Watson Internet of Things

“Shift Left” for higher quality at greater speed

Reduce the risk of problems in software development by moving testing earlier in the delivery lifecycle

Watson IoT™

IBM

Contents

Many companies embracing modern software delivery processes that include both development and operations teams—known as "DevOps"—are learning to solve with speed the numerous challenges associated with delivering complex distributed systems. They're working across systems ranging from web to mobile, mainframe and third-party applications to meet the needs of today's empowered consumers, who have a myriad of choices in the marketplace. These choices are driving organizations to seek faster and more reliable ways of delivering high-quality products—products that consumers really want.

As an outgrowth of "agile" development practices, which address improvements in development practices, DevOps tightens handoffs between development and operations functions. But why is it that a recent survey of more than 600 IBM clients found that only about 30 percent of these organizations' budgets is used for innovation and research, while about 70 percent is used for maintenance and operations?[1] And why is it that these same organizations need four to six weeks to deliver even a minor software change?

It seems that agile development in its most basic form takes the software delivery lifecycle only so far. The time has come to embrace DevOps and evolve software development to the next level, where speed and quality are possible through the use of "lean" development methods, which aim to eliminate unnecessary, unclear, slow or otherwise wasteful processes.

An approach growing from agile development

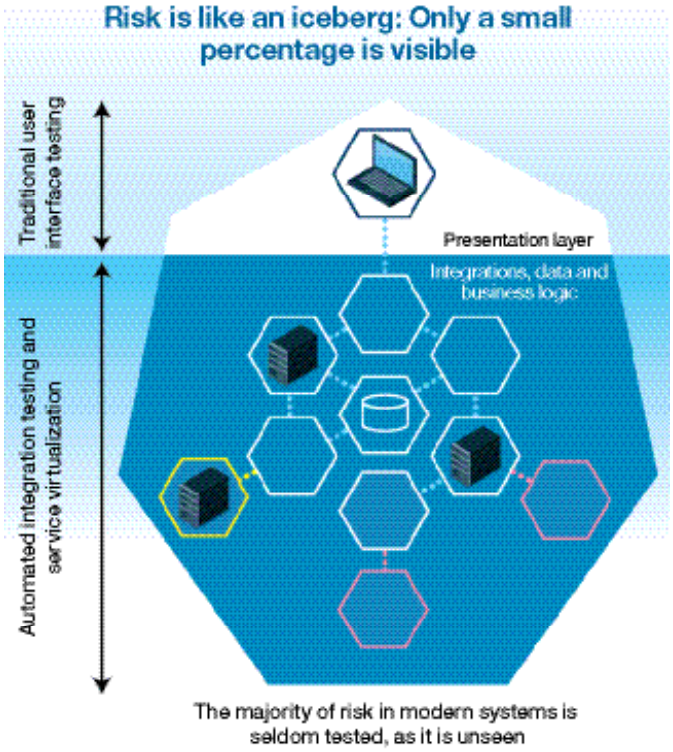
DevOps practices and technology make it possible to deliver smaller, more frequent batches of functionality and to progressively release a more feature-rich working product to the end user for feedback.[2] The expanded IBM DevOps approach to software development offers tighter internal and external feedback loops, which empower teams to deliver a better product sooner than ever before. Organizations that adopt enterprise-level DevOps automate, automate and then automate some more to accelerate the delivery lifecycle. The result can enhance predictability and speed, especially in testing and deployment activities.

Test the riskiest elements early and continuously. Providing early, iterative feedback on code quality directly to development teams helps ensure that fewer problems are found late in the lifecycle where they are more expensive to fix.

The IBM view is that software testing is the area in which DevOps-related improvements afford companies significant savings and a competitive edge. Testing is an area ripe for better processes—as the average spending on quality assurance as a percentage of the total IT budget is projected to rise from 18 percent in 2012 to 29 percent by 2017.[3] Costs are aggravated by the fact that that many organizations spend about a third of their testing budget on test environments,[3] and that test teams spend between 30 and 50 percent of their time on setting up test environments—instead of testing. In response, IBM focuses on optimizing testing and the related deployment operations so that more resources remain available for innovation. The new initiative is based on practices known as "Shift Left."

What is Shift Left?

The term Shift Left refers to a practice in software development in which teams focus on quality, work on problem prevention instead of detection and begin testing earlier than ever before. The goal is to increase quality, shorten long test cycles and reduce the possibility of unpleasant surprises at the end of the development cycle—or still worse, in production. In many organizations, automated testing of today's composite applications is being executed via the user interface, once the complete application has been developed and deployed. Waiting for all the pieces to become available before testing commences, however, often causes delays, adds risk to the project or results in discovery of late-stage defects—when they are more expensive to fix.



Shift Left practices help to avoid rework, delays and churn that can occur when major defects are discovered late in the testing cycle—after all integrations and product components are finally brought together as a composite application and made available for the team to test. It aims to avoid these issues by performing integration tests as the code is being delivered and deployed in a realistic production-like test lab. If any of the dependent application components are not available to test, virtual services can mimic the real components' behavior until they are ready.

In recent years, agile development teams have been able to minimize the risk associated with defect isolation by delivering working code in small batches, making it easier when problems occur to trace their cause to the offending code. But agile teams might not start with the most technically challenging and potentially disruptive parts of the code, which in complex products resides in the integrations. By contrast, teams that continuously test their code against integrated components can have better outcomes in reducing risk, as they are better able to find and fix potentially disruptive defects earlier in the lifecycle.

Simply put, this is what Shift Left means—shifting integration testing to the left of its usual position in the delivery pipeline so that it occurs as close as possible to the build process. By proactively testing high-risk integrations early and frequently, delivery teams can isolate the most disruptive, significant defects sooner for faster remediation. This smarter approach to continuous testing results in "quality at speed"—the concept that quality and speed are not only *not* mutually exclusive, but that with processes in place for concurrent testing and development, they can be achieved together for faster, more accurate results with fewer problems.

Avoiding unpleasant surprises late in the development lifecycle is an important benefit of shifting integration tests to the left, but there is another important benefit. Continuously delivering, building and testing in a tightly operationalized fashion allows the delivery team to receive feedback on code quality non-stop. This means that delivery teams can speed quality products to the market and validate new products or features with the customer. By shifting integration testing and feedback left—that is, closer to the beginning of the delivery pipeline—technical risk and the possibility of market failure are reduced. These are sound offensive and defensive tactics.

Shifting Left is a good defense and a good offense

For *defense*, Shift Left practices make it possible to run the product development and delivery processes more efficiently, with less waste, overhead and rework. These improvements are due to agile development processes and the practice of testing early and continuously to reduce the risk of catastrophic problems in production such as issues related to performance, security, data corruption or fraud. With a good defense in place, product teams have more resources available to help the organization succeed in the marketplace. As development teams can receive direct and immediate feedback on the quality of the code, good defensive practices also invest team members in product quality.

For *offense*, Shift Left practices help organizations aggressively deliver software products to the marketplace before their competition. Through early and continuous testing combined with ongoing and continuous monitoring, the delivery team receives feedback on software quality more quickly than ever before. Organizations can benefit from faster time to market and they can quickly react to how customers respond.

Better quality at higher speed is the result of early integration testing to improve quality earlier in the delivery process and reduce the risk of rework and delay. The result: products can reach the marketplce faster than those of the competition.

Is the notion of Shift Left new?

Yes and no. The idea of shifting testing and feedback loops to earlier points in the lifecycle is a recognized way to reduce risk and is an essential principle of the IBM DevOps approach to software development. Software development and project management experts have been talking about the need to attack the most difficult parts of software development first for a long time. Business leaders have insisted that delivery teams develop the highest priority functionality first for evaluation and feedback to ensure they are building the right product. What is new is that all the ingredients are now available to make shifting left possible:

- General **acknowledgement that agile development methods have value**, demonstrated by a recent survey that identified 52 percent of 231 teams studied as "agile."[1]
- **The onset of a new DevOps mentality**, which magnifies the problems caused by testing bottlenecks.
- **Software tools that efficiently configure and deploy software** to nearly any imaginable platform for testing, reducing time and resource needed for operations. These tools also make it possible to create production patterns for deployment so development teams can test in a production-like environment from the start.
- **Software tools that automate testing**, especially integration and regression testing, on newly delivered code. These tools can reduce the risk of rework and delays.
- **Service virtualization** that simulates the behavior of select components within an application. This capability enables earlier end-to-end testing of the application as a whole.
- **The proliferation of cloud environments** that are easy to spin up and destroy as needed, resulting in less overhead and more automation.

With these ingredients now available, software development teams have the methods and tools they need to code, continuously test and deploy more quickly for faster customer feedback and to streamline operations for the entire development lifecycle.

An illustration of Shift Left practices

The following scenario illustrates Shift Left practices in action:

An interdependent mobile team needs to deliver a late-breaking business feature that will mean higher client satisfaction and revenue. Throughout the project, the team has been testing integrations and code deliveries continuously, so the risk of a disruptive change is minimized.

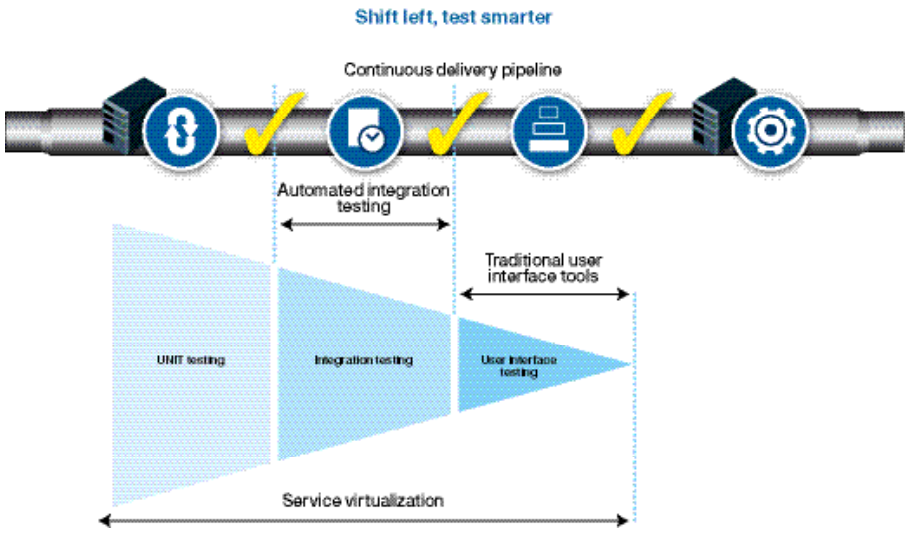
The team uses service virtualization software to stub out the dependent mainframe and third-party applications that are not available for testing. They also use application deployment automation software to quickly spin up test environments built using a combination of actual and virtualized components.

The status of testing results is visible through the use of a quality management software dashboard, which the highly integrated delivery team uses for reporting. Through the dashboard, the operations team learns about the new feature and, in anticipation of higher volumes, tweaks performance parameters to accommodate higher client activity.

Now whenever a new mobile build is made, the successful build process triggers the following automated activities:

1. The new mobile application build is installed in the development cloud-based test environment.
2. The stub for the mainframe service is started.
3. A stub for a third-party service is started.
4. An automated integration test suite is triggered for execution.
5. The test results are captured and feedback is made available for the delivery team on the quality dashboard.
6. Provided the integration tests pass, a snapshot of the applications in the development test environment is deployed to the system test environment.
7. The automated, mobile functional test suite is triggered and executed in the system test environment.

This is the Shift Left world—one of efficiency, effectiveness and increased opportunity for innovation.



An actual IBM example

An example of Shifting Left for faster feedback and delivery also occurs within the IBM® Global Technology Services® organization.

The IBM Service Delivery Application Development (SDAD) group is a team of 400 developers, testers and project managers located across 25 countries. It is responsible for design, development, requirements, quality and service for some 40 applications that manage and control development environments of 2,000 IBM strategic outsourcing clients.

Struggling to collaborate effectively, deploy applications quickly and keep applications up to date, the team needed a way get the latest technology into the client team's hands as quickly as possible—so clients could have the best possible analytics, service management, security, compliance and other solutions. What's more, the SDAD team needed to provide these capabilities while reducing costs.

As with many software development organizations, development and test teams in SDAD were separate, and all testing was conducted at the end of a cycle. The teams were able to change this model, however, so that all testing was automated as a set of test suites during development in an agile fashion.

The SDAD group also took advantage of the SoftLayer[4] cloud infrastructure and started to adopt a virtual desktop model by which workstations with the necessary tools are available in a cloud environment. The team also moved development, test and production-level environments to a cloud platform, and now uses deployment automation software to rapidly deploy new applications and updates.

> "We're going to start delivering value three months post-project start, versus nine months or 12 months under a more traditional type of approach and model."
>
> —IBM SDAD team director Scott Kinane[5]

Before these changes, software releases from development to test typically required 16 hours, while releases to production could take weeks to months. Releases to either environment can now take less than 30 minutes, a reduction of more than 90 percent. The team saves additional hours weekly now through automated testing, while improving software consistency and quality.[5]

Shifting Left is a mindset

Many organizations are slow to adopt early, continuous, automated testing and feedback due to the delivery team culture. Development teams are not accustomed to frequent, early deployments for customer feedback, either. When working on a complex project, teams are tempted to work on easy deliverables first. These are usually the visible parts of the software product such as the graphical user interface (GUI). Early wins bolster team morale and make management happy.

However, in a Shift Left world, software development teams and leadership honestly confront the thorniest and most potentially disruptive technology from the beginning as it is coded, and then automate the testing of these pieces (for example: invisible application program interfaces—APIs—and third-party services). Using this process, the team can focus on new features and early customer feedback. Good morale in the development organization then comes from delivering workable code on time that customers love.

For organizations to transform, they need to leverage new technical approaches. Delivery organizations, including management, must understand that testing is not the time for dropping a load of buggy code at the feet of testers. Instead, integration tests provide a channel for the entire team to take shared ownership in holistic quality. Testing must be tightly coupled with development—and it must happen nearly at the same time as code is delivered. In a Shift Left world, the lines between testers and developers blur, test and production environments are very similar, and leadership encourages continuous collaboration, discovery, learning and optimization.

Summary

A Shift Left approach to software development shortens the testing cycle providing the speed and quality that companies need to be competitive, whether they are delivering a single mobile application or a complex multiplatform enterprise software suite. Shift Left practices involve a transformation for the delivery team. Now the entire delivery team contributes to quality, not just the quality assurance or testing team. Lines between the developer and tester fade as they work together to automate testing and partner with operations to make deployments fast, reliable and repeatable. As teams become more efficient and effective, more resources and time often become available to spend on innovation.

The ingredients to make Shift Left are real and available: automated testing and deployment tools, service virtualization technology, agile practices, and cloud environments. The main obstacle that remains is the mindset of delivery organizations. These groups must transform themselves. The role of the quality assurance tester must include advanced automation and integration testing skills, and developers must see themselves as responsible for quality by testing integrations earlier and more often.

In a way, the adoption of Shift Left processes is a familiar story: people, process and tools. Advances in tooling and acceptance of agile methods are now in place. What remains is to overcome organizational and cultural challenges.

For more information

To learn more about implementing Shift Left capabilities with IBM DevOps solutions, please contact your IBM representative or IBM Business Partner, or visit: **ibm.com**/cloud/devops

Additionally, IBM Global Financing provides numerous payment options to help you acquire the technology you need to grow your business. We provide full lifecycle management of IT products and services, from acquisition to disposition. For more information, visit: **ibm.com**/financing

About the authors

Laurel Dickson-Bull – Product Manager, UrbanCode Deploy
Laurel is a Certified Project Manager who has worked in software development, including release management, globalization, and software support and client management for more than 20 years.

Marianne Hollier – DevOps Solution Specialist
Marianne is an IBM and Open Group Master Certified IT Specialist with strong, practical expertise in measurably improving the software development lifecycle and driving the necessary cultural changes to make it work.

Al Wagner – Testing Evangelist
Al is a Testing Evangelist with IBM, driving thought leadership, strategic initiatives and tangible solutions with a specific focus on quality management, test automation and service virtualization.

Footnotes

1. Ashok Reddy, "Eight critical DevOps practices: innovate, deliver, repeat," *Forrester*, October 22, 2014. http://devops.com/features/eight-critical-devops-practices-innovate-deliver-repeat/

2. Eric Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses* (New York: Crown Business, 2011).

3. "World Quality Report 2014 Shows Majority of Testing Budget Now Dedicated to New Development Projects as CIOs Prioritize Transformation over IT Maintenance," *Capgemini*, October 7, 2014. http://finance.yahoo.com/news/world-quality-report-2014-shows-063000372.html

4. SoftLayer Technologies was acquired by IBM in July of 2013.

5. "DevOps Internal Success: IBM Global Technical Services," *IBM Corp.*, October 3, 2014. https://www.youtube.com/watch?v=HuJiaEATkmw

Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
December 2017

RAW14380-USEN-01

Please Recycle

Watson IoT™