DevOps At Scale: Chapter 3

# DevOps Methodology: Aligning your Organization

PLUTORA

Previously, we've covered what DevOps is and how it can benefit your organization. In this article, we'll cover the changes you'll need to make to achieve DevOps.

DevOps can provide several benefits, such as faster delivery of new features, increased stability, and better collaboration. It improves the entire service life cycle, but it requires a significant structural shift to implement. To do this, you'll need to look at three key areas:

- Cultural change
- DevOps processes
- A shift in technology

It takes several roles and responsibilities to incorporate DevOps effectively. Some of these roles are unique to DevOps. Others are part of traditional software development, but implementing DevOps will mean adjusting responsibilities for them. We'll define each role and consider how it contributes to DevOps success.

## DevOps and Cultural Change

The most important step in realigning as a DevOps organization is getting everyone to share the same goals. This step is frequently the most challenging.

A fundamental tenet of DevOps is increased communication and collaboration between teams. This most commonly applies to development and operations teams, but it can also apply to test engineers, product managers, executives, and business teams. This change can represent a significant cultural shift within the organization, particularly if teams are used to working mostly in isolation, with clear divisions of responsibility.

How can you help bring about this level of change? Trust, honesty, and responsibility are key. Changes must begin with top-level management. These managers will need to convince the rest of the organization that they all share a common goal: better software, delivered more quickly. Understanding of a shared goal will help align different teams rather than pitting them against one another. Ideally, this will result in increased empathy, which will then facilitate trust between teams—especially as they see functions they once controlled being adopted by other teams as well.

Workforce transition planning is also essential. It will ensure adequate training for employees who are taking on new functions. This type of planning will help ease employee concerns and build trust in the overall transition to DevOps.

## DevOps Processes

Once your organization has successfully implemented this cultural change, the next step is to update your processes.

DevOps processes are based on the idea of continuous improvement. Teams often do this by setting up an automated deployment pipeline, which enables quick, reliable delivery of new features to users. A typical pipeline contains three stages: continuous integration, continuous testing, and continuous deployment. This type of pipeline allows your organization to integrate new features into the code base, test them, and roll them out on a continuous basis. In most cases, large segments of the process will rely on automation to increase efficiency and reliability.

### Continuous Testing

Integrating testing throughout the development process is critical for catching issues early. This becomes especially important as your organization increases its level of automation. Testing within the pipeline ensures that new features work as expected before they are released, from check-in through to delivery.

Monitoring and testing everything, however, may not be possible and is probably not necessary. So, how do you decide what is important? Focus on these categories:

- Development cycles
- Deployments
- Vulnerabilities
- Server health
- Application performance

In particular, vulnerabilities point to the need for security testing early in the software development process. To minimize the risk of a security failure, testing for vulnerabilities should become everyone's responsibility, not limited to a single team at the end of the development process. This mindset is often referred to as DevSecOps.

If your organization is taking care of these categories, it will be easy to notice quickly if anything isn't operating as expected.

### Governance and Compliance

Monitoring software development is important when it comes to satisfying governance requirements and compliance regulations. Governance involves the established company and IT standards that your services are required to follow. Compliance means that the software development life cycle adheres to company standards including regulatory requirements for data storage, business operations, and other practices. Each organization must have a strategy for following governance and compliance. Defining workflows that meet governance standards and tools that monitor releases to create traceable audits provides organizations with the ability to review compliance rates and identify root causes of non-compliance in order to improve processes.

You may find that fulfilling governance and compliance requirements slows down your releases. Having multiple pipelines can also lead

to a misalignment between process governance criteria and configurations. Both of these issues are related to issues with policy implementation, execution, resource coordination, and end-to-end oversight.

To make sure your organization is meeting all requirements, consider using a management tool. Several of these tools exist for managing compliance and risk, and in some cases they'll allow you to manage performance management as well. Plutora's platform is a leading example of this, allowing for control of quality and compliance across all projects.

## Value Stream Mapping

A value stream is the flow of a product feature from check-in to delivery that creates business value. Value stream mapping allows you to visualize the entire process in an application's development and delivery. This bird's-eye view allows you to identify constraints in the process that you or your team can address and improve. Or you can calculate which activities are generating value and optimize them.

Value stream maps also provide insight into performance baselines and other metrics through measured time-to-value in different stages of the process. This post summarizes the benefits of value stream mapping:

The goal of value stream mapping is to provide a tool to help migrate the process from what is actually happening to what should be happening.

Traditionally, people mapped value streams by hand. While beneficial, this method was prone to errors. Manual maps depend on estimated timing between steps, so they may be unable to effectively capture a continuous, fluctuating process. Modern tools allow for more accurate tracking and reporting of process data, which in turn provides a better visualization of your organization's flow of value.

## Value Stream Management

Software delivery performance monitoring lets your organization track performance outcomes. These include:

- Deployment frequency
- Mean time to recover
- Lead time for changes
- Change failure rate

While these outcomes are informative for software delivery throughput and stability, just viewing them doesn't provide insight into the processes that could improve an organization's metrics. On the other hand, value stream mapping captures the processes involved and provides insight into how to improve these performance outcomes. Value stream management extends this idea by including more support for other development life cycle management functions. For example, you can coordinate and configure activities within multiple pipelines to run with fewer delays from inefficient sequencing. It's also possible to oversee all active environments because you can simplify and centralize all configurations, code changes,

and scheduling. The result is a powerful tool that allows for continuous improvement across your organization's entire platform through small, localized improvement experiments.

As a market leader, Plutora is the most complete value stream management platform and the only platform that provides live metrics and analytics portfolio-wide. It was built to support the complexity of enterprise software delivery, including:

- Multiple release processes
- Governance and compliance requirements
- Intermixed methodologies
- Diverse collections of tools

Plutora unifies and streamlines all the disparate components involved and provides complete visibility and control of the portfolio. The result is not only accurate data but also unparalleled insights that will help guide your organization's continuous improvement in application delivery processes.

## A Shift in Technology

The third key area of change needed to achieve DevOps alignment is a shift in technology. Although tools alone will not be very effective, providing the right people with the right tools will greatly increase your chances of success.

Primarily, these tools involve incorporating automation. Doing this allows tasks to be easily repeated. Likewise, it reduces the risk of individual error, which makes it more likely that anyone can reliably accomplish the task. Automation can also help you reduce costs, improve testing, and speed up releases.

Other key DevOps tools include:

- Source code repository
- Build server
- Configuration management
- Virtual infrastructure

## Roles and Responsibilities

Successfully shifting an organization to become more DevOps aligned is a complex process. We've discussed the necessary tools and technologies, but relying on tools alone is not enough. Effectively incorporating DevOps depends on having the right people with the right skills and a willingness to collaborate.

DevOps doesn't mean including a third category of people in addition to those working in development or operations. Rather, DevOps involves aligning everyone toward a common goal—a streamlined process for creating business value.

"This all sounds great," you might say, "but what should a DevOps team look like?"

There's no standard format for a DevOps team. However, consider including these roles as a minimum.

## DevOps Evangelist

We've already discussed the need for a significant cultural shift when adopting DevOps. Promoting and driving that kind of change requires a leader. Often known as the DevOps evangelist, this individual owns the change and is familiar with the benefits of DevOps. More important, this person is able to communicate these benefits to other members of the team. This communication ensures buy-in and a unified commitment to change.

The DevOps evangelist is also responsible for the success of DevOps processes and people. The evangelist determines which roles are necessary to optimize the process and what training is needed so that everyone is prepared and empowered to make the necessary changes.

## Product Owner

The product owner is the key stakeholder in the project. This person holds the vision for the final product and communicates it to other team members by prioritizing the backlog. They are also responsible for keeping product feature development in line with business priorities.

The product owner is traditionally a very "project-centric" role, but product owners in DevOps organizations must shift their focus to the bigger picture. Instead of fixating on implementing features, the goal becomes efficient operations over the entire life cycle of the product. In addition to functionality, so-called nonfunctional tasks—such as adding logging or database optimization—should be equally important. These are related to the optimal running and operating of the product, both now and over its lifetime.

## Release Manager

Also known as a release engineer or product stability manager, the release manager is responsible for overseeing the overall progress of

a release. This includes managing the integration and coordination of development, testing, and deployment to support continuous delivery. In this regard, a release manager is similar to a traditional project manager. Unlike a project manager, however, the release manager also needs technical skills and knowledge to run and maintain the entire application delivery tool chain, as well as to measure and interpret metrics on all tasks.

### Automation Architect

Given the importance of automation within DevOps, the automation architect plays a vital role. Also known as integration specialists or automation experts, their job is to analyze, design, and implement strategies and tools for continuous deployment.

The goal of the automation architect is to provide an efficient and reliable automated environment for other team members. This role becomes especially important with distributed teams.

### Software Developer/Tester

The role of software developer is integral to any software organization. Within a DevOps environment, however, the role comes with increased responsibility.

In addition to writing code to meet specified business requirements, developers must also perform unit testing, deployment, and ongoing monitoring. Incorporating testing into the role of the developer makes finding and fixing issues more efficient. Of course, in order to maintain quality and improve efficiency, it helps to automate the testing process as much as possible.

### Experience Assurance

Most people are already familiar with quality assurance. QA team members confirm the quality of a product by determining whether it meets requirements. Within DevOps organizations, a new type of control becomes necessary. Instead of simply testing functionality, team members must test the overall user experience as well. Experience assurance (XA) professionals make certain that the final product has all the features that were originally specified.

### Security Engineer

In traditional software development, security is often an afterthought. Increased threat of attack and fear of noncompliance are strong motivations to make security a priority, but adding it on at the end is still not enough. In aligning with DevOps, it is important for teams to build security into the product. Beginning early in the process, security engineers work with developers and make recommendations. The result is a final product that resists attacks.

## Shifting Left

“Shifting to the left” means moving something into earlier stages of the product life cycle. It started in the 1990s, when people discovered that the then-standard waterfall methodology resulted in poor-quality software that required expensive fixes. Testing was happening too late in the production timeline—in other words, too far “to the right.” We now know that discovering defects later in the pipeline makes them increasingly expensive to fix.

### Testing and Deployment

In DevOps, shifting left requires two key practices: continuous testing and continuous deployment.

Continuous deployment results in regular build deployments, which allows continuous testing to take place quickly and efficiently. Both of these practices are possible through the use of cross-functional teams instead of the old model of development, operations, and QA teams operating in separate silos.

Moving testing into earlier stages of the pipeline is one method of shifting left. The role of software developer/tester is an obvious example. This role incorporates the need for testing in early stages of development. Release managers also work directly with continuous testing and continuous deployment.

Even with testing, sometimes unstable code can be introduced into the release branch. If a developer commits work that causes the build to fail, the result is not only compromised quality and decreased velocity but also a breakdown in trust between team members as they try to locate the source of the issue. A simple way to solve this problem is by introducing a gated check-in system. This is a software integration pattern that allows code to be verified prior to each commit—and only incorporated once it has returned successfully.

### Other Ways to Shift Left

Shifting left applies to more than just testing and deployment. The security engineer shifts security into the development stage. Automation architects build automation into the pipeline as early as possible. DevOps evangelists, product owners, and experience assurance professionals also benefit from continuous deployment and testing by using the feedback they receive to improve design early on.

Finally, shifting left occurs when non-functional requirements are incorporated throughout the development lifecycle. During each stage of the process, these requirements must be addressed in order to meet the given acceptance criteria. The result is that tasks related to the operation of the system are not pushed off in favor of feature development, which would make them increasingly expensive to implement.

## Conclusion

Becoming DevOps aligned can be challenging. Company culture, established processes, and tools and technologies used will all require significant changes. This process won't happen overnight. However, the benefits of incorporating DevOps vastly outweigh the challenges it presents. Shorter development cycles, faster innovation, reduced failures, recovery time, and better collaboration are just some of the benefits of a DevOps organization.

## Want to learn more about DevOps?

Check our series of white papers about DevOps to learn
from the foundations, to the cultural change and maturity model.

1. What is DevOps?
2. The Benefits of DevOps Strategy
3. DevOps Methodology: Aligning your Organization
4. DevOps Tools: Why You Need Them
5. The DevOps Maturity Model
6. DevOps Pipeline: The Functional Building Blocks
7. Mastering the DevOps Process
8. Making your DevOps + Agile Transformation a Success

Visit www.plutora.com/devops to learn more.

## About Plutora

Plutora, the market leader of value stream management solutions for enterprise IT, improves the speed and quality of software creation by capturing, visualizing and analyzing critical indicators of every aspect of the delivery process. Plutora orchestrates release pipelines across a diverse ecosystem of development methodologies, manages hybrid test environments, correlates data from existing toolchains, and incorporates test metrics gathered at every step. The Plutora Platform ensures organizational alignment of software development with business strategy and provides visibility, analytics and a system of insights into the entire value stream, guiding continuous improvement through the measured outcomes of each effort.

PLUTORA®

Learn more: www.plutora.com
Email: contact@plutora.com