bcg.perspectives
by THE BOSTON CONSULTING GROUP

LEANER, FASTER, AND BETTER WITH DEVOPS

By Hanno Ketterer and Christian N. Schmid

Agile methodologies have given companies just what they need in the digital era: a collaborative and flexible way to develop software. (See "The End of Two-Speed IT," BCG article, August 2016.) Cross-functional teams, iterative development, and continual testing and feedback are powerful practices that have enabled companies to improve their development process. But businesses can realize even greater transformative change—and success—by implementing a second set of practices known as DevOps.

DevOps calls for some staff from IT operations, which traditionally works apart from developers, to be brought onto cross-functional agile teams. In addition, DevOps puts heavy emphasis on automation. This set of practices is not an alternative to agile but a complement—one that takes agile a step further and applies it to the rest of the software life cycle: deployment, release, operation, and monitoring.

Indeed, the results are eye opening. By implementing DevOps, the financial services company Nationwide achieved a 70% reduction in system downtime and a 50% improvement in code quality. In our experience, DevOps has helped companies tackle security issues in half the time they traditionally required, release new code on demand instead of on a fixed schedule, and reduce IT costs significantly. Indeed, companies that have combined DevOps with a standardized and fully virtualized infrastructure have seen IT costs drop by as much as 25%. Such results translate into a crucial benefit: sustainable competitive advantage.

DevOps Helps Create Value

For all the benefits that agile brings, it doesn't change the basic relationship between development teams and IT operations. As a result, some companies are finding that delays persist at critical handovers because IT operations is still working in time-tested but largely manual ways. ING Bank's CIO, Ron van Kemenade, put it this way: "We found that working in an agile way solely in development didn't really make much of a difference. IT operations needed to be included as well, since that's basically where the buck stops before you go into production." (See "Building a Cutting-Edge Banking IT Function: An Interview with Ron van Kemenade, the CIO of ING Bank," BCG article, December 2015.) By bringing some staff from IT operations onto cross-functional agile teams and stressing automation, DevOps helps companies address this issue while delivering other key benefits. Indeed, DevOps creates tremendous value in three important ways. (See Exhibit 1.)

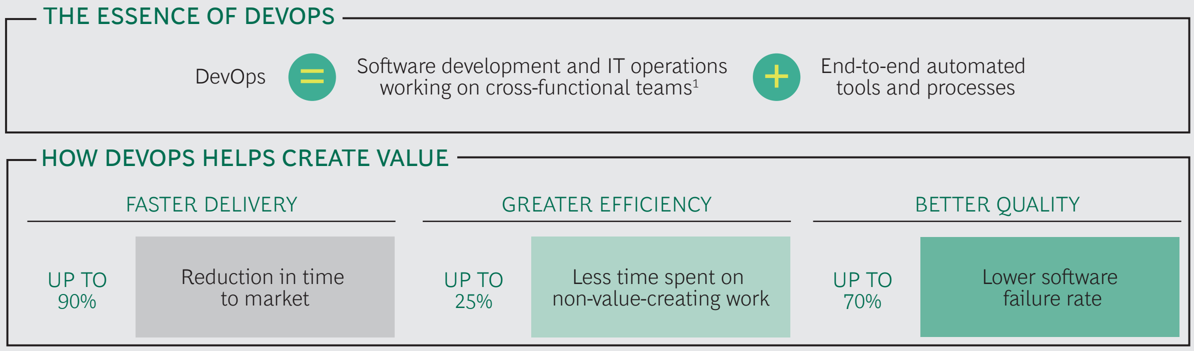
Faster Delivery of Features and Changes. Developers—even agile ones—typically call IT operations when they are ready to send software to testing. If the software is brand new, operations needs to configure the test environment. If the software is a new release of a current product and a test environment already exists, interfaces and side applications still need to be configured and added. These tasks, like many in operations, are labor-intensive, and a developer can deploy code only so often under such a system. Therefore, developers tend to work with large chunks of code. But this approach comes at a price: an individual change isn't released until all the revisions packed with it are working.

DevOps helps companies accelerate the time-consuming tasks in IT operations. For example, automating testing provides developers with faster feedback, and automating integration incorporates developers' changes more quickly into the code base. Such changes encourage developers to work with smaller chunks of code—even just a few lines. Then, instead of eventually releasing a big kitchen-sink application, companies can release a series of small, incremental updates, getting new features into the field fast.

Being able to release on demand is particularly beneficial when it comes to security. In "zero day" exploits, hackers try to take advantage of a vulnerability before it can be fixed. ("Zero day" refers to the fact that the software's creator effectively has zero days to solve the problem.) If creating patches follows the traditional development, testing, and release processes, the hackers' window of opportunity stays open. DevOps helps companies release fixes as needed and seal breaches fast—often within hours rather than days.

Greater Efficiency. DevOps helps IT professionals devote more time to value-creating work. For example, when companies automate testing and integration, developers no longer spend big chunks of their day waiting for machines to be configured or code to be integrated. They can do both by clicking a button on a self-service portal. After a large European bank implemented DevOps, it reported efficiency improvements of up to 25% in developing updates for its online banking application. The IT operations staff is also freed up for more challenging work that adds more value, which will prove rewarding for employee and employer alike.

EXHIBIT 1 | DevOps Helps Improve Agility, Efficiency, and Quality



THE ESSENCE OF DEVOPS

DevOps = Software development and IT operations working on cross-functional teams[1] + End-to-end automated tools and processes

HOW DEVOPS HELPS CREATE VALUE

FASTER DELIVERY: UP TO 90% Reduction in time to market

GREATER EFFICIENCY: UP TO 25% Less time spent on non-value-creating work

BETTER QUALITY: UP TO 70% Lower software failure rate

Sources: *2016 State of DevOps Report,* Puppet and DevOps Research and Assessment, 2016; *The Total Economic Impact of the Microsoft DevOps Solution,* Forrester Research, 2016; BCG analysis.

[1] Cross-functional teams typically include representatives from the business side of the company.

Better Quality Code and Faster Recovery from Failures. After software is released, developers typically move on to their next project. Therefore, they don't have an incentive to anticipate or prevent longer-term problems. Future fixes will be operations' headache.

DevOps keeps developers involved and on the hook throughout the life cycle of a feature or an application, resulting in better-quality code. Fewer fixes are required because developers look for and eliminate potential problems as they write code. When failures do occur, bugs are more easily traced to their source because developers are working with smaller chunks of code. Meanwhile, human error is reduced by all the automation that DevOps brings to the software life cycle. As a result, companies can deliver fixes fast.

Of course, having the right talent is critical if organizations are going to develop better-quality code. The top IT professionals embrace new approaches that deliver better results. By implementing DevOps, companies can not only attract but also retain top-tier talent. There is no better way for businesses to show that IT is on the cutting edge than by using innovative tools and methodologies.

Making DevOps Work

Like agile, DevOps has no textbook implementation model. But successful companies follow five best practices.

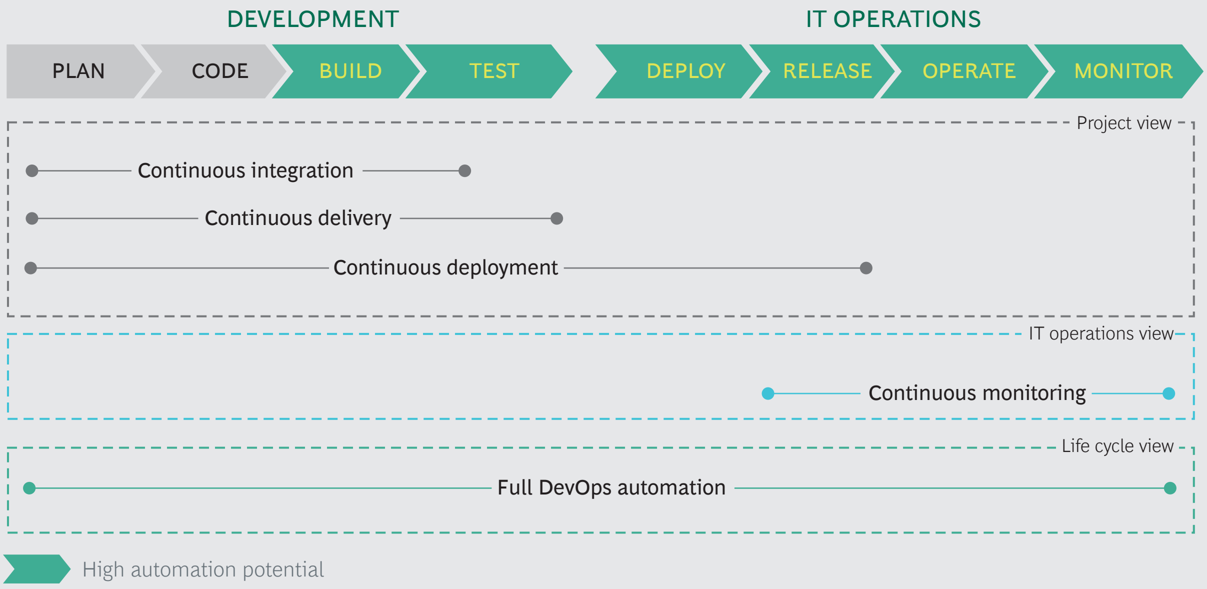
Automate the software life cycle in stages. One can't overstate the importance of automation to an effective implementation of DevOps. Automation is crucial to developing quality software faster and more efficiently. But companies should see automating the software life cycle as a journey—one that should be carefully orchestrated.

The full software life cycle encompasses many steps; which steps should be automated and in what sequence needs to be considered. For instance, automating integration testing won't eliminate bottlenecks if acceptance testing is still performed manually.

The key is to think about orchestration from the very start of a DevOps implementation. Analyze the current development process and challenge frequently used routines. If there are tests that code always passes, maybe they should be discarded rather than automated.

Companies generally implement automation in four stages. (See Exhibit 2.)

EXHIBIT 2 | DevOps Stresses Automating the Software Life Cycle



Source: BCG analysis.

- **Continuous Integration.** This first step calls for software developers to merge their code with the larger code base frequently, often many times a day. Each time a chunk of code is integrated, the code base is automatically tested.

- **Continuous Delivery.** This is usually the next logical step. It calls for developers to work on code in cycles that are short enough for a company to reliably release an incremental update at any time. Continuous delivery doesn't mean that every change is released; sometimes changes are held back for business reasons or regulatory requirements. But there is always an update that could be released.

- **Continuous Deployment.** This stage calls for automatically deploying every change to production as long as the code has passed all testing. Not every company will want to implement this. Continuous deployment works best for businesses that need to move particularly fast in getting out new features, applications, or security measures. Online retailers and media companies are two prime examples. But companies in other industries, such as the automated investment services firm Wealthfront, have also become big proponents and beneficiaries of continuous deployment.

- **Continuous Monitoring.** Most companies will want to implement this stage, which calls for constantly assessing the behavior of released applications. It lets companies home in on—and even predict—issues as quickly as possible so that they can be prevented or corrected before they become big problems.

Standardize tools, processes, and practices. Another best practice for implementing DevOps is standardizing tools, processes, and practices across teams. This is something that is often absent within organizations. Standardized approaches, as opposed to ad hoc solutions, reduce errors and improve knowledge sharing. They are also a prerequisite for implementing automation, which lies at the heart of DevOps. For instance, automating server configuration through a self-service portal requires a defined set of configurations, instead of an endless array of possible permutations.

Standardization also reduces the overall number of tools companies use, increasing efficiency and potentially reducing costs. Traditionally, tools haven't been coordinated across IT. Developer teams and operations teams all select and manage their own tools despite any overlap. By contrast, DevOps requires an integrated, centrally managed tool chain. Companies will have to make changes, but the effort is easily outweighed by the benefits.

As part of their drive toward standardization, companies should challenge tradition and ask if existing tools, processes, and practices should be updated or improved. For example, we find that DevOps works particularly well in Infrastructure as a Service and Platform as a Service environments, because the necessary IT resources are available on demand. Companies that adopt these services could eliminate the need for potentially lengthy infrastructure-provisioning processes. Indeed, DevOps helps companies get the most out of cloud-based services (especially when those services are accessed through a public cloud). The cloud's ability to rapidly deploy virtual machines—and hence a company's software—is of value only if new features and changes are developed and released quickly.

On the application side, decoupled architectures—made possible by the use of standard application programming interfaces (APIs) and microservices—enable developers to build modular components. Perhaps the most important benefit to this approach is that if a faulty piece of code is deployed, only that piece—and not the whole system—fails.

Rethink the team structure. DevOps, by definition, brings the development and operations sides of IT together on cross-functional teams that have end-to-end responsibility for the full software life cycle. (See Exhibit 3.) But the spirit of DevOps is about collaboration and communication

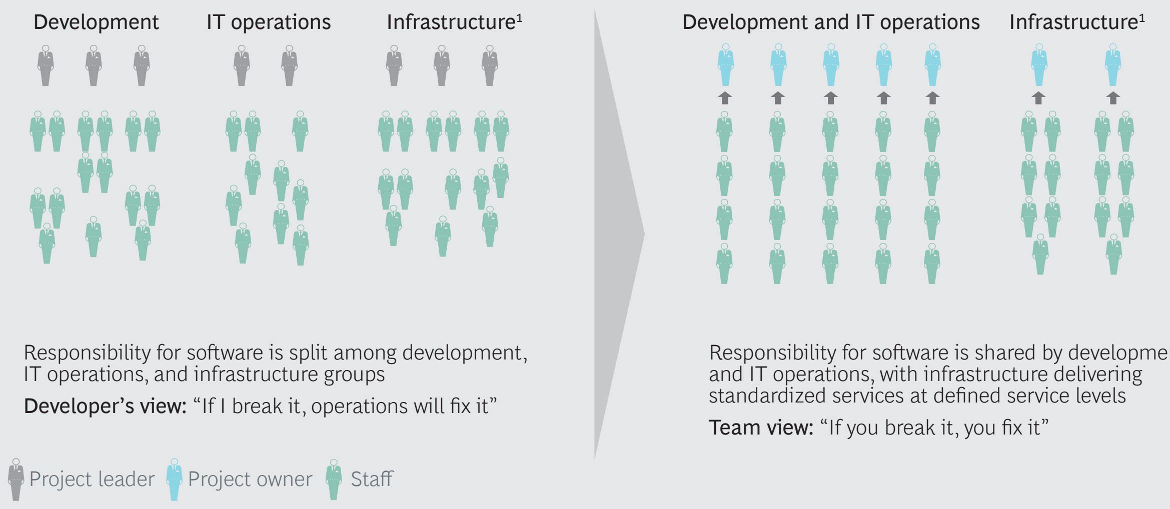
Exhibit 3 | DevOps Stresses End-to-End Responsibility

BEFORE DEVOPS

PROJECT-BASED TEAMS WORK IN SILOS AND AT VARIOUS LOCATIONS

AFTER DEVOPS

LIFE-CYCLE-BASED CROSS-FUNCTIONAL TEAMS WORK TOGETHER AT ONE LOCATION



Responsibility for software is split among development, IT operations, and infrastructure groups

Developer's view: "If I break it, operations will fix it"

Responsibility for software is shared by development and IT operations, with infrastructure delivering standardized services at defined service levels

Team view: "If you break it, you fix it"

Source: BCG analysis.

[1] Managing infrastructure is never the responsibility of development, IT operations, or DevOps teams.

among everyone with a stake in the software. So a team's roster doesn't have to end with IT; it can include other stakeholders, such as security experts. Indeed, some companies, such as Capital One, implement a variant called DevOpsSec.

As companies build their teams, they should keep several points in mind. Implementing DevOps changes the skills that employees will need. Operations personnel, for example, will no longer configure environments manually; instead they will need to write scripts that automate configuration. Determining the right team size is also critical. Although it may be tempting to include as many stakeholders as possible, large teams can make communication unwieldy. What's the best size? Generally, a team should have seven to nine members. Indeed, at Amazon, the rule of thumb is the "two-pizza team"—one that can be fed with two pizzas. Finally, depending on a company's needs, some IT professionals, such as application architects and database engineers, may be better positioned in centers of excellence than on DevOps teams. (It's important to note that DevOps teams never manage the infrastructure. That role falls to either an in-house infrastructure group or an outside provider, such as Amazon Web Services.)

Facilitate the needed cultural shift. Implementing DevOps requires cultural change, so top management must be strongly committed to the effort. Moving to agile wasn't seamless for many companies. Many of the coordinating tasks that mid-level managers had traditionally handled ceased to exist, and the managers moved onto the teams, acting more as coaches than supervisors. Pushback wasn't unusual, and it should be expected as employees from IT operations join cross-functional teams. Education, enablement, and ongoing experience with DevOps can smooth the transition, but so, too, can support from the top.

A cultural shift will be needed at the team level as well. With code changes integrated and released much more frequently, teams need to rethink, and even overturn, longstanding practices—formal and informal ones alike. Gary Gruver, former Hewlett-Packard director of engineering for LaserJet enterprise firmware, once described the dos and don'ts that the company established after it implemented continuous integration: “We had certain rules. [If] you commit code and then you go home before it goes green, that’s a lab felony.”

Diplomacy may also be required. Some companies have found that bringing IT operations staff onto cross-functional teams can create certain tensions, as the employees are likely to have preexisting, and even unflattering, views of each other. One way to overcome such friction is to emphasize—and specifically, show—the value of working in this manner. Pilots that are staffed with those most open to DevOps can create champions and achieve results that promote collaboration and defuse tensions. However, finding a pilot that produces quick and clear results can be tricky. Developing a product or service that doesn’t depend on a fixed release schedule—and that can be rolled out separately from other applications—is ideal.

Finally, implementing DevOps means that teams need to learn to share knowledge. Regularly having internal “DevOps days” can provide a venue for team members to spread the word about best practices while learning more about one another’s roles. New hires, meanwhile, should be trained from the outset in DevOps. This will not only make their transition to a team smoother but also send a message: this is a company that is embracing new ways, and new hires can be part of the change from day one.

Develop new KPIs. As DevOps helps companies reengineer the way teams work, management should develop new KPIs to gauge how teams are performing and how processes are working. These could measure, for example, the number of code commitments occurring each day and the time between commitment and deployment to production. To home in on red flags, companies could track metrics such as the number of code commitments made each weekend or how often someone is called for help outside working hours. A solid set of KPIs helps identify not only bottlenecks but also processes that can be improved. And this highlights a crucial point: DevOps can and should evolve over time.

FOR TODAY’S COMPANIES, the pressure to reduce the time to market, improve efficiency, and boost quality is constantly increasing. Each day seems to bring new urgency. Agile methodologies go a long way toward helping companies develop software in the digital era; DevOps practices help them transform the software life cycle. With careful attention and planning—and management’s commitment to working in the new ways that today’s business landscape requires—DevOps can help companies deliver great software more efficiently and more effectively than ever before.

About the Authors

Hanno Ketterer is a senior partner and managing director in the Amsterdam office of The Boston Consulting Group and the global leader of the insurance sector in the firm’s Technology Advantage practice. For more than 20 years, Ketterer has led large-scale technology transformations and postmerger integrations in the banking and insurance industries. You may contact him by e-mail at ketterer.hanno@bcg.com.

Christian N. Schmid is a principal in BCG’s Munich office. He is active in the firm’s Technology Advantage practice, where he is a global leader of the DevOps topic, and he is the global segment leader of the Simplify IT topic in the Financial Institutions practice. For the past ten years, Schmid has driven large-scale transformations—including digital and agile transformations, simplification and restructuring programs, and postmerger integrations—for numerous companies. You may contact him by e-mail at schmid.c@bcg.com.