DEVOPS METRICS FOR OPTIMIZING CI/CD PIPELINES

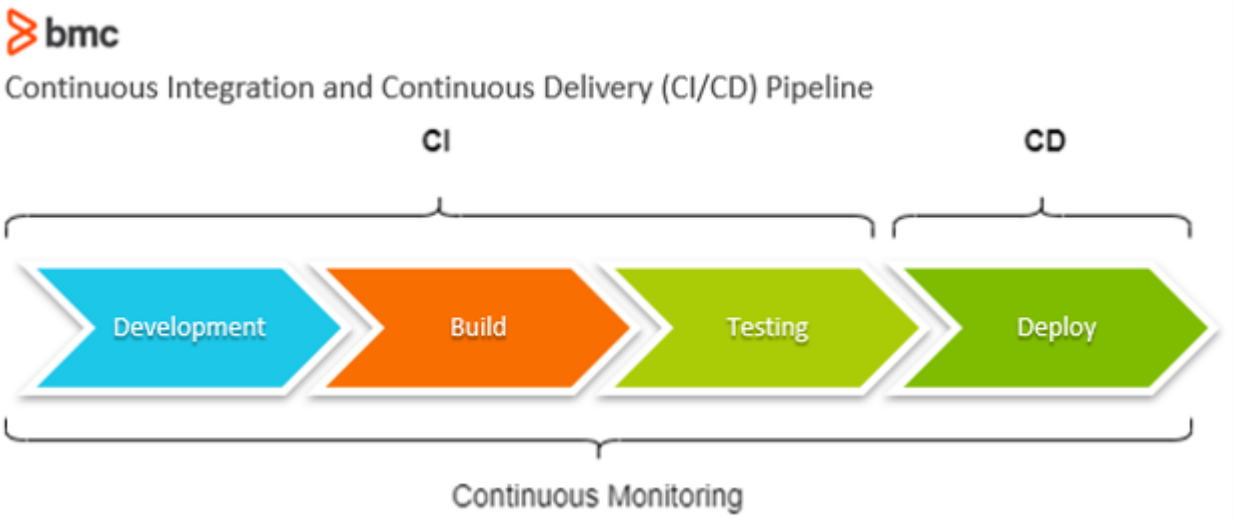
DevOps organizations monitor their CI/CD pipeline across three groups of metrics:

- Automation performance
- Speed
- Quality

With continuous delivery of high-quality software releases, organizations are able to respond to changing market needs faster than their competition and maintain improved end-user experiences. How can you achieve this goal?

Let's discuss some of the critical aspects of a healthy CI/CD pipeline and highlight the key metrics that must be monitored and improved to optimize CI/CD performance.

(This article is part of our DevOps Guide. Use the right-hand menu to go deeper into individual practices and concepts.)



CI/CD brief recap

But first, what is CI/CD and why is it important?

Continuous Integration (CI) refers to the process of merging software builds on a continuous basis. The development teams divide the large-scale project into small coding tasks and deliver the code updates iteratively, on an ongoing basis. The builds are pushed to a centralized repository where further automation, QA, and analysis takes place.

Continuous Delivery (CD) takes the continuously integrated software builds and extends the process with automated release. All approved code changes and software builds are automatically released to production where the test results are further evaluated and the software is available for deployment in the real world.

Deployment often requires DevOps teams to follow a manual governance process. However, an automation solution may also be used to continuously approve software builds at the end of the software development (SDLC) pipeline, making it a Continuous Deployment process.

(Read more about CI/CD or set up your own CI/CD pipeline.)

Metrics for optimizing the DevOps CI/CD pipeline

Now, let's turn to actual metrics that can help you determine how mature your DevOps pipeline is. We'll look at three areas.

Agile CI/CD Pipeline

In regard to delivering high quality software, infusing performance and security into the code from the ground up, developers should be able to write code that is QA-ready.

DevOps organizations should introduce test procedures early during the SDLC lifecycle—a practice known as shifting left—and developers should respond with quality improvements well before the build reaches production environments.

DevOps organizations can measure and optimize the performance of their CI/CD pipeline by using the following key metrics:

- **Test pass rate.** The ratio between passed test cases with the total number of test cases.
- **Number of bugs.** The number of issues that cause performance issues at a later stage.
- **Defect escape rate.** The number of issues identified in the production stage compared to the number of issues identified in pre-production.
- **Number of code branches.** Number of feature components introduced into the development project.

Automation of CI/CD & QA

Automation is the heart of DevOps and a critical component of a healthy CI/CD pipeline. However, DevOps is not solely about automation. In fact, DevOps thrives on automation adopted strategically—to replace repetitive and predictable tasks by automation solutions and scripts.

Considering the lack of skilled workforce and the scale of development tasks in a CI/CD pipeline, DevOps organizations should maximize the scope of their automation capabilities while also closely evaluating automation performance. They can do so by monitoring the following automation metrics:

- **Deployment frequency.** Measure the throughput of your DevOps pipeline. How frequently can your organization deploy by automating the QA and CI/CD processes?
- **Deployment size.** Does automation help improve your code deployment capacity?
- **Deployment success.** Do frequent deployments cause downtime and outages, or other performance and security issues?

Infrastructure Dependability

DevOps organizations are expected to improve performance *without* disrupting the business. Considering the increased dependence on automation technologies and a cultural change focused on rapid and continuous delivery cycles, DevOps organizations need consistency of performance across the SDLC pipeline.

Dependability of infrastructure underlying high performance CI/CD pipeline responsible for hundreds (at times, thousands) of delivery cycles on a daily basis is therefore critical to the success of DevOps. How do you measure the dependability of your IT infrastructure?

Here are a few metrics to get you started:

- **MTTF, MTTR, MTTD: Mean Time to Failure/Repair/Diagnose.** These metrics quantify the risk associated with potential failures and the time it takes to recover to optimal performance. Learn more about reliability calculations and metrics for infrastructure or service performance.
- **Time to value.** Another key metric is the speed of Continuous Delivery cycle release performance. It refers to the time taken before a complete written software build is released into production. The delaying duration may be caused by a number of factors, including infrastructure resources and automation capabilities available to test and process the build, as well as the governance process necessary for final release.
- **Infrastructure utilization.** Evaluate the performance of every service node, server, hardware, and virtualized IT components. This information not only describes the computational performance available for CI/CD teams but also creates vast volumes of data that can be studied for security and performance issues facing the network infrastructure.

With these metrics reliably in place, you'll be ready to understand how close to optimal you really are.

Related reading

- BMC DevOps Blog
- Deployment Pipelines (CI/CD) in Software Engineering
- Continuous Delivery Metrics
- DevOps Team Structure
- Choosing IT Metrics That Matter