The 12 Must-Dos For Achieving Continuous Software Testing

Say Goodbye To Testing As Usual

by Diego Lo Giudice and Bill Seguin

August 14, 2019

Why Read This Report

Application development and delivery (AD&D) leaders must not forget about the importance of QA. Teams need to develop software faster, but never at the cost of lower quality. So how can teams continue to push for faster development without letting testing hinder their progress? The answer: continuous testing. In this report, we introduce a dozen must-dos that will bring AD&D bring continuous testing practices up to speed.

Key Takeaways

To Deliver Continuously, You Must Test Continuously
AD&D pros can't allow testing to be the Achilles' heel of development, slowing down the whole process as a separate step of software development and delivery. If continuous delivery is your aim, make sure continuous testing is also top of mind.

Old Testing Strategies Must Go Away Forever
Parting is such sweet sorrow, but abandoning old testing strategies will be well worth it. The change that dropping traditional testing practices requires will be painful, but it's the only way forward.

Here Is Your New Testing Textbook
The new world of testing affects everything: It comes with new day-to-day practices, huge changes in the testing organization from managers to testers to developers, and new supporting tools and technologies. Read our 12 must-dos to find out more.

The 12 Must-Dos For Achieving Continuous Software Testing

Say Goodbye To Testing As Usual

by Diego Lo Giudice and Bill Seguin
with Christopher Mines, Andrew Dobak, and Andrew Reese

August 14, 2019

Table Of Contents

Related Research Documents

How Developers Deploy Test Automation To Enable Continuous Software Delivery

The Path To Autonomous Testing: Augment Human Testers First

Shift Performance Testing Left To Streamline App Delivery

FORRESTER®

Forrester Research, Inc., 60 Acorn Park Drive, Cambridge, MA 02140 USA
+1 617-613-6000 | Fax: +1 617-613-5000 | forrester.com

Traditional Software Testing Has No Place In Modern App Delivery

Enough is enough. Companies looking to modernize AD&D need to sacrifice old ways of testing software. Old-school testing creates a bottleneck that Agile and DevOps methodologies can't handle. The only way to accelerate development is through new practices, ways of organization, and technologies — and tough cultural change for testers and developers. These aim to make testing:

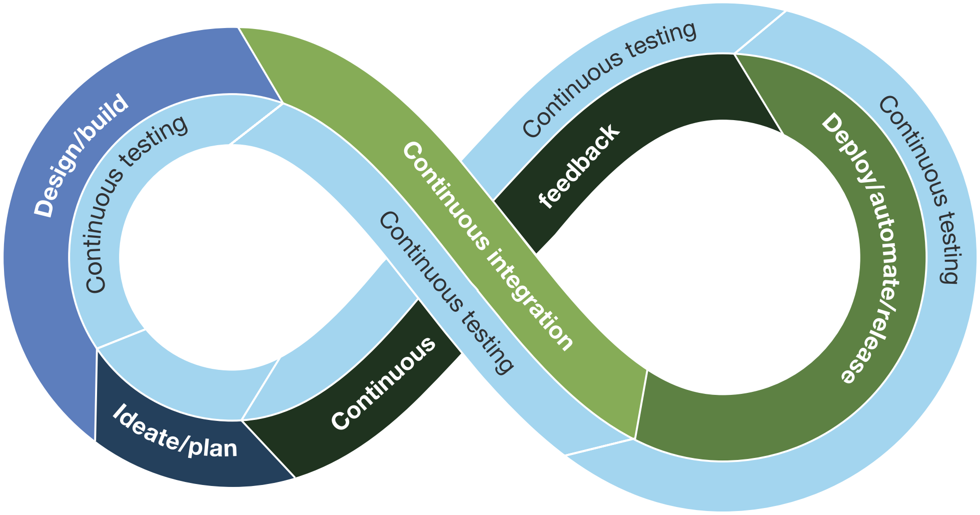
- **Continuous.** Obviously, continuous testing needs to be continuous. In Agile and DevOps environments, you should treat testing like a first-class citizen, infusing and executing it from beginning to end of the software development lifecycle (SDLC) (see Figure 1). As continuous delivery becomes the norm for application delivery teams, testing has to follow suit.

- **Incremental.** The idea of incremental development, releasing minimum viable products now and readying improvements for future sprints, lies at the heart of Agile and DevOps methodologies. Testing must shift to reflect the iterative nature of software development. This means making testing a help instead of a hindrance. AD&D leaders can't allow testing to form a bottleneck right before deployment. It must start when product owners and business analysts design new ideas, continue as developers build code, and ultimately ensure that the code will operate as closely as possible to users' expectations in production.

- **Fast and high quality.** While modern testing places a high value on moving quickly, leaders shouldn't forget the importance of quality. However, modern testing changes how testing teams need to approach QA. They won't have days or weeks to walk through every test manually, covering every edge case as release time nears. Instead, testers will need to check for quality earlier in the cycle. All stakeholders need to advocate for quality, optimize what to test, and automate as efficiently as possible — testing faster and better with less.

- **Lean and data driven.** As teams move from sprint to sprint, it's crucial to find areas for improvement and cut out waste in AD&D processes. The same holds true for testing teams, which must also adopt a Lean mindset. They need to keep focusing on how to get the most coverage with the least amount of effort, spotting inefficiencies and changing how they approach their job. To do this, AD&D and QA leaders need to arm themselves with data. Testers need to gather metrics and monitor application use and performance in production to close the feedback loop and drive focused testing.[1]

FIGURE 1 The Continuous Testing Möbius Loop



The 12 Must-Dos To Adopt Continuous Testing

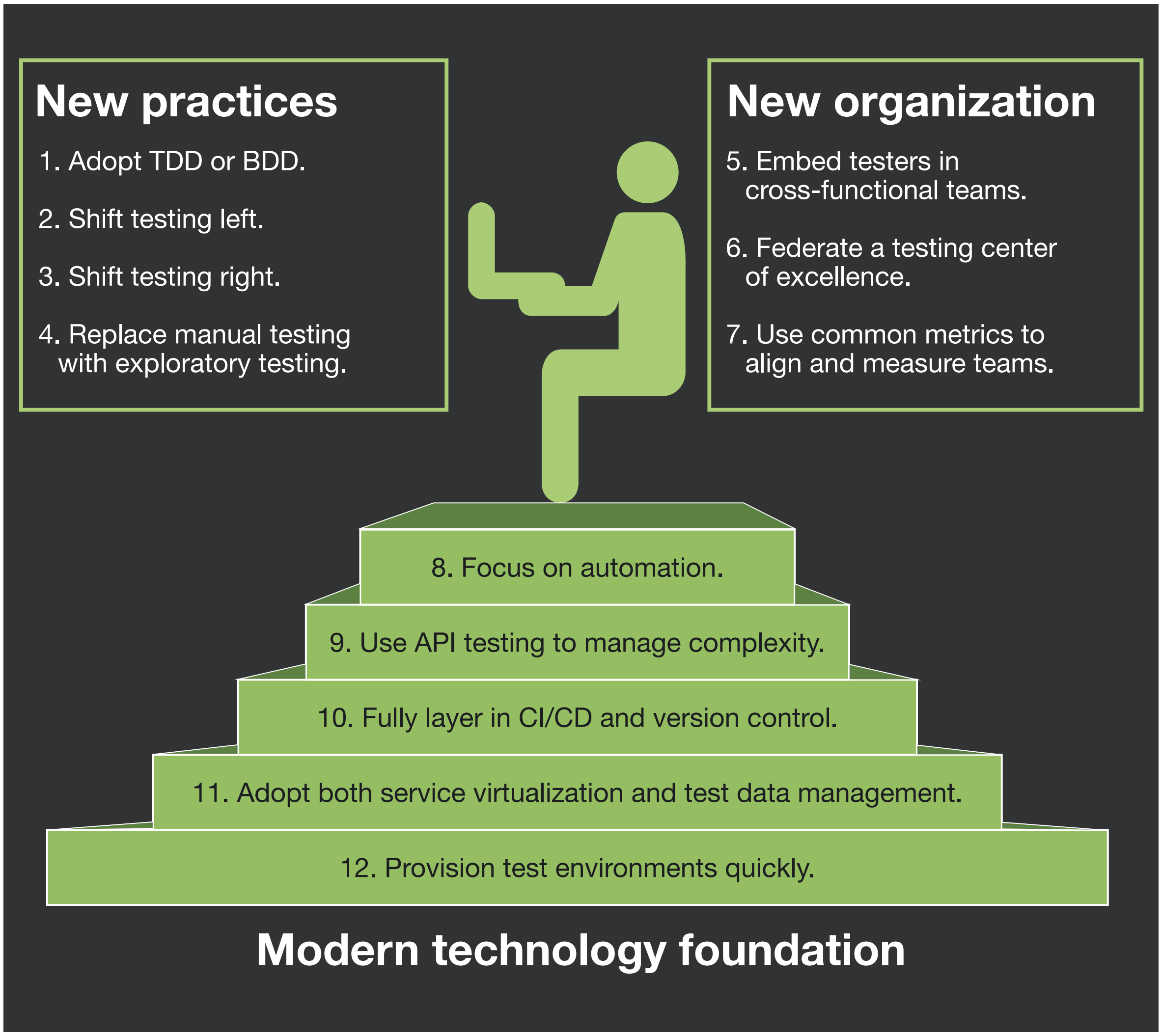
Those who find themselves at the trailhead of updating their testing practices face a long journey. Leaders need to do many things to ensure their move to continuous testing is successful. The 12 must-dos fall into three categories (see Figure 2):

- **New practices that change day-to-day operations.** In addition to transformational, Big Bang changes, AD&D teams will need to make more tactical changes to the way they work. By adopting new practices and abandoning old ones, they can innovate at the grassroots level and clear the way for broader innovation.

- **Organizational improvements that change relationships and responsibilities.** People are the most important focus when undergoing transformations — and also the biggest barrier to overcome. When continuous testing is the goal, centralized testing centers of excellence (TCoEs) as we know them don’t cut it. Continuous delivery requires servant leadership.[2] Instead of managing timelines, leaders need to focus on busting silos, embedding testers in product teams, and democratizing tester responsibilities.

- **Foundational preparations that set the stage for success.** These must-dos set up AD&D leaders for success in their transformation ahead. Continuous testing upends the status quo. As such, it’s important to give testers and developers a basis for adapting to this new world of testing more easily.

FIGURE 2 The 12 Must-Dos For Continuous Testing



New practices

1. Adopt TDD or BDD.
2. Shift testing left.
3. Shift testing right.
4. Replace manual testing with exploratory testing.

New organization

5. Embed testers in cross-functional teams.
6. Federate a testing center of excellence.
7. Use common metrics to align and measure teams.

8. Focus on automation.
9. Use API testing to manage complexity.
10. Fully layer in CI/CD and version control.
11. Adopt both service virtualization and test data management.
12. Provision test environments quickly.

Modern technology foundation

TDD: test-driven development; BDD: behavior-driven development; CI/CD: continuous integration and continuous delivery

New Practices Change Day-To-Day Operations

New practices for bringing continuous testing into Agile and DevOps contexts mean new behaviors and new technology tools.[3] The best development teams:

1. **Adopt TDD or BDD.** Use behavior-driven development (BDD) to give business product owners and developers a common language for sharing quality requirements. Setting proper, optimized, and clear test cases and user scenarios upfront sets the stage for successful business and technical functional coverage. Test-driven development (TDD) is also picking up; teams that adopt it modify their SDLC, defining the tests first and then writing the code to pass them.

   How are TDD and BDD different? Organizations mistakenly use these terms interchangeably. To put it simply, BDD requires teams to shift their focus, while TDD requires them to flip their whole approach to development on its head. Depending on their appetite for change and risk, it makes sense for some organizations to favor TDD and others BDD only.

2. **Shift testing left.** Continuous testing means that what was once an isolated step late in the SDLC now permeates all aspects of it. Shifting left means defining and executing unit testing, functional tests, and nonfunctional tests early on in the lifecycle. Developers need to take on some testing responsibilities, applying their skills to writing automated tests. Getting devs to think about testing code quality will get them thinking about writing higher-quality code from the start. Manhattan Associates found that "by shifting left, we got developers to understand quality from the onset and start testing in the early stages of the SDLC."

   Our interviewees stressed that shifting left doesn't just mean doing testing as usual earlier in the SDLC. Yes, it involves starting to test earlier on in the SDLC. But it also means adopting new practices for modernizing and simplifying testing — following the Practical Test Pyramid approach, for example.[4]

3. **Shift testing right.** Continuous testing also means that development and product teams must change how they monitor the software they build while it's in production. Think of this as testing in production, looking at how applications behave in the wild and gathering valuable data on customer preferences and user behavior. Shifting right can also mean testing different versions of features in production with A/B multivariate testing tools and running chaos-engineering tests to evaluate resilience.[5] The stakes are much higher when testing in production, so make sure you minimize risks and use feature flags to revert to previous versions quickly.

   Remember to close the loop by putting data to use. Share the data with the team that's ideating and prioritizing user stories for the next sprint. Even better, have a tester participate in that ideation process. By combining shifting left and shifting right, you can close the loop and truly make testing continuous.

4. **Replace manual testing with exploratory testing.** Manual testing won't go away on the journey to continuous testing, but it will certainly look different. Instead of retesting components they've already vetted in previous sprints, delivery teams will restrict manual testing to newly written features. The goal is to manually test only the edge cases — the hard business and technical ones that are too costly to automate.[6]

   You will need fewer experts in manual testing. Most Agile teams Forrester sees have an expert-tester-to-developer ratio of 1 to 2 or 1 to 3. Forward-thinking companies will put their expert testers through training and choose UI-led scripting tools (including Selenium) to help them write automation scripts. This trend might revert as more sophisticated AI and machine learning (ML) capabilities augment subject matter experts, driving toward automation and autonomous testing.[7]

Organizational Improvements Change Relationships And Responsibilities

Forget the highly segregated world of testers who sit in a testing CoE, isolated from product owners, developers, and the rest of the team. As everyone becomes a steward of quality, testing teams need to organize differently. They must:

5. **Embed testers in cross-functional teams.** Instead of sitting in their own organization, testers must sit in squads or scrum teams with developers and product owners. Amit Shah, director, enterprise quality and release management at United Airlines, attests to this approach: "[In] every org I've been in, success means not being siloed, working with app dev hand in hand but also being connected to pods that do most of the testing." This arrangement engages the tester early on in the process — during ideation, planning, development, and deployment. Design, execution, and orchestration of most testing will happen in the team, not centrally.[8]

   Embed testers on the dev teams even when your devs are taking on more testing, because shifting testing into the hands of developers isn't happening fast enough. Also, developers don't like testing at all (only about 53% of them even unit test their code) — or only like testing happy paths.[9] Embedding a tester can help developers get involved in testing more quickly while ensuring the team gets testing done — and done right.

6. **Federate a testing CoE.** Some companies pool highly skilled testers in a collective CoE, while others take the opposite approach and distribute them across development teams. But companies should strike a middle ground — what we call a federated test CoE. This is a small, centralized group of highly skilled automation engineers who harvest best practices, design patterns, and tools to share with teams. Add skilled performance engineers, test data management (TDM) experts, and service virtualization testing (SVT) designers. Think of this team as a shared pool of consultants for the rest of your organization that handles the complex tasks individual product teams would have a hard time doing themselves. A federated test CoE's aim is to supply helpful tools and practices that make testing software easier, not do the testing itself.

As organizations adapting the Spotify model mature and adopt matrixed structures, testers will be part of chapters that focus on testing competencies and best practices (akin to a CoE) on one side — and on embedding operational testing in delivery teams (squads) on the other.[10]

7. **Use common metrics to align and measure teams.** Think about it: What do companies promote when they incentivize testing teams based on the number of bugs they detect before deployment? These teams will push to get as much time as they can to catch bugs; their pay depends on it. Additionally, they won't push developers to write higher-quality, easier-to-test code. Teams need to organize around metrics that will get testers, developers, and operations alike focused on the outcomes the business actually wants.[11]

   And which metrics are those? Track velocity sprint over sprint, and make sure delivery teams are addressing more user stories more quickly. Balance this with quality metrics like bugs found in production or decreased mean time-to-repair. Combining these metrics with overall business value metrics (think online sales or web traffic) will give leaders a better view of how well application performance testing is working overall. At United, digital technology teams actually measure revenue, accounting for the incremental revenue their online products and features yield when customers upgrade or take advantage of a promotion.

Foundational Preparations Set The Stage For Continuous Testing

Cross-functional teams with embedded testers will need a sturdy foundation to build their continuous testing practice on. To do this:

8. **Focus on automation.** And focus on it relentlessly. Testers and developers need to find ways to automate whatever they can during software testing — including test creation/design and orchestration, not just execution. Optimizing test creation and design saves time and increases automation coverage. To increase their level of automation and leverage new features, teams need to embrace new technologies as they emerge.

   AI is changing what we can automate. Companies like Eggplant and Conformiq, as well as a number of testing services companies, are using AI to automate creation of tests from requirements. Other tools, like Diffblue Cover and source-d, can generate up to 60% of unit tests by analyzing the lines of code in a feature. AI can also facilitate bug clustering — relating bugs directly to code and the developers in charge of fixing them.[12] So we expect AI and ML will augment business, technical, and developer testers, not replace them.

9. **Use API testing to manage complexity.** Software testing needs to change along with the composition of the applications it's testing. Modern applications are made up of microservices that are loosely coupled with other components. These applications rely much more on the connections between microservices, putting a premium on API testing.[13] Testing teams need to update the types of tests they run and classify them in a way that permits more automated unit and API testing and focuses less on manual UI tests.

More Agile and DevOps practices will increase the need for API testing. Agile encourages frequent changes to apps. In early sprints, most of these focus on improving UIs. As devs update the UI code, the automated test scripts that once worked will break — and the team will have to slow down and redo its work. Focusing on the process behind the UI by replacing UI-led automation testing with API-led automation testing will help to curb this.

10. **Fully layer in CI/CD and version control.** Embed test automation into continuous integration and continuous delivery (CI/CD) tools like Jenkins or GitLab. These integrations will allow developers to initiate automated unit and performance tests at build time. Leaders should also look to version all test assets in source code management tools like GitHub to help them iterate from sprint to sprint, run automated regression tests, and maintain the resulting automation assets.

    Test management will be disrupted. Tools for doing this used to rule the roost, but they're becoming less and less relevant.[14] Continuous delivery delegates most test management functions to continuous integration tools. Vendors of standalone test management tools are getting snatched up by other companies like Tricentis, which acquired QASymphony, and SmartBear Software, which bought Zephyr. Other tools, such as Quality Center by Micro Focus (formerly HP Quality Center), aren't getting love from Agile teams anymore. For its part, Micro Focus has shifted attention to its ALM Octane product.[15]

11. **Adopt both service virtualization and test data management.** Advanced shops enhance their testing capabilities with robust TDM practices that feed synthetic or masked data that resembles real-world data into tests. In parallel, service virtualization tools introduce the concept of simulation in software, enabling teams to test with early designs and simulated assets instead of real-world ones. Both offer high value to AD&D and QA teams, especially if they leverage them together.

    In an Agile-plus-DevOps context, TDM has shifted to sophisticated synthetic test data creation, aligning test data and production data more closely. The EU's General Data Protection Regulation is pushing this trend, too, and new vendors like GenRocket and Datical are filling the gaps. Service virtualization helps create virtualized data, and it also helps Agile teams decouple testing dependencies. Large banks like Lloyds Bank are using the technology to automate complex integration scenarios.[16]

12. **Provision test environments quickly.** Teams investing time and treasure in becoming Agile, increasing automation, and iterating more quickly can't afford to go through the tedious process of provisioning a test environment. Testing teams need to be able to quickly spin up and tear down testing environments on demand.

    Cloud makes provisioning simpler and allows teams to implement a self-service approach that lets them spin up their own environments in minutes — or even seconds. But it doesn't come cheap. One government agency experienced this firsthand. It didn't matter if its test environments lived in the cloud: Since the protracted approval process stayed in place, provisioning an environment still took days. Provisioning test environments is the biggest cost for testing teams, so invest in ways to make it quicker.

Engage With An Analyst

Gain greater confidence in your decisions by working with Forrester thought leaders to apply our research to your specific business and technology initiatives.

Analyst Inquiry

To help you put research into practice, connect with an analyst to discuss your questions in a 30-minute phone session — or opt for a response via email.

Learn more.

Analyst Advisory

Translate research into action by working with an analyst on a specific engagement in the form of custom strategy sessions, workshops, or speeches.

Learn more.

Webinar

Join our online sessions on the latest research affecting your business. Each call includes analyst Q&A and slides and is available on-demand.

Learn more.

Forrester's research apps for iOS and Android.
Stay ahead of your competition no matter where you are.

Supplemental Material

Companies Interviewed For This Report

We would like to thank the individuals from the following companies who generously gave their time during the research for this report.

accelQ

API Fortress

Applause

Broadcom

Capgemini

Cigniti Technologies

Compuware

Cyara

Eggplant

Experitest

Functionize

Hexaware Technologies

IBM

Infosys

Micro Focus

Microsoft

Mindtree

NTT Data

Parasoft

Perfecto

Qualitest

ROQ

Sauce Labs

SmartBear Software

Tata Consultancy Services

Tech Mahindra

ThoughtWorks

Tricentis

United Airlines

Wipro

Worksoft

Endnotes

[1] For more on application performance monitoring, see the Forrester report "Shift APM Left For Better DevOps Performance."

[2] Longtime AT&T executive and consultant Robert Greenleaf introduced the concept of servant leadership in the 1970s. Unlike traditional leaders focused on accumulating power at the top, servant leaders focus first on the well-being of the people and communities they lead. Source: "What is Servant Leadership?" Greenleaf Center For Servant Leadership (https://www.greenleaf.org/what-is-servant-leadership/).

[3] For more information on how AI can augment the work software testing professionals do, see the Forrester report "The Path To Autonomous Testing: Augment Human Testers First."

[4] Find more information on the concept of the Practical Test Pyramid, which outlines what tests you should consider, at the following address. Source: Ham Vocke, "The Practical Test Pyramid," Martin Fowler, February 26, 2018 (https://martinfowler.com/articles/practical-test-pyramid.html).

[5] For more information on experimental development tools and how to use them within the SDLC, see the Forrester report "AD&D Pros: Stop Guessing And Start Experimenting."

Source: "Chaos Engineering Whitepaper," Gremlin (https://www.gremlin.com/chaos-engineering-guide/?utm_source=google&utm_medium=cpc&gclid=CjwKCAjw9dboBRBUEiwA7VrrzeH0FV_pjzdkdn3VN-B4tPBzx-2a_sIL8m3225Ko10I1Ae1pAihikxoCucAQAvD_BwE).

[6] See the Forrester report "The Forrester Wave™: Omnichannel Functional Test Automation Tools, Q3 2018."

[7] See the Forrester report "The Path To Autonomous Testing: Augment Human Testers First."

[8] See the Forrester report "Scaling Agile: Can The Spotify Approach Work For You?" and see the Forrester report "Put Product Management To Work In Software Development."

[9] Source: Forrester's Q3 2017 Global Agile Software Application Development Online Survey.

[10] See the Forrester report "Scaling Agile: Can The Spotify Approach Work For You."

[11] See the Forrester report "Build The Right Software Better And Faster With Agile And DevOps Metrics."

[12] See the Forrester report "The Path To Autonomous Testing: Augment Human Testers First."

[13] For more information on API testing, its benefits, and the tools available for doing it, see the Forrester report "The Forrester Tech Tide™: Continuous Software Delivery, Q3 2018."

[14] See the Forrester report "The Forrester Tech Tide™: Continuous Software Delivery, Q3 2018."

[15] Source: "ALM Octane Enterprise Agile Delivery," Micro Focus (https://www.microfocus.com/en-us/products/alm-octane/overview).

[16] Source: "Lloyds makes the jump from service virtualization to full DevOps," QA Financial, March 1, 2017 (https://www.qa-financial.com/articles/lloyds-makes-jump-service-virtualization-full-devops).

FORRESTER®
CHALLENGE THINKING. LEAD CHANGE.

We work with business and technology leaders to develop customer-obsessed strategies that drive growth.

PRODUCTS AND SERVICES

- Core research and tools
- Data and analytics
- Peer collaboration
- Analyst engagement
- Consulting
- Events

Forrester's research and insights are tailored to your role and critical business initiatives.

ROLES WE SERVE

Marketing & Strategy Professionals

CMO

B2B Marketing

B2C Marketing

Customer Experience

Customer Insights

eBusiness & Channel Strategy

Technology Management Professionals

CIO

› Application Development & Delivery

Enterprise Architecture

Infrastructure & Operations

Security & Risk

Sourcing & Vendor Management

Technology Industry Professionals

Analyst Relations

CLIENT SUPPORT

For information on hard-copy or electronic reprints, please contact Client Support at +1 866-367-7378, +1 617-613-5730, or clientsupport@forrester.com. We offer quantity discounts and special pricing for academic and nonprofit institutions.

Forrester Research (Nasdaq: FORR) is one of the most influential research and advisory firms in the world. We work with business and technology leaders to develop customer-obsessed strategies that drive growth. Through proprietary research, data, custom consulting, exclusive executive peer groups, and events, the Forrester experience is about a singular and powerful purpose: to challenge the thinking of our clients to help them lead change in their organizations. For more information, visit forrester.com.