to affect organizations behind that, with a need for the infrastructure (45%), required IoT tools (44%), and a test automation strategy (41%) closely following. The need for the right resource skills (33%) to understand interconnected systems and perform validations is also a noteworthy challenge.

Figure 23: Key challenges organizations are seeing in testing IoT applications



Note: Respondents could select multiple answers.

Let's look at ways to address a few of the top challenges in more detail.

Testing security: Organizations should divert more time and resources to testing the security of IoT applications. The challenge of testing security can be addressed through a focused approach that covers five key areas – authentication/authorization protocols, cloud interfaces, encrypted & decrypted data, physical and infrastructure security, and hardware and software quality (Figure 24).

Infrastructure: Organizations typically struggle to establish the right infrastructure to build and test connected systems. Challenges here include increased mobility needs of users, network availability and bandwidth, and data center capacities needed to cope with vast amounts of data.

To address these issues, organizations should establish the right network selection and develop flexible and modular solutions that address different levels of hardware and software integration such as physical devices and controllers, connectivity, data aggregation, and collaboration.

Test automation strategy: Testing challenges are magnified in IoT when using devices comprised of different software and hardware and varied communication protocols. This yields several combinations that need to be validated, often through a combination of automation and manual-driven test execution. When prioritizing a testing strategy, IoT's biggest dependencies – minimal latency and fast movement of data – should be top of mind. Variables including device capability, Wi-Fi setup, and network infrastructure should be thoroughly validated upfront.

How to approach IoT testing

Thoroughly defining a broader testing strategy is critical for successfully navigating all of the challenges surfaced in our survey. While the task may seem large and complex, sometimes going back to the basics can yield the greatest benefits.

- **Follow the trail**: Adopt a user-driven approach to "see through the noise" and identify the combination of devices facing either sustained

Figure 24: Five key focus areas for testing security

Authentication/ Authorization Protocols	Cloud Interfaces	Encrypt & Decrypt Data
Validation of authentication/ authorization protocols such as MFA (Multi-Factor Authentication)	Validate the cloud: on-prem connectivity for potential security gaps/pitfalls	Adopt industry proven encryption and decryption standards as data moves between different applications within the enterprise

Physical & Infrastructure Security	Hardware and Software Quality
Constant review and upgrade of physical and infrastructure security to eliminate both external and at times, internal threats	Periodic review of hardware and software quality standards to identify and resolve potential bottlenecks

Source: Deloitte analysis.

user traffic over a defined interval or are subject to ebbs and flows.

- **Sampling for success**: To wade through the complexity, employ a sampling technique to determine those applications most likely to be vulnerable or that need to be tested for potential reliability, scalability, and availability issues. This approach supports continuous testing and validation of applications most vulnerable to attacks.
- **IoT labs to emulate devices**: Set up in-house diagnostic solutions and IoT labs to perform validation of the connected ecosystem. Deploy emulators that mimic various devices and software applications to evaluate potential failures from issues such as software upgrades, hardware upgrades, fault lines, and network issues.
- **Scalability & availability**: Evaluate the scalability of devices and software applications to maintain data flow without significant latency. Conduct detailed validations to ensure availability of the applications in the middle of upgrades in order to ensure the continuous flow of data at all times.

The takeaway

5G has driven a rapid expansion in IoT adoption, unlocking myriad use cases across industries that create more immersive experiences for end users. But **vulnerabilities and security challenges are multiplied within such a vast and connected ecosystem**. To address complexity challenges, organizations should define an overarching IoT testing strategy upfront that goes back to the basics – **user-centric approaches to prioritize devices**, **widespread sampling**, **in-house labs and emulators**, and **thorough scalability evaluations**.

XI. Wisdom of Crowd Testing

XI. Wisdom of Crowd Testing

A high level of intimacy with user needs and user-centric design have become pre-requisites for building customer loyalty. The need to keep up with rapidly evolving preferences creates strain on quality engineering teams, and conventional windows for testing are getting shorter. Crowd testing supports the testing function by enlisting real users to test software under real-world conditions. By using crowd testing, teams can easily scale up their testing capacity before big releases and other significant events. This allows teams to maintain a consistent core of in-house QA resources full-time, while having the ability to flex up when the need arises.

Crowd testing market

When it comes to augmenting a quality engineering team, companies may add in-house resources, run an outsourced test, or run a crowd test.

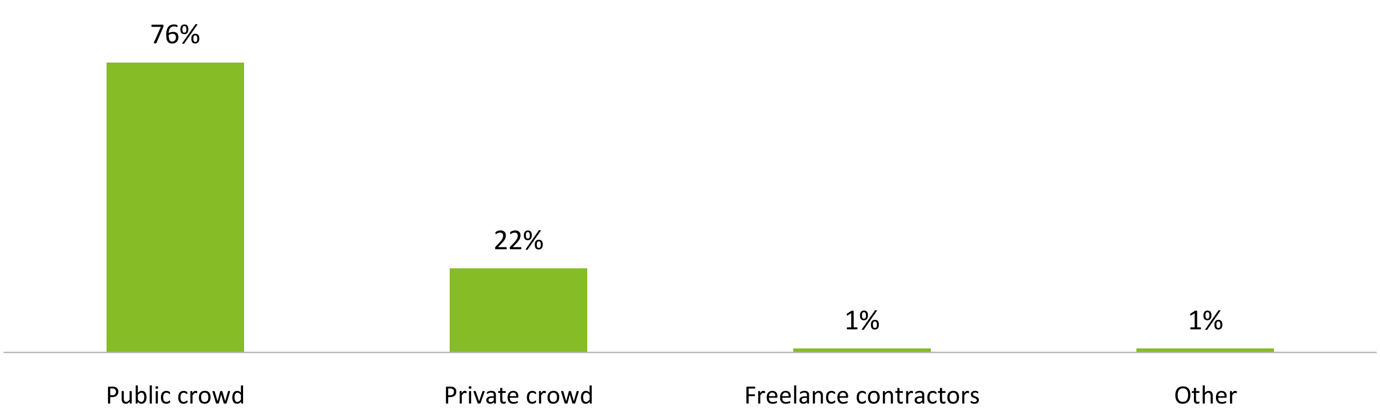
Crowd testing is usually used on an as-needed basis and used as a complement to an internal team. Crowd testers can be deployed however they best complement the core team, from broad exploratory testing to high specific verification testing. Crowd testers often provide application assessments and list defects found while executing assigned tests.

There are a variety of compensation models for crowd testers. One common one is micro-tasking. In the micro-tasking model, testers are paid according to the type of defect and its market price. Some organizations pay according to the business impact and severity of the defect.

Per-performance is another compensation model. Here, organizations pay based on the overall quality of the test. This method is common across a wide variety testing types including bug testing, mobile testing, and usability testing. Cybersecurity testing is one of the most sought-after use cases for crowd testing and testers here are considered near the high-end of the compensation spectrum.

While some organizations may set up their own crowds, crowd testing is often set up through third-party crowd testing platforms that aggregate supply and make it easier to commission projects. Our survey respondents indicated a strong preference for tapping the external crowd for testing (76%) when it came to models most suitable to their organizations (Figure 25). To drive efficiencies in this market, we're starting

Figure 25: What crowdsourcing model organizations see as best suited to them



to see more formal partnerships emerge between crowdsourcing platforms, corporations, and consulting service providers.

Crowd testing process

Most third-party crowd testing platforms provide a platform and project management framework, often providing a qualified project manager to oversee the testing process. Clients specify the required tests, tester skill sets, and types of devices to test on.

Before a project starts, testers are provided detailed test plans, sample scenarios, tools, scripts, and instructions. During execution, the testers document their observations and are rated based on the volume and quality of their reports, which drive their compensation.

Testing communities can combine aspects of competition and collaboration, as testers often work in tandem to find solutions to the problem. Furthermore, forums discussing bugs and relevant issues facilitate knowledge management and networking. Rating systems also help crowd participants gain credibility and advance their careers.

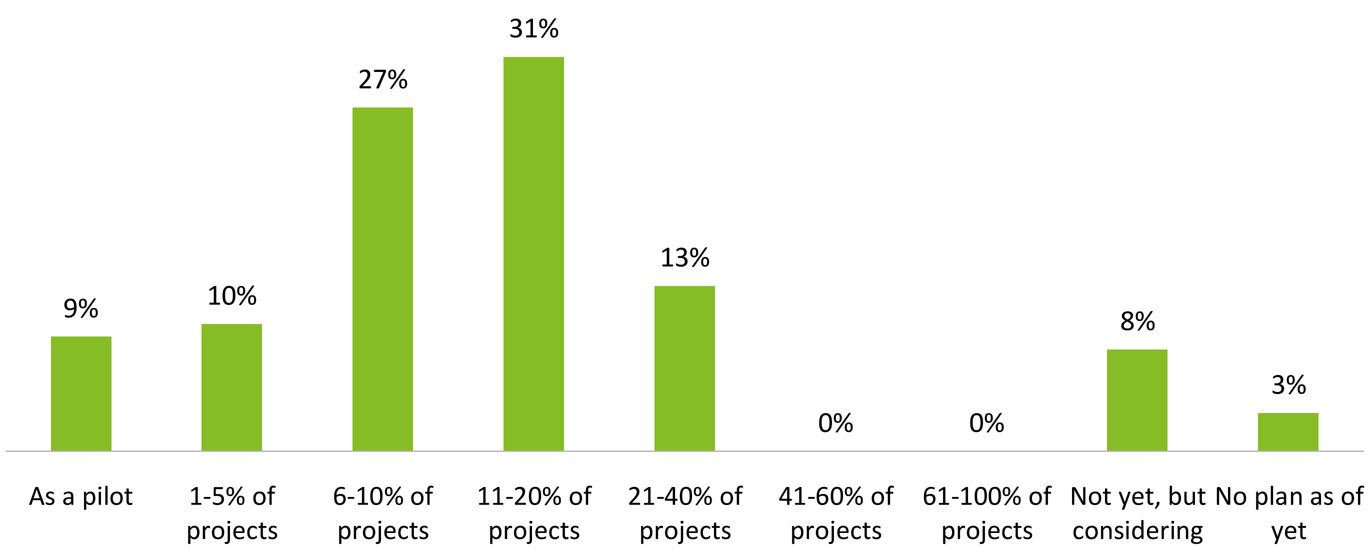
It's telling that 90% of our survey respondents have at least piloted crowd testing (Figure 26). But 48% estimate that less than 10% of their projects are crowd tested. With the variety of platforms and approaches out there, organizations looking to scale their use need to find what works best for them.

Management challenges of internal crowd testing

Tapping internal resources to form an internal crowd is another approach. With regard to internal crowd testing, we see common challenges. They often include:

- Insufficient IT competencies for internal staff forming the crowd (i.e., not everyone at an organization is trained in IT).
- Increased job complexity and amplified stress for internal resources asked to switch between regular work and crowd testing during the day.
- Needs for developing a new motivational and compensational structure.

Figure 26: Extent to which crowd testing has been used by the organization



- Hesitance by supervisors to let their employees participate in the internal crowd because it pulls time away from core work.

Our survey respondents (Figure 27) also perceive limitations including application and data security (59%), coordination (56%), and needs for control, reporting, and governance (41%).

Comparing internal and external crowd testing

Crowdsourcing is an excellent implementation method for validating features at later stages of development. Teams practicing CI/CD have also found that using crowd testing earlier in a development workflow can be helpful in finding bugs and issues before they reach production. So, which is better – internal or external crowdsourcing? Well, it depends.

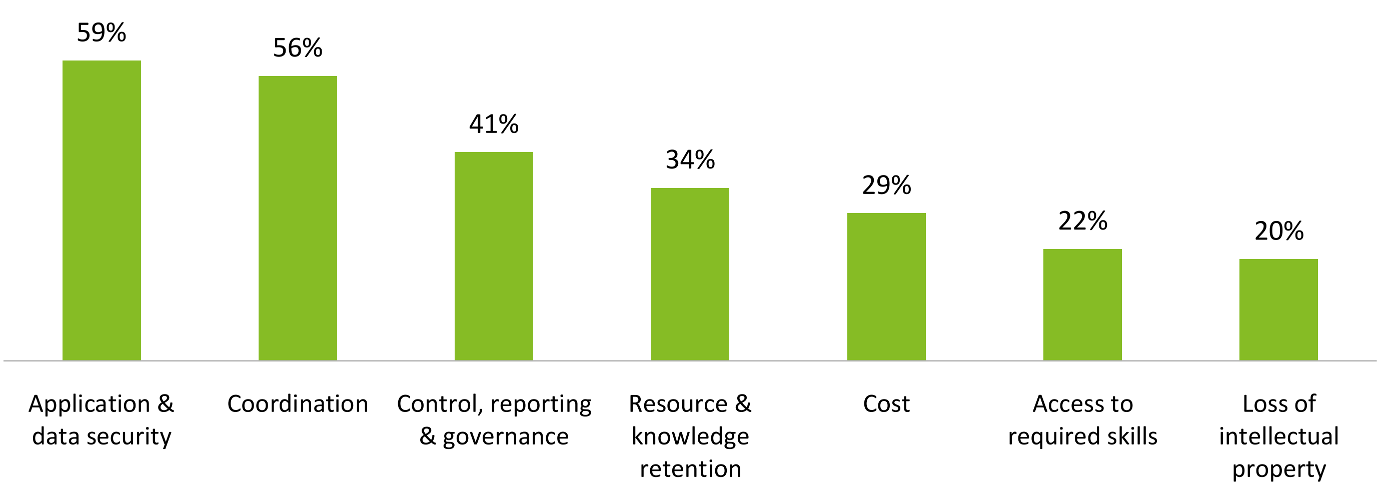
Use of internal crowds can be a more optimal route when data confidentiality, leveraging internal business knowledge, and maximizing utilization of internal IT resources are at a premium. Internal crowds can help prevent sensitive customer data, for instance, from being disclosed publicly. In-house expertise can also be better tapped in some cases to improve software quality.

External crowds help in situations where there is insufficient quality of software testers or for more complex or novel projects. Tapping the external crowd opens organizations up to a global pool of resources with an endless variety of skills. A much larger range of hardware and software releases are possible.

Use of internal crowds can be a more optimal route when data confidentiality, leveraging internal business knowledge, and maximizing utilization of internal IT resources are at a premium.

Data privacy concerns and the sometimes-unknown reliability of testers are potential drawbacks. Lack of internal knowledge can also hamper alignment between external testers and the organization. To ensure confidentiality and maximize testing quality, look for crowd testing solutions that provide rigorous talent screening and privacy assurances like built-in non-disclosure agreements.

Figure 27: Limitations that organizations see in crowdsourcing



Note: Respondents could select multiple answers.

Comparing crowd testing and outsourced testing

Having looked at the differences between tapping the internal crowd vs. the external crowd, it's helpful to understand the tradeoffs between leveraging external crowd sourcing against traditional outsourcing. Figure 28 lays out the key differences.

Figure 28: Comparison of crowdsourcing vs. outsourcing in testing

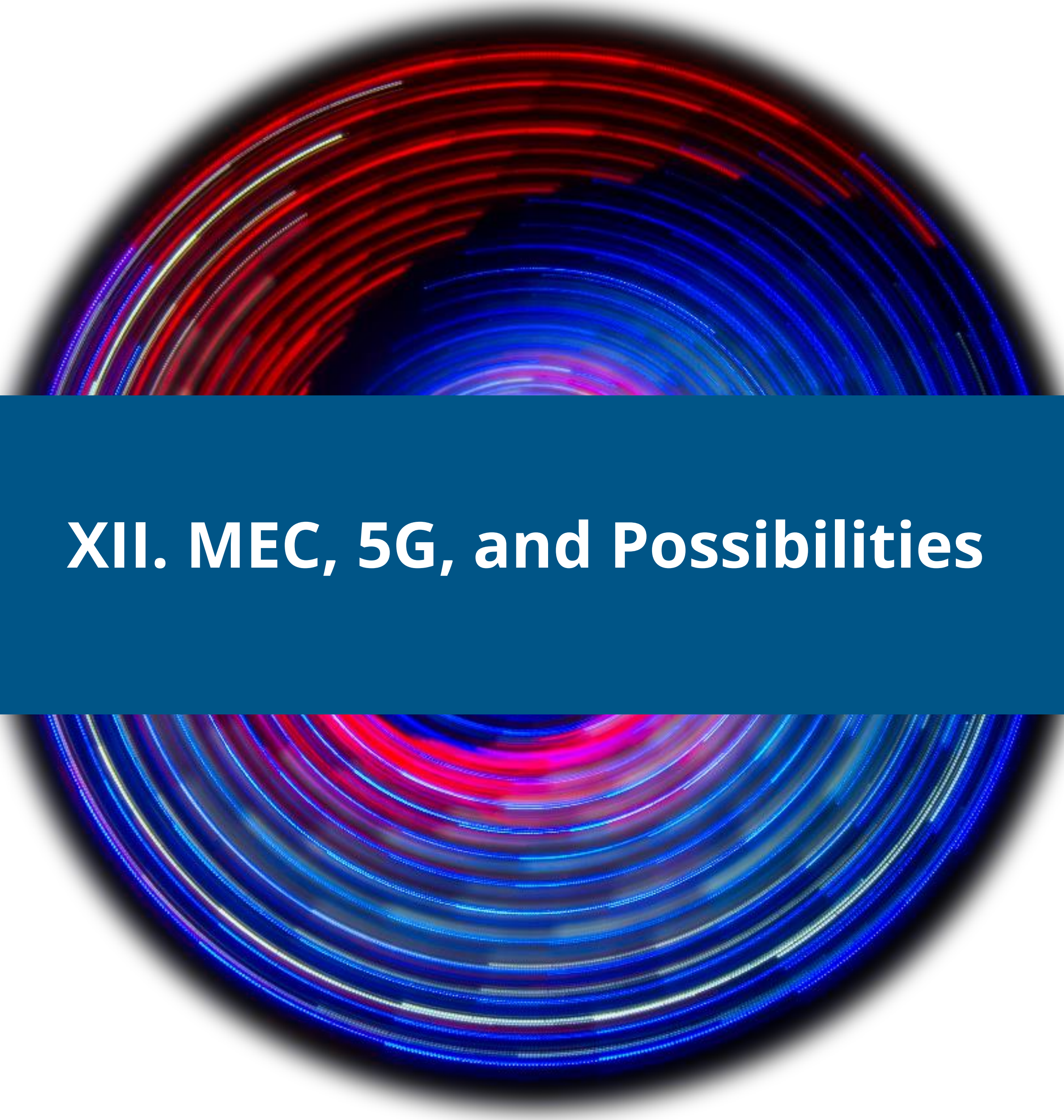
Crowdsourced testing	Outsourced testing
Compensation is often based on finding defects and is cost effective.	Compensation is often by the hour which tends to be higher cost (usually less than in-house testing teams).
Communication relies on understanding defects which can make it challenging to talk to testers.	Representatives are there to help make communication easier.
Testing environment depends entirely on individual testers, where capabilities differ from person to person.	Testing platforms, management, environment, and tools are wholly owned by one company which can be an efficiency advantage.
Limited concept of team, time zone, or deadlines as defects are reported very quickly.	Planned budget with one team, time zone, and specific deadlines.
Unknown group of test resources with work quality based on reported defects.	Dedicated team with a group mindset.
Data confidentiality can be at risk in cases of leakage or limited confidentiality assurances.	Application data is kept confidential.
Focuses on quantity more than quality and it is the organization's responsibility to determine which defects recorded are real and worth repair.	Focuses on identifying all valid defects within a specified timeframe and budget.
Quality of a defect depends on the experience of the testers, which can often be mixed.	Uses a fixed number of testers with skills in specified types of functional and non-functional software tests.

Source: Deloitte analysis, Software Testing Material[12].

The takeaway

Many software development organizations are ramping up their use of **crowd testing**. It **provides a major advantage in that resources can be scaled up and down as needed,** while also serving as a reliable way to gain real-time feedback and find bugs. Not every project is suitable for crowd testing. Organizations should **consider their internal capabilities, complexity and novelty of the project, and privacy needs** when deciding on the use of crowdsourcing, outsourcing, or internal teams.

XII. MEC, 5G, and Possibilities



XII. MEC, 5G, and Possibilities

That we are on the cusp of a 5G revolution is an understatement. 5G is a generational transformation that will profoundly impact businesses and consumers across the globe. It shifts the fundamentals at all layers, creating an array of new opportunities for differentiation, cost re-structuring, and performance.

Organizations across industries are rapidly transitioning their ecosystems to be 5G ready. 83% of our survey respondents indicated they are already in the process of moving to 5G platforms in some shape or form.

Enter, MEC

One of the transformational developments in the evolution of 5G is multi-access edge computing (MEC). MEC moves the computing of traffic and services from a centralized cloud to the edge of the network and closer to the customer. Instead of sending all data to a cloud for processing, the network edge analyzes, processes, and stores the data. Collecting and processing data closer to the customer reduces latency and brings real-time performance to high-bandwidth applications.

5G is encouraging many technology operators to evolve for advantages in speed, latency, penetration, and capacity. It's also bringing a fresh approach to quality-assured systems, connected devices, and interoperability (Figure 29).

MEC offers application developers and content providers cloud-computing capabilities and an IT service environment at the edge of the network. It also enables the convergence of IT and telecommunications networking.[13]

MEC on 5G technology opens a plethora of use cases across industries

MEC will enable new vertical business segments and services for consumers and enterprises. It will help enable a range of 5G use cases ranging from self-

Figure 29: Comparison between pre-5G and 5G network technology

	Pre 5G	5G	Assuring Transformation
Global Connections* *in billions*	7.8	31	**Multi-Access Edge Computing (MEC) Conformance** Distributed cloud infra, Dynamic orchestration, API's Service exposure Multi-Cloud delivery app platforms
Technology	Monolith OSS/BSS (Operating Support Systems, Business Support Systems)	Cloud-Native applications supporting granular services, edge compute solutions, access-agnostic software	
Product Set	Narrow Phones, Tablets, Wearables	Diverse IoT, Smart Cities/Home, High def AR/VR, Remote Health Monitoring, Autonomous Cars, Intelligent Transportation	
Network	Commercial / Package / IP based	mmWave / Slicing / MIMO (Multiple Input and Multiple Output)	

Source: Deloitte analysis.

driving cars that communicate with each other to ultra-rich streaming video and AR/VR experiences. Emerging MEC technology, with its advantages in data transfer rate, mobility, transfer delay, and number of terminal connections, will power the Internet of Everything (Figure 30).

Some of the most promising impact areas of 5G here include:

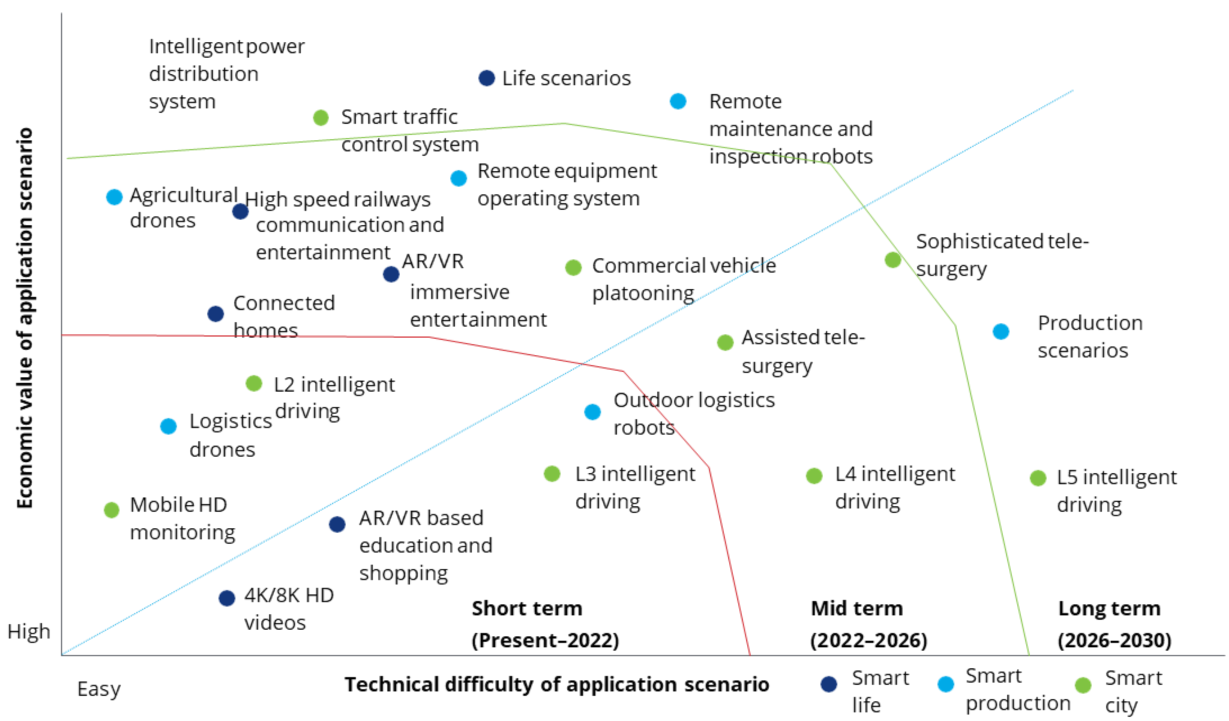
- **Smart cities**: 5G networks enable smart management and operation of mobility, grid management, and public security, driving sustainable city development.
- **Smart life**: The application of 5G technology will improve lives through improvements in healthcare, education, cultural and recreational consumption, and the connected home.
- **Smart production**: 5G will facilitate the evolution production and manufacturing by helping improve industrial and agricultural production conditions and increasing the feasibility of remote-controlled work to limit risks to humans.

Telecom operators are leading the way on the development of 5G application scenarios that rely heavily on communications and will work with industry partners to explore complex MEC applications.

Testing challenges and possibilities with myriad use cases

MEC testing involves more than verifying lightning-fast download speeds, low latency, and expansive coverage density. It requires differentiated testing approach. End-to-end test solutions are playing a vital role in the

Figure 30: 5G application implementation timeline



Source: Deloitte analysis.[14]

development, deployment, and operational excellence of emerging 5G networks.

Building a test strategy for MEC on 5G is a challenge, since testing may involve millions of IoT sensors, autonomous vehicles, ultra-high throughput, beamforming, and more. A one-size fits all testing approach won't work. Unique challenges exit across security, connectivity and interoperability, data volume, and use case variety.

With testing scopes that span across devices and network applications, multiple test tools and simulators and an expansive test environment need to be set up and maintained.

How organizations are transforming quality and gearing up to be 5G-ready

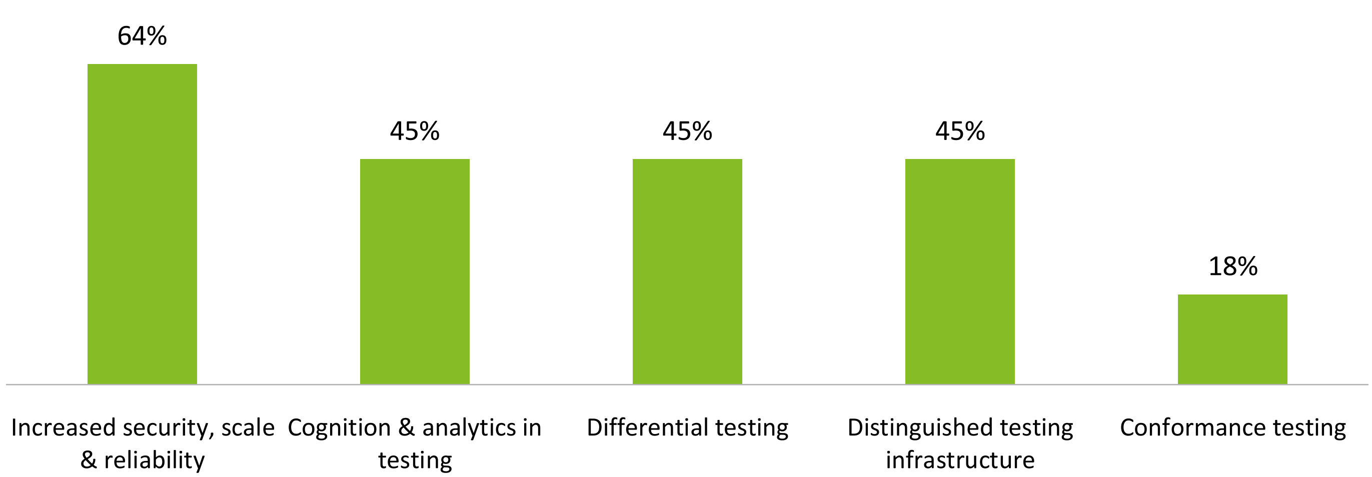
With 5G's service-based architecture and plethora of use cases, validating a 5G network for a good user experience gets challenging. Overhauling the test strategy is key to ensuring fast-time-to-market.

Our survey respondents echoed the need for differentiated testing approaches and toolsets, with an emphasis on ramping up security and reliability as the highest priority (Figure 31).

Shifting the fundamentals of quality assurance is necessary for the 5G era. There are 5 key areas to consider:

- **Managing customer experience**: Leverage cognitive techniques to aggregate KQIs and arrive at quality of experience (QoE). Focus on enhanced quality with seamless transition, high throughput and low latency data transmission, and high-image quality with least buffering and congestion.
- **Connected device enablement**: Ensure interoperability between devices and networks through simulation, API automation, and modeling languages. Leverage field and app testing with field trials, device voice and data performance, and mobility and user experience testing.
- **Network validation & rollout**: Ensure seamless interoperability, cloud testing and migration between multiple networks. Increase security and conformance testing. Maximize scale, resiliency, and reliability from hardware and software failure and updates.

Figure 31: Testing approaches and strategies organizations are employing to test systems/applications on 5G



Note: Respondents could select multiple answers.

- **Technology transformation**: Leverage intelligent automation (e.g., robotics, AI, and cognitive technologies) and continuous testing – through Agile, scalable, highly automated frameworks and approaches tailored to DevOps and robust enabling functions.
- **Connectivity & communication**: Run conformance tests on distributed infrastructure, dynamic orchestration, virtualized services, and multi-cloud delivery platforms.

Investments in new age testing techniques will unlock a wide variety of application clusters in 5G. An unparalleled test transformation approach and expertise ranging from lab to field can help improve visibility, accelerate time-to-market, and optimize 5G revenue streams.

The takeaway

5G is a generational technology redefining the communications ecosystem. Multi-access edge computing **(MEC) will play a key role in its evolution, unlocking a variety of 5G use cases** across industries from **self-driving cars to AR/VR**. Building a specific test strategy for MEC on 5G is a challenge, however. Organizations need to shift the fundamentals of quality assurance for the 5G era and should frame their strategies across 5 key pillars – **managing customer experience**, **connected device enablement**, **network validation & rollout**, **technology transformation**, and **connectivity & communication**.

XIII. Permissioned Granted, Blockchain

XIII. Permissioned Granted, Blockchain

Blockchain may be considered one of the biggest technology breakthroughs in recent history, similar to the advent of the internet in the early 1990s. Now, a decade has passed since it came into prominence. While blockchain was once classified as a technology experiment catering to cryptocurrency payments, it now represents a true agent of change with a variety of business use cases.

For instance, blockchain-based applications are driving energy trading and supply chain efficiencies in the power sector to streamline electricity sourcing, production, and delivery – aiming to reduce a global electrification deficit that affects over 1B people globally.[15] The global business impact of blockchain technology could be massive. Gartner projects the business value-add of blockchain will reach roughly $176B by 2025 and surpass $3.1T by 2030.[16]

Blockchain is a distributed ledger technology that has the potential to eliminate the pain points of traditional record-keeping, lower costs, reduce the need for centralized control, and streamline supply chains. It also has the potential to disrupt IT in ways not seen since the internet arrived.

Enter, permissioned blockchains

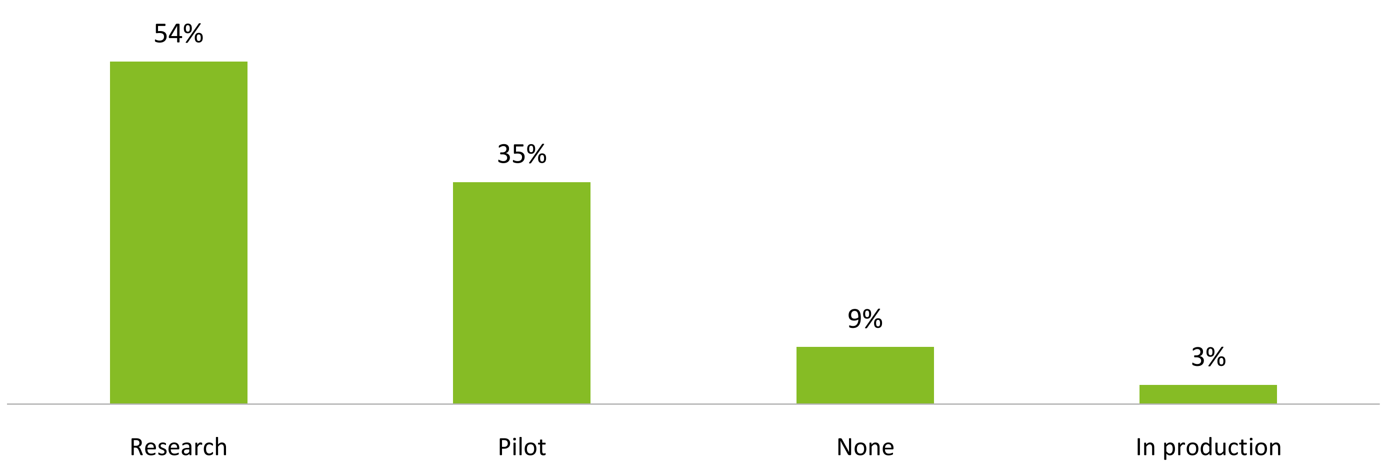
While there are three main types of blockchains – public blockchains, private blockchains, and permissioned blockchains – we'll zoom in on the permissioned type here. Permissioned blockchains are consortium-based – meaning they are operated by a group of organizations and require an invitation to participate. They combine aspects of public and private blockchains in the sense that they can involve multiple organizations, but they otherwise operate similar to private blockchains, preventing participation from anonymous participants.

Permissioned blockchains are fast, scalable, and highly customizable. A control layer runs on top of the blockchain that governs allowable actions performed by permitted participants. Governance typically permits new individuals to join the permissioned network after suitable verification of their identity. It also allocates select permissions for each individual.

Blockchain adoption

Our survey respondents indicated widespread interest and experimentation in blockchain technology

Figure 32: Progress of organizations in blockchain adoption



generally (Figure 32). 91% are at least in the research phase, with 35% in pilots and 3% in production.

Permissioned blockchains need a scalable and reliable operating model that works in a trust-minimized environment.

Widespread blockchain experimentation across industries is driving a high growth outlook for the technology, particularly in the permissioned blockchain type. Gartner forecasts that by 2023, viable permissioned blockchains will integrate tightly with public blockchains, using architectures like sidechains. And by 2028, permissioned and public blockchains will merge at the infrastructure level, supporting either public or private transactions.[17]

With the enthusiasm and overall growth potential for blockchain, effective testing is paramount. Permissioned blockchains need a scalable and reliable operating model that works in a trust-minimized environment.

Challenges in blockchain testing

A tailored focus on blockchain testing is paramount. We turned to our survey to see what challenges are top of mind for organizations (Figure 33).

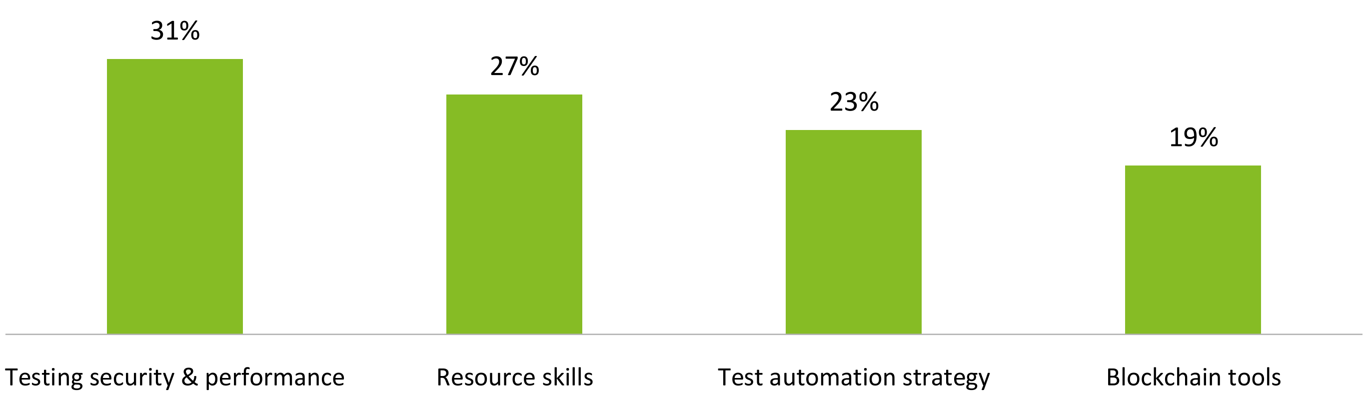
Performance and security issues: Security and performance testing was the top blockchain testing challenge highlighted, selected by 31% of respondents. Indeed, an estimated $1.9B in cryptocurrency was stolen in 2020.[18] The importance of security here is clear.

Managing and projecting blockchain transaction volume is complicated as the network continues to grow. Ensuring expected response times, for instance, with increases in the number of miners is a challenge. Security testing is crucial for blockchain networks because, once the identity layer is hacked, instantaneous transactions cannot be stopped.

Lack of experience and resource skills: Unsurprisingly, resource skills (27%) followed closely. Blockchain testing requires specific skills with a blend of technical and emerging domain knowledge that are at times hard to find.

Strategizing for testing blockchain applications is challenging. With a shortage of in-depth knowledge in the technology and tools required for testing smart

Figure 33: Key challenges organizations are seeing in testing blockchain applications



contracts, many organizations don't know where to start. For instance, replicating production environments for testing is expensive and complex. If not provisioned correctly, a new version of a block with all existing data might need to be inserted. This can make setting up a test environment quite tedious.

Improving blockchain testing approaches

Testing should be considered an integral part of the blockchain development lifecycle to ensure security and scalability of the blockchain ecosystem.

Testing of smart contracts is critical to ensuring security vulnerabilities are mitigated before go-live. Once a contract is deployed, no centralized body controls it – meaning it functions on its own. Defects need to be identified beforehand. Standardized test frameworks can be used to set up a proper test environment and test smart contracts.

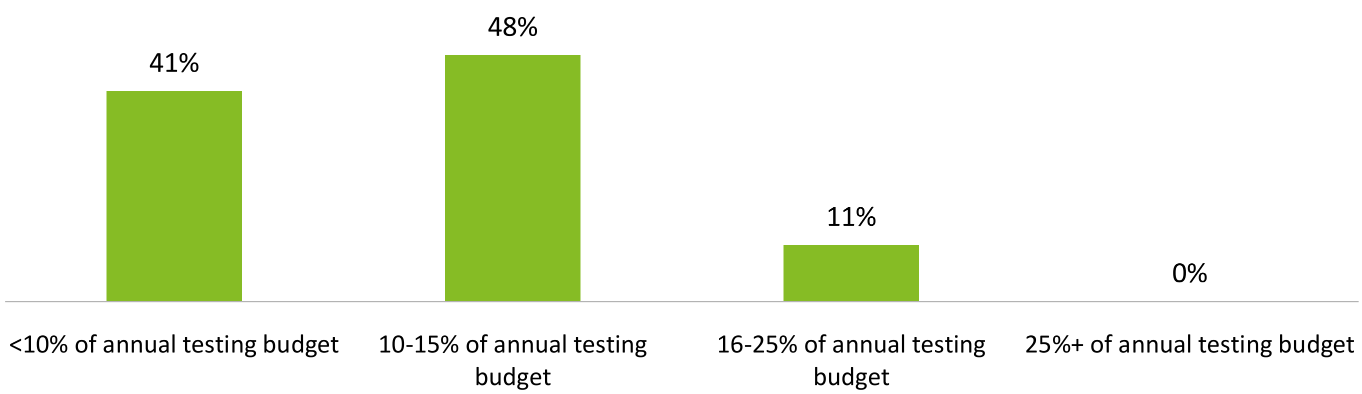
To stay ahead of progress and effectively perform course corrections, test teams should also focus on metrics like test coverage and trust improvement. KPIs help demonstrate how effectively the solution is achieving business objectives.

Because data added to the blockchain is immutable, integrations that ensure seamless data flow are also at a premium. Test automation frameworks can be used to automate the testing of APIs.

Current levels of focus

Right now, it's no surprise that testing practices are catching up to blockchain deployments given the novelty and complexity of the technology. Those that are experimenting with blockchain are allocating expected proportions of blockchain testing to their overall testing budgets (Figure 34). 48% of survey respondents are allocating between 10-15% per year. None are allocating more than 25%. Expect these proportions to increase over time as use cases proliferate and more blockchain-based applications are deployed.

Figure 34: Percentage of testing budgets organizations are allocating to blockchain-based applications



The takeaway

Permissioned blockchain technology, with time, will see a surge in use cases and applications across a growing variety of sectors. Due to potential security concerns, **digital identity is one potential area of high impact**. Lack of organizational experience and resource skills remain friction points to blockchain adoption. **Effective blockchain application testing** will not only **ensure quality** but also **minimize risks associated with inadequate knowledge** that could leave gaps during development.

XIV. About the Sponsors

Our insights can help you take advantage of emerging trends. If you're looking for fresh ideas to address your challenges, let's talk.

Reach out for a discussion

Rohit Pereira
Principal, Deloitte Consulting LLP
ropereira@deloitte.com

Avneet Chatha
Managing Director, Deloitte Consulting LLP
achatha@deloitte.com

Tejas Vijay Desai
Principal, Deloitte Consulting LLP
tedesai@deloitte.com

Saurayan Chaki
Senior Manager, Deloitte Consulting LLP
schaki@deloitte.com

Chaithanya Kolar
Managing Director, Deloitte Consulting LLP
ckolar@deloitte.com

Jeremy S Scott
Managing Director, Deloitte Consulting LLP
jscott@deloitte.com

Vivek Balakrishna Pai
Managing Director, Deloitte USI Consulting LLP
vivepai@deloitte.com

Authors

Krishna Duggyala, Aswin Natarajan, Mukesh Gupta, Noor Khoja, Neeraj Rana, Julker Nain, Ahed Mossolly, Sandeep Gorai, Rebicca Mandal, Srinath Narayan, Sandeep Parashar, Prabhanjan Nagendra, Pallabi Chakraborty, Srinidhi Magge, Praveena Gunda, Reema Rajput, Antony Chandy, Roshan Shetty, Ritesh Merchant

Contributors

Rajat Rai, Nitish Jain, Senthil Subbiah, Ganga Narayanasamy, Harshit Garg, Swati Seth

XV. Endnotes

1. Joachim Herschmann et al., *Gartner Magic Quadrant for Software Test Automation*, Gartner, November 2019.
2. *2020 Cloud Computing Study*, IDG, June 8, 2020.
3. "Spending on Public Cloud IT Infrastructure Surpasses Spending on Traditional IT Infrastructure for the First Time in the Second Quarter of 2020, According to IDC," IDC, September 29, 2020.
4. William Clark et al., *Predicts 2018: Application Development*, Gartner, November 21, 2017.
5. Dr. W. Edwards Deming, *Out of the Crisis*, The MIT Press (reprint), 2000.
6. Manjunath Bhat et al., *Predicts 2020: Agile and DevOps Are Key to Digital Transformation*, Gartner, December 5 2019.
7. Katie Costello and Meghan Rimol, "Gartner Forecasts Worldwide Public Cloud End-User Spending to Grow 18% in 2021," Gartner, November 17, 2020.
8. Andrew Lerner, "The Cost of Downtime," Gartner, July 16, 2014.
9. Research and Markets, "Test Data Management - Global Market Trajectory & Analytics," Global Industry Analysts, Inc., April, 2021.
10. Diego Lo Giudice et al., *No Testing Means No Trust In AI: Part 1*, Forrester, November 2019.
11. Michael Desmond, "IDC: Global Spending on IoT to Top $1 Trillion 2022," ADT Mag, June 14, 2019.
12. Rajkumar, "Crowdsourced Testing Guide for Companies and Testers," Software Testing Material, August 31, 2020.
13. Deloitte, *5G Ecosystem, The digital haven of opportunities*, September, 2019.
14. Ibid.
15. Kathleen O'Dell et al., "Powered by blockchain – Reimagining electrification in emerging markets," Deloitte Insights, August 1, 2018.
16. John-David Lovelock et al., *Forecast: Blockchain Business Value, Worldwide, 2017-2030*, Gartner, March 2, 2017.
17. Avivah Litan, William Clark, *The Future of Blockchain: 8 Scalability Hurdles to Enterprise Adoption*, Gartner, September 4, 2018.
18. Jastra Kranjec et al., "Crypto Criminals Stole $1.9B in 2020, down from $4.5B in 2019," Finaria, March 10, 2021.

About Deloitte