Infosys®
Navigate your next

CASE STUDY

DEVOPS TOOLING FOR DEVOPS SUCCESS

DevOps is more about people than tools, something we explored in the first two articles of this six-part case study, outlining one global bank's DevOps transformation journey.

But a software development process cannot work effectively without the right tools. IT giants like Google, Facebook and Salesforce have invested to establish their DevOps tooling framework so that they can boost productivity while continuing to serve their customers at the highest level. In this third article, we continue on the bank's transformation journey. We introduce the DevOps tooling framework that allows development and operations to work well together so that they can enable process automation, reduce deployment time and continuously integrate and deploy software when and where it's needed.

Infosys® | Knowledge Institute

Introduction

In the previous articles in this series we have seen that DevOps is more about people than tools.

But a software development process cannot work effectively without the right tools. These tools enable process automation, reduce deployment time and continuously integrate and deploy software when and where it's needed.

When all these tools work in concert, and with the right mindset, DevOps enables large firms to innovate like a startup and bring new applications to market quickly. IT giants like Google, Facebook and Salesforce have invested to establish their DevOps tooling framework so that they can boost productivity while continuing to serve their customers at the highest level.[1]

The second article in this series introduced the new operating model that a leading global bank used to become more agile. Instead of developers throwing code over the wall to operations teams, a combined team of development and operations, along with testers and other support functions, worked together on a product backlog in a series of sprints.

Over a period of 18 months, the deployment time for new code reduced from two weeks to two days for several teams. This dramatic success meant that the bank could overcome the competition of smaller, tech-driven, agile financial services disrupters.

In this article, we continue to follow this global bank's transformation journey and introduce the DevOps tooling framework that allows development and operations to work well together.

How is code integrated, validated and delivered in DevOps?

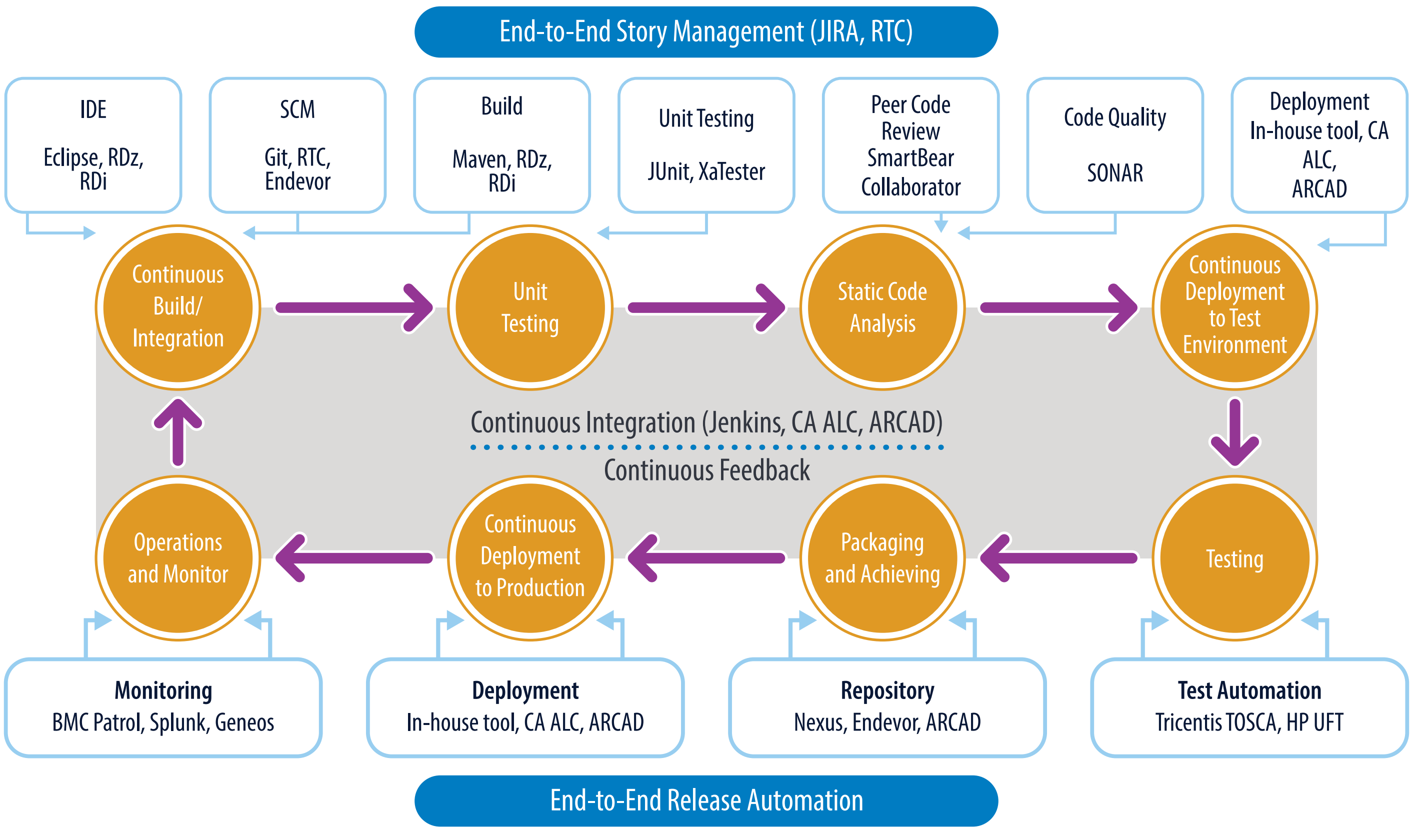
When DevOps works effectively, all code is delivered in a continuous fashion, with no bottlenecks in the process.[2]

Code is frequently checked in so that an orchestrator (like Jenkins) can build the solution in a continuous fashion; code quality is also checked with every build with an enterprise-wide code analysis tool like SONAR.

After the quality check, code is prepared for build before peer reviews are conducted. The tooling framework also ensures that testing is performed early in the process. Once these steps are complete, code is packaged and deployed automatically into higher environments of testing or production.

This DevOps framework makes use of end-to-end story management (in JIRA), continuous integration (through technologies such as Jenkins, CA ALC and ARCAD), continuous feedback and end-to-end release automation (see Figure 1).

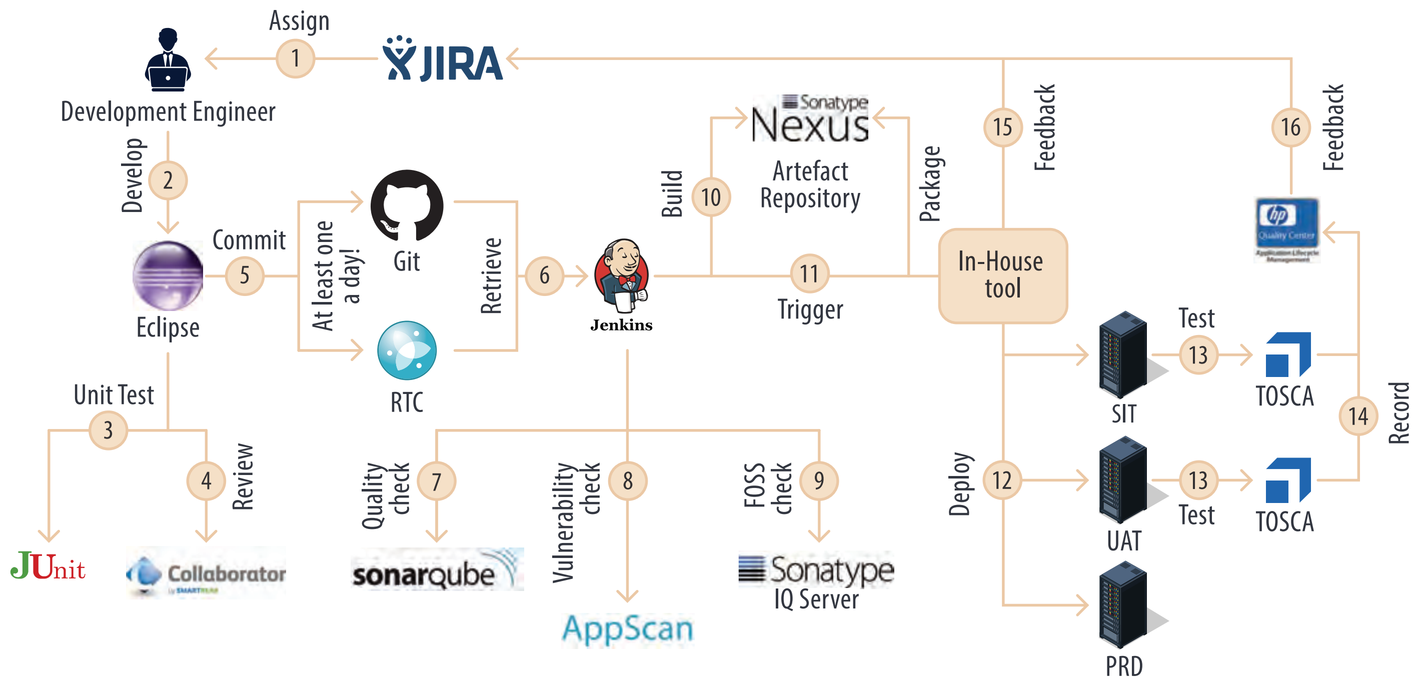
Figure 1. Agile teams demonstrate continuous integration, validation (testing) and delivery of software in the DevOps engineering framework



This life cycle isn't rigid, however, and can be set up in numerous ways using a wide range of off-the-shelf technologies.

One such life cycle - from the point of view of a developer working in an agile team on distributed systems - is shown in Figure 2.

Figure 2. Off-the-shelf technologies are integrated in the DevOps life cycle of continuous integration, testing and delivery of code



DevOps as a practice

Software integration is a process of merging two or more diverse software programs so that the data and functionality flow between them smoothly. It is intended to make an existing application better and more robust.[3] Getting that application or product to market is the next stage in the process. These two steps are typically difficult in traditional software environments, with many milestones along the way. Software quality must be manually reviewed before rigorous checks (quality gates) are carried out. Additionally, this software must meet the nonfunctional requirements of a huge enterprise, including quality attributes such as testability, maintainability and scalability. These checks lead to delays and sometimes obscure feedback loops as system experts provide their input. Additionally, the whole process is subject to human error.

With DevOps approach, these review processes are automated. DevOps tools improve quality, increase speed and cut costs.

In the DevOps transformation, five principles were followed so that software integration, testing and delivery would be able to support more and more processes as the IT landscape grew in size.

1. No manual deployments	Before the transformation, if a bug or other quick fix was needed, project teams deployed their changes into the production environment manually. In the new way of working, continuous deployment and delivery tools like CA ALC and ARCAD (for i-series systems) are combined in the software pipeline. This ensures that no manual deployments are allowed, reducing the risk of human error. Access privileges are reviewed and revoked for most teams to ensure that the system is completely foolproof.
2. Reduce technical debt	High levels of technical debt are an unwelcome reality at large enterprises. The cost of performing additional work in the future increases as teams struggle to deliver code and resort to ad hoc improvements in code quality and testing. As the bank's transformation progressed, many issues were found in the technology estate. These issues were addressed, and a process was set up to ensure no new software issues were released into the estate. To aid this process, DevOps tools like SONAR (code quality) and JUnit (unit testing) created reports that were submitted to a change advisory board. Change approvals were then tied to a reduction in technical debt for each application.
3. Minimum operating standards	A data-driven, minimum operating standard (MOS) was determined for each software release before it was deployed into production. For example, the number of issues identified by a code quality tool like SONAR or the number of build failures uncovered by Nexus was compared with the MOS. If they were inferior in any way, the code was not deployed. This method allowed agile teams to maintain a consistent level of quality before release to the end customer.
4. Manage by fact	The bank initially had few measures to score project teams, and performance evaluation was qualitative. MOS and technical debt measures introduced objective evaluation of team performance. Talent management and rewards and recognition were then tied to these KPIs. DevOps also enabled other sources of data to evaluate teams, including adhering to best practices (frequent check-ins), adopting DevOps principles (expanding unit test coverage for legacy code) and demonstrating continuous improvement (enhancing the pipeline through continuous automation).
5. Eliminate vulnerabilities	Manual reviews were the norm before the transformation, and a security expert reviewed nonfunctional requirements such as resilience and security. Adoption of tools like IBM AppScan (to check for security vulnerabilities) and Sonatype Nexus IQ Server (to check for free and open source software usage) eliminated these manual reviews. These tools were integrated in the continuous integration (CI) pipeline, improving delivery pace and ensuring code quality. As a further gate to ensure code wasn't at risk from attack, the whole technology estate was scanned for new vulnerabilities by modifying a central configuration in the cloud.

TOSCA for test automation

A test automation team was set up to reduce as much of the testing effort as possible using a minimum set of automated scripts.

With DevOps, automated testing tools are capable of executing tests, reporting outcomes and comparing results with earlier test runs. Tests can be run at any time of day, and repeatedly, so all the software infrastructure is engaged in the process.

Test cycles run at a fast pace, avoid human error and prevent phantom defects.

The reusable nature of DevOps ensures scripts are used over and over again across applications.

Delays associated with manual handoffs were reduced as a result of this accelerated software engineering life cycle.

Prominent tools like HP UFT, Selenium and TOSCA were selected to operate within the continuous testing platform. TOSCA had the advantage of automating tests across all layers of the application — including the application programming interface (API), web, mobile, business intelligence and data warehouse stack.

TOSCA also integrates well with existing scripts (automated through other tools like Selenium or HP UFT), is easily configurable within the DevOps CI and CD pipeline, and provides a central repository where all scripts can be stored — avoiding rework.

TOSCA was a huge success: The team achieved near 100% automation of regression test suites, with a 70% reduction in regression testing effort.

This is a short introduction to the DevOps engineering framework implemented, the strategy used, and why success was achieved early on in the global bank's transformation.

The next article (Article 4) describes how operations comes into the picture. Successful operations within the DevOps methodology ensure that all individuals within the agile team have a seat at the table. With operations working closely with developers, code is successfully deployed with fewer service issues. This leads to the increased deployment frequency and bug-free code discussed in earlier articles, with faster time to market for applications.

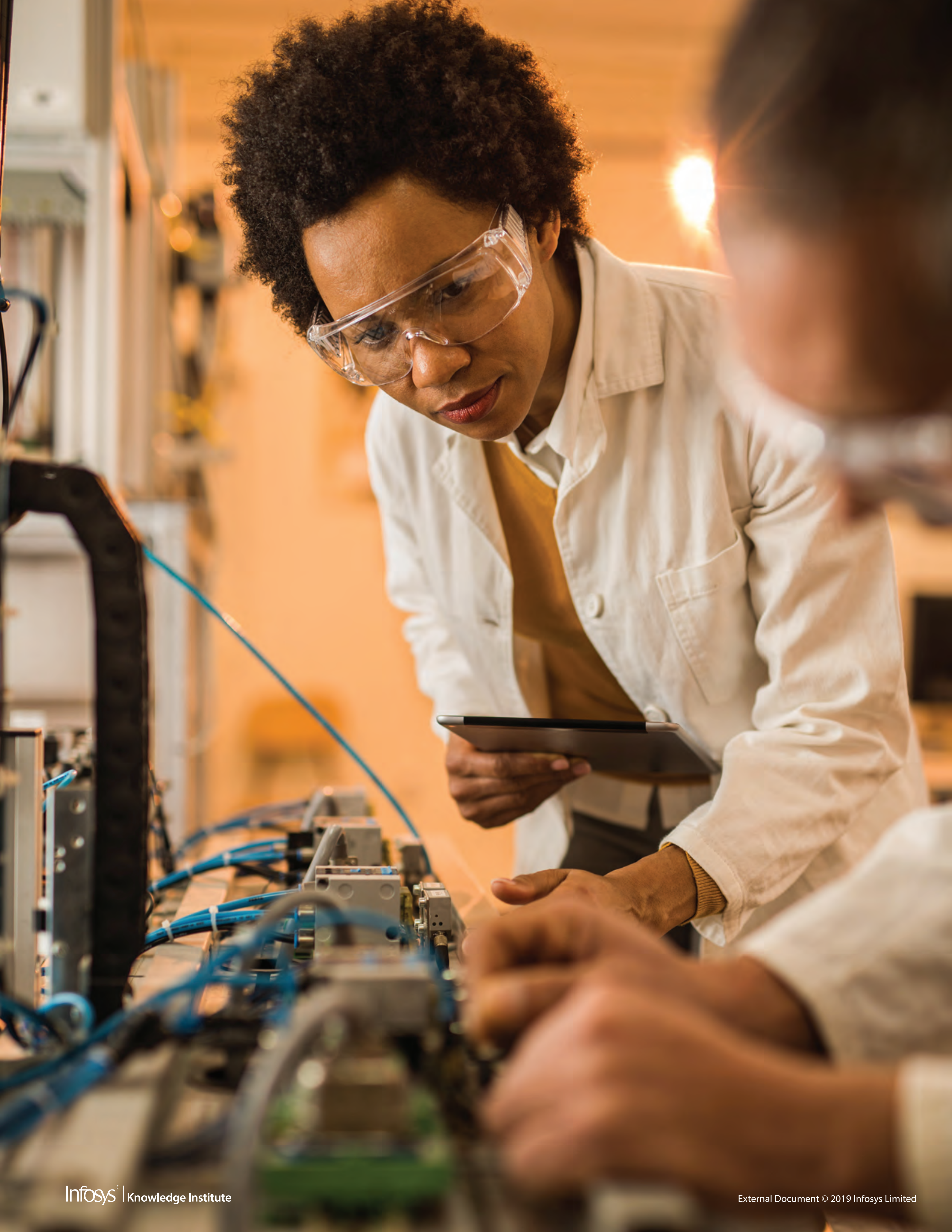
With DevOps tooling in place, improved customer experience awaits.

References

1 https://www.cognixia.com/blog/impact-devops-bottom-line

2 DevOps: The IBM approach — White Paper

3 https://www.pcmag.com/encyclopedia/term/65245/software-integration



Authors

Alok Uniyal

Vice President and Head of IT Process Consulting
alok_uniyal@infosys.com

Harry Keir Hughes

Senior Consultant - Infosys Knowledge Institute
harrykeir.hughes@infosys.com

About Infosys Knowledge Institute

The Infosys Knowledge Institute helps industry leaders develop a deeper understanding of business and technology trends through compelling thought leadership. Our researchers and subject matter experts provide a fact base that aids decision making on critical business and technology issues.
To view our research, visit Infosys Knowledge Institute at infosys.com/IKI

For more information, contact askus@infosys.com

Infosys®
Navigate your next