IDC ANALYZE THE FUTURE

IDC TECHNOLOGY SPOTLIGHT

Sponsored by: Hexaware Technologies

This IDC Technology Spotlight examines the use of autonomous testing by quality assurance (QA) teams and explores the role that autonomous testing services can play in aiding organizations with enhancing and streamlining their QA activities.

Using Autonomous Testing to Power Next-Gen Application Delivery

October 2020

Written by: Peter Marston, Research Director, Worldwide Intelligent Application Services

Introduction

Today's application delivery teams face increasing pressures to deliver application value faster amid ever-changing business requirements. As a result, organizations must approach and execute quality assurance (QA) differently. QA disciplines must evolve from a cost-driven activity to a value-driven activity. Delivery methodologies, such as Agile and DevOps as well as the Agile-Waterfall hybrid model, have emerged to meet the evolving needs of business.

AT A GLANCE

KEY STATS

- On average, 40% of organizations' application portfolios are built and managed using microservices and containers today.
- Organizations expect that more than 50% of their application portfolios will be built and managed using microservices and containers in five years.

However, while modern application delivery methodologies have grown in popularity and adoption, many organizations struggle to align and optimize testing resources amid increased testing volumes and shortening test cycle times. Application delivery teams now need to be smarter about what they test and lean more on automation principles and techniques. Teams must be better equipped to predict issues and prevent them from surfacing before applications are deployed while they test more and test faster.

This paper reviews the use of machine learning (ML) to enhance the capabilities of QA teams as they deal with modern application delivery challenges, and it explores the role that autonomous testing can play in aiding organizations with enhancing and streamlining their application testing and QA activities.

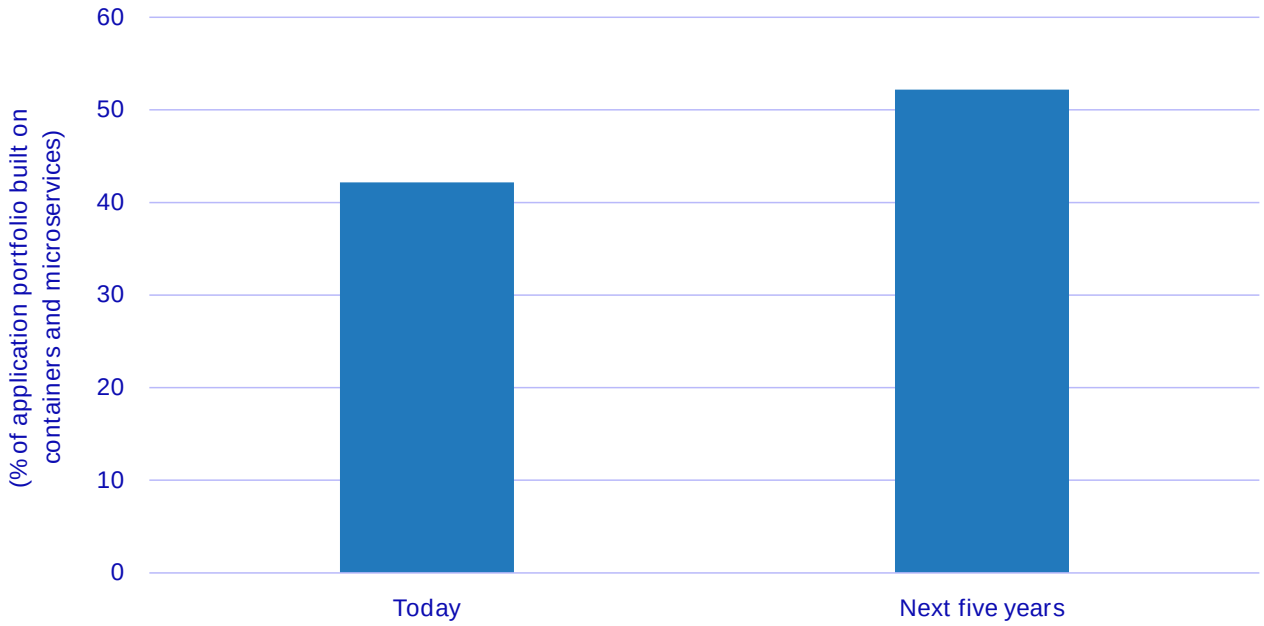
Major Trends in Application Life-Cycle Management Help Delivery Teams Better Meet Business Needs

Key trends emerging in application life-cycle management are creating opportunities to approach testing in a more intelligent and streamlined fashion. IDC has observed the following trends:

- **Application portfolios are expanding.** Application portfolios are growing and forcing QA organizations to change. According to IDC's *U.S. Application Services Survey* conducted in 4Q19, organizations have an average of 198 applications in their portfolios. In five years, organizations expect the size of their application portfolios to grow an average of 33%. IDC research also finds an increasing intent to remove organizational silos through increased application integration across varying application types. As a result, organizations need better and faster ways to execute testing tasks to support the massive amount of integration work from higher levels of integration across more applications in the future.
- **Use of DevOps and containers is gaining momentum**. Not surprisingly, amid the changes in portfolio size, organizations are also evolving their application delivery processes. Many organizations have built and deployed their applications in a Waterfall fashion. These traditional approaches centered on amassing requirements, holistically, before progressing to the stages of design, building, testing, and deployment. Organizations that used a Waterfall approach expected that defects would be addressed and resolved before the applications were released into production and that application functionality would suit user requirements. However, such conventional approaches for application delivery were not without challenges. While Waterfall approaches aided with mitigating risk and creating an assembly line–like model, they tended to be time consuming and inflexible to changing requirements. In this regard, IDC has found that organizations have evolved and are continuing to evolve their application delivery practices. In fact, organizations estimate that an average of 40% of their application portfolios are built and managed using microservices and containers today, and they expect that more than 50% of their portfolios will be built and managed using microservices and containers in five years (see Figure 1).
- **Automation is no longer an option; it's a competitive advantage**. Use of containers and microservices has increased to help further unlock Agile and DevOps capabilities within organizations. Why? Because containers and container orchestration streamline and automate code delivery and help organizations achieve faster time to market for new application functionality through code use and reuse. IDC research has shown that 78% of early DevOps adopters have already invested in containers and container orchestration or plan to invest in containers and container orchestration in the next 12 months to better streamline and drive higher levels of automation in their application delivery. This means that while the benefits of faster code delivery have organizations excited about the speed that automation can bring, organizations must be aware of the implications that faster code builds and the increasing volume of code builds will have on their testing and QA teams and activities.

FIGURE 1: ***Evolution of Application Delivery***

Q ***What percentage of applications in your organization's application portfolio would you estimate is built on containers/microservices today, and what would you expect that percentage to be in five years?***



n = 400 (all respondents)

Note: Data is weighted by industry and by employee size.

Source: IDC's U.S. Application Services Survey, 4Q19

Progressive Shifts in Application Delivery Spawn New Challenges

While growth in application portfolios has contributed to organizations adopting more progressive methodologies to transform application delivery, the use of DevOps and Agile has forced application delivery teams to shorten cycle times for application deployment. Condensed testing cycles coupled with elevated expectations for faster application deployment translate to testing teams maintaining (or oftentimes expanding) testing scope while speeding up testing execution. This can lead to organizations becoming compromised over:

- **Governance models for shortened delivery cycles and increased release volume.** Adoption of new delivery methodologies has created greater challenges for traditional methods of application delivery governance. Conventional methods of governance struggle to accommodate modern delivery methods and can become compromised against pressures for ballooning development scope, faster application delivery, and more frequent application releases. As a result, organizations must explore new methods of delivery governance to mitigate risk, ensure proper test coverage, and uphold application quality standards.

- **Heightened time-to-value expectations for application deployment.** Even though DevOps and Agile delivery can help bolster application release speed, delivery teams can face pressures of testing traditional scope within more condensed time frames when line-of-business personnel are not familiar with or trained on how modern application delivery is different from traditional delivery. The fundamentals of backlog and requirements prioritization as well as user story trade-offs are concepts not easily understood by business stakeholders unless organizations have gone through large-scale transformation programs where major change management exercises underpin application delivery transformation. Organizations that are accustomed to conventional application delivery and switch to DevOps and Agile without having a robust change management program can expect the full scope from traditional delivery will be delivered faster than what they've previously experienced. This misconception can lead to unrealistic expectations and create divides between line-of-business and application delivery teams that can weaken the effectiveness that DevOps can bring.

- **Increased development and testing costs.** Less time to execute testing tasks can result in teams shrinking the number of tests they perform to deliver applications more quickly into production. However, running fewer test cases can expose applications to risks and vulnerabilities and cause post-deployment rework that can be more costly than fixing the problem prior to deployment.

Infusing Autonomous Testing as Part of Delivery Is More Critical to Avert Challenges

Although more progressive methods for application delivery have created more challenges for application testing, application delivery teams can overcome those hurdles by evolving their testing approaches through implementing autonomous testing to help pinpoint risk areas and eliminate defects from reaching production. Autonomous testing is the application of high degrees of automation testing using artificial intelligence (AI) and machine learning techniques for testing tasks and disciplines.

The combination of automation, machine learning, deep learning, and AI helps organizations create, execute, and refine tests. Moreover, autonomous testing enables organizations to learn from failed tests and make decisions on how new tests should be created and executed without the need for laborious human interaction. Through autonomous testing, application delivery and QA teams can circumvent and eliminate testing challenges in modern application delivery by:

- **Developing smarter testing strategies.** Application delivery teams need to employ more progressive testing strategies to optimize application delivery. Testing smarter ensures that coverage of major impact test cases isn't affected. Additionally, understanding where new development work has direct (and indirect) linkages to other facets of deployment can help application teams surface risks, and understand which testing tasks may be superfluous. Additionally, it can help assess and triage critical areas that must be tested manually and determine which tests can leverage automation to drive value.

- **Driving more predictability with testing.** Traditional testing approaches can prevent defects from being released into production but often fall short with effectively and efficiently understanding defect root causes and establishing effective means for defect prevention rapidly. Using autonomous testing in place of manual testing, by contrast, enables organizations to focus on value-added operational work — such as improving the testing processes and application architecture to understand the impact of code changes — and developing predictability in how application code changes affect production applications before tests are run. Employing higher levels of automation around test execution and other routines in application testing exercises helps application delivery teams focus more on value-added work in test strategy development, defect predictability and prevention, and impact analysis.

- **Leveraging machine learning to enhance current automation assets.** According to IDC research, organizations estimate that nearly 40% of their testing activities are automated today and expect to automate almost 50% of testing activities in three years. Introducing ML tools for code reviews, regression testing, and performance testing lets organizations execute testing tasks faster and identify systemic issues with code development that could have gone unnoticed. ML used on top of testing automation also helps organizations become more predictive in areas where certain tests drive more value, such as security testing. It also streamlines test script management and maintenance.

Benefits of Autonomous Testing

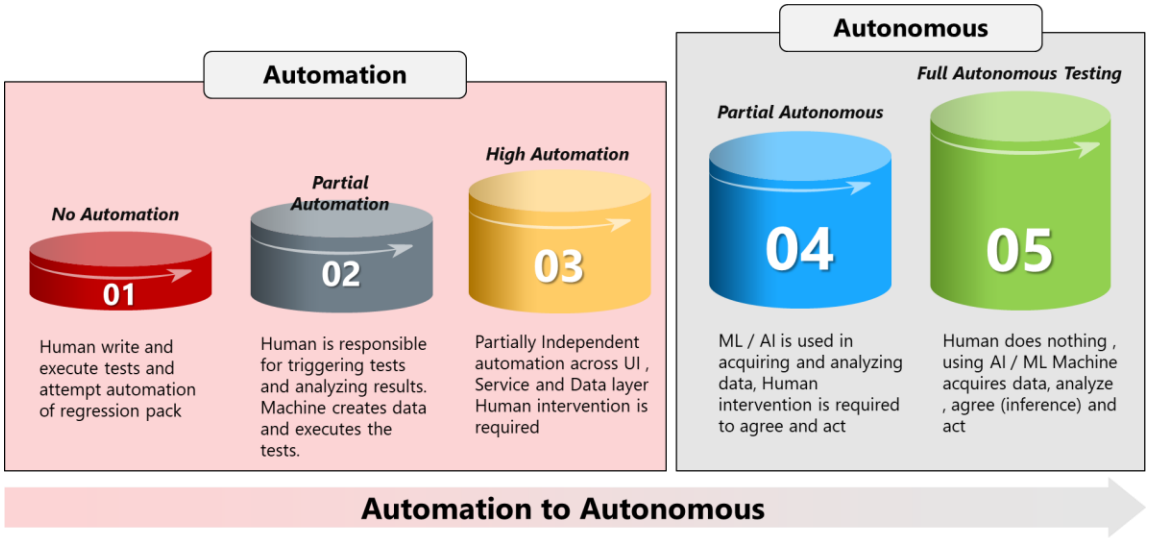
Employing autonomous testing as part of application delivery yields the following benefits:

- **Faster testing cycle times.** Using autonomous testing within QA activities enables application delivery teams to proactively prevent defects from surfacing versus finding and fixing them. Leveraging AI and ML to develop test cases and make decisions on how tests should be run when conditions change reduces labor time for test case generation and refinement. Additionally, by using ML and AI to power autonomous testing, organizations can establish foundations for standardization, from which they can build pillars for automation. That automation, combined with analytics that spot code issues before tests are run, can help organizations identify higher-priority testing tasks that can be fed back to improve upstream development activities, enhance resource utilization, and eliminate superfluous costs that may emerge in application delivery.
- **Increased innovation.** Enhancing existing automation capabilities through autonomous testing can refocus QA and testing resources on more value-added work and innovation, steering them away from manual test case execution, monitoring, and defect finding. Evolving the role of testers from test execution and monitoring to test strategy development, business risk mitigation, and testing governance helps application delivery teams drive more value for their organizations and fosters a culture of innovation.
- **Cost savings and improved ROI.** Through autonomous testing, organizations can more effectively target and speed up cycle times for testing tasks, defect analysis, and remediation. This has a twofold effect. First, leveraging automation for testing reduces labor costs by redirecting tasks traditionally done by labor to automated execution. Second, labor can be assigned to other testing and application life-cycle management areas that require deeper levels of business logic, management, and industry expertise to drive higher resource productivity. By applying testing automation to a larger portion of testing areas and more deeply in the application portfolio, organizations can be positioned to generate significant cost savings over time to achieve higher cash flows and boost ROI.
- **Risk mitigation through intelligent business insights and predictive issue avoidance.** Incorporating autonomous testing within QA and application delivery can aid organizations with proactively identifying and preventing application defects from surfacing versus spending time researching and fixing root causes. Possessing the right analytics that spot code issues before tests are run or identifying high-priority testing tasks that circumvent application risks can help organizations improve upstream engineering activities, enhance resource utilization, and avoid unnecessary costs throughout the overall process of application delivery.
- **Expedited root cause analysis and bug triaging.** Autonomous testing helps organizations predict defect root causes and automatically triage bugs for remediation. Immediate identification of defect root causes helps eliminate delays and costs due to bug investigation. Autonomous testing also streamlines bug triage activities by enabling engineering teams to focus on fixing defects that significantly impact an application release. Accelerating these activities speeds defect resolution and eliminates waste.

Considering Hexaware's Autonomous Testing Profile

Even with the highest level of automation, testing activities such as impact analysis, maintenance, identifying optimal tests to run, and proactive learning from different data sources such as production logs to preempt issues are still dependent on human intervention. Autonomous testing helps customers mature from high automation to fully autonomous testing, which makes testing independent of human intervention through self-learning and self-healing (see Figure 2).

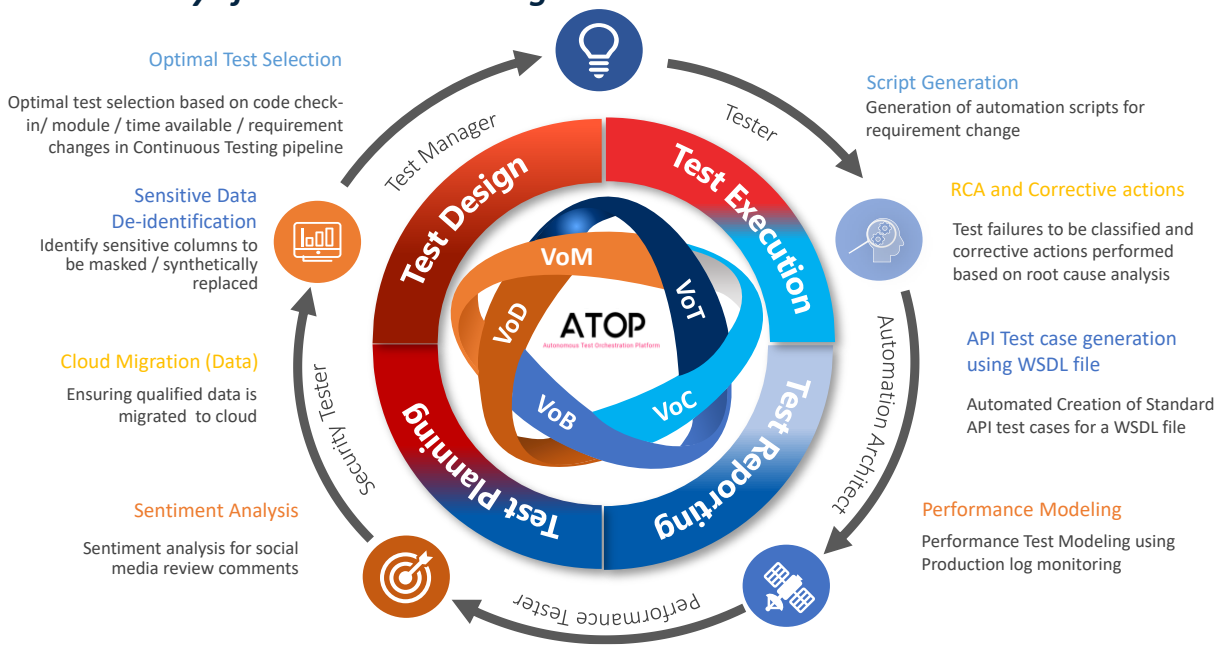
FIGURE 2: ***Automation to Autonomous***



Source: Hexaware, 2020

To enable autonomous testing for its customers, Hexaware has identified 100+ use cases across functional and nonfunctional testing catering to different personas and phases of the testing life cycle. These use cases leverage data from various sources, such as Voice of Customer (e.g., app ratings and comments), Voice of Machines (e.g., log files), Voice of Tests (e.g., test results), Voice of Business (e.g., user stories), and Voice of Developer (e.g., code check-in inputs), to make a given testing activity independent of human intervention. Figure 3 summarizes the use cases.

FIGURE 3: ***Summary of Autonomous Testing Use Cases***



Use Cases grouped by Personas across Testing Types

Test Type / Personas	UI Testing	Services Testing	Data Centric Testing	Performance Testing	Security Testing	Test Data Management	*Total*
Test Analyst (SDET)	23	13	9	8	0	8	*61*
Security Tester	0	0	0	0	6	0	*6*
Test Architect	3	2	3	2	2	0	*12*
Test Manager	7	7	6	10	5	0	*35*
Total UC by Test Type	**33**	**22**	**18**	**20**	**13**	**8**	**114**

Source: Hexaware, 2020

All these use cases are implemented using Hexaware Autonomous Test Orchestration Platform (ATOP). ATOP utilizes machine learning and deep learning algorithms combined with natural language processing techniques.

ATOP enables the acquisition of massive amounts of data (Acquire) generated during various phases of the application life cycle categorized as Voice of Customer, Voice of Machines, Voice of Tests, Voice of Business, and Voice of Developer; analysis of the data (Analyze); creation of inferences (Agree); and decision making (Act) leveraging AI/ML technologies.

ATOP has out-of-the-box integrations with leading application life-cycle management tools, application performance management tools, log aggregators, defect management tools, and app stores. ATOP aggregates data from these tools; it applies the built-in AI and ML algorithms to analyze the data and come up with insights and predictions to build test case libraries, test bodies, and all artifacts that are usually created manually throughout the testing life cycle.

Some of the use cases that are already lined up for implementation are as follows:

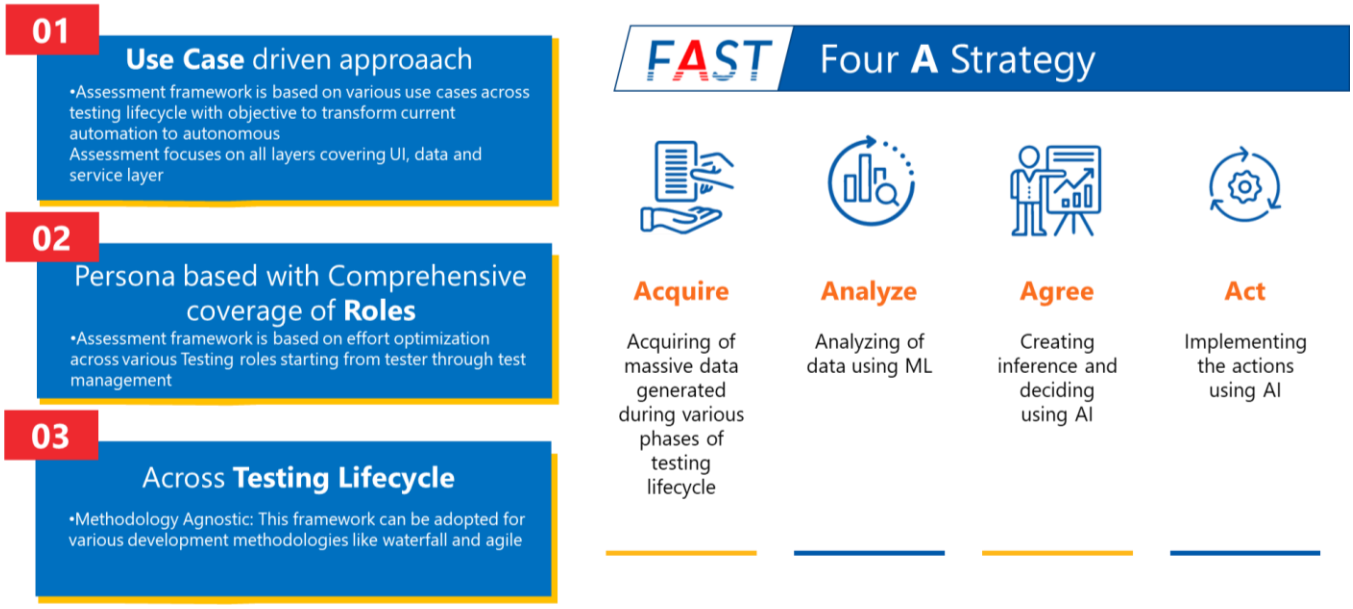
- Generating feature files and automation scripts from business requirements (Voice of Business)
- Generating automation scripts and test data from production logs to reinforce the regression bed (Voice of Machines)
- Autonomously grouping test failures to reduce test execution analysis efforts (Voice of Tests)
- Autonomously identifying the impact of a change in code and maintaining the automation scripts (Voice of Developer)
- Autonomously identifying features of the application with poor customer feedback through sentiment analysis and strengthening test coverage (Voice of Customer)

ATOP can be deployed in the following modes:

- A full-service platform enables automation until the customer achieves autonomous maturity. This deployment includes the script-less automation platforms to enable rapid automation across functional and nonfunctional testing types and application layers such as:
  - Front-end layer (including web, mobile, thick client, and chatbot)
  - Service layer (SOA, RESTful, and microservices)
  - Data layer
  - Intuitive user interface (UI) and AI capabilities to achieve the autonomous testing use cases
- Autonomous services through the ATOP platform along with intuitive UI capabilities work with a customer's current functional and nonfunctional automation infrastructure and frameworks.
- Autonomous services consumption through the ATOP APIs leverage a customer's current automation framework UI to achieve autonomous testing.

For implementing autonomous testing, Hexaware takes a consulting-led approach where the company's test consulting team performs a detailed due diligence over a period of up to three weeks using an Autonomous Test Maturity Assessment (ATMA) framework. ATMA is a comprehensive assessment framework that evaluates a customer's maturity against every use case covering functional and nonfunctional testing catering to different personas and phases of the testing life cycle (see Figure 4).

FIGURE 4: ***Autonomous Test Maturity Assessment Framework Overview***



Source: Hexaware, 2020

The ATMA framework helps the Hexaware team assess the current autonomous testing maturity levels of the customer on a scale of 1 to 5 and accordingly provide a detailed road map for progressing to the next higher level of maturity.

Key Differentiation

The company believes it's ahead of its time compared with peers in the industry in terms of having a clear vision for making autonomous testing real by identifying a comprehensive set of use cases and having an integrated test orchestration platform in the form of ATOP to implement these use cases. The plug-and-play architecture enables customers to either go with Hexaware's automation solutions or integrate their existing or third-party automation solutions. They even have the option to consume the platform's features as services.

The ATMA framework is one of the biggest differentiators that help customers get a grounded view of their current state of maturity for autonomous testing.

Value Proposition

According to the company, the complete implementation of autonomous testing use cases would not only deliver unprecedented speed and quality in DevOps environments but also substantially reduce QA efforts anywhere in the range of 40% to 70%, depending on the current QA maturity of the customer.

Challenges

Constant and swift changes in business and technology environments are imposing greater pressures on service providers to provide exceptional service delivery. Moreover, client expectations of application services performance have increased. IDC research has found that application environments for development, testing, and production are becoming more complex, and highly federated infrastructure environments — which are extending from on premises to host based to hybrid clouds and edge computing — have created new sets of challenges for services providers.

Hexaware not only must ensure application functionality and performance amid varied hosting and infrastructure environments but also must be prepared to support evolving and complex needs for test environment management to help clients address challenges that more complex infrastructure and hosting environments may pose. Application services providers such as Hexaware that continually invest and reinvest in their solution offerings to span a wide range of testing disciplines as well as possess deep, integrated technical architecture expertise through the application stack stand to build and gain competitive advantages against their rivals.

Conclusion

Employing autonomous testing as part of application life-cycle management helps organizations expand and unlock business capabilities and value. Through autonomous testing, organizations can be better positioned to suppress application and related business risks as well as put application functionality into users' hands more quickly. IDC believes implementing autonomous testing as part of application life-cycle management will grow in importance over the next several years to help organizations get a leg up on their competitors. To harness the full benefits autonomous testing provides, organizations must:

IDC believes implementing autonomous testing as part of application life-cycle management will grow in importance over the next several years.

- **Define clear and measurable goals and objectives.** Organizations should outline specifically what autonomous testing will and will not bring to their application testing activities and use these goals and objectives as the foundation for how they intend to implement comprehensive QA across the various facets of application life-cycle management.
- **Assess the existing state of automation within application testing.** Many organizations have "pockets" of automation within their application testing and deployment activities but struggle to link automation to end-to-end facets of their application life-cycle management. It's important to get a macro-level understanding of where automation can be applied within key testing functions and adjacent application life-cycle functions to understand maturity levels as well as to spot opportunities where automation and continuous application testing can drive more value in other areas of application life-cycle management.
- **Develop a governance and overarching performance monitoring model**. Even with increased autonomous computing in application testing, organizations still need to develop a governance and oversight model to monitor performance and explore areas for further automation and deeper adjacencies to other application life-cycle management areas. While the goals of autonomous testing are to enhance efficiencies, accelerate application testing cycles, and eliminate defects and costs, organizations still must develop escalation paths and define measures of success to ensure autonomous testing activities and tools provide value. Establishing a set of resources that will guide, direct, and manage the program, such as a steering or management committee, will ensure that the program has line-of-business representation and buy-in.

About the Analyst

Peter Marston, *Research Director, Worldwide Intelligent Application Services*

Pete Marston is Research Director for IDC, responsible for the Worldwide Intelligent Application Services practice. He develops research focused on modern application delivery and the life cycle of application services markets, which include Custom Application Development (CAD), Testing, Application Management (AM), also referred to as ADM (Application Development and Maintenance), and Hosted Application Management (HAM).

MESSAGE FROM THE SPONSOR

Hexaware is a global leader and the fastest growing next-generation provider of IT, BPO and consulting services. We focus on transformation through our industry-leading delivery and execution model, built around the strategy— Automate Everything™, Cloudify Everything™, and Transform Customer Experiences™, enabling enterprises fast-track their digital journey.

Digital assurance is one of the strategic service lines that focuses on Quality Engineering and Assurance services. They help customers transform the testing function leveraging AI & ML technologies to move from Test Automation to Autonomous Software Testing. In the world of Agile & DevOps, testing activities would be independent from human intervention through self-learning and self-healing. It will also help organizations to deliver unprecedented speed with quality, while substantially reducing quality assurance (QA) efforts anywhere in the range of 40% to 70%, depending on the current QA maturity of the organization.

Curious to make the leap from automation to autonomous? Contact us at ***Marketing@hexaware.com***

IDC Custom Solutions

The content in this paper was adapted from existing IDC research published on www.idc.com.

IDC Research, Inc.
5 Speen Street
Framingham, MA 01701, USA
T 508.872.8200
F 508.935.4015
Twitter @IDC
idc-insights-community.com
www.idc.com