GitLab

How to measure success in software development

Metrics and measurement frameworks tech leaders need to know.

Table of Contents

Introduction

In order for organizations in any industry to be successful today, they need to balance a host of responsibilities: providing high-quality digital experiences, keeping up with the latest tech, delivering value efficiently, and delighting customers, all while shipping secure, vulnerability-free code.

To manage this huge task, leaders need a reliable, scalable way to measure their success — because accurate metrics can lead to actionable insights. Tracking the right metrics can help teams optimize workflows, minimize disruptions, identify and resolve bottlenecks, enhance productivity, and improve developer experience. Put together, these pieces can ultimately help you make better business decisions and meet your goals.

But with all the challenges facing today's software teams, how do you know which metrics to track? And how can you implement these measurement frameworks without adding even more bottlenecks to your team's workflow?

This ebook will explore how engineering leaders can effectively measure the value of their team's software delivery, including industry-wide metrics and individual team performance stats. We'll also offer strategies on how to best implement these measurement frameworks for long-term operational efficiency and continuous improvement, from the first line of code to production and beyond.

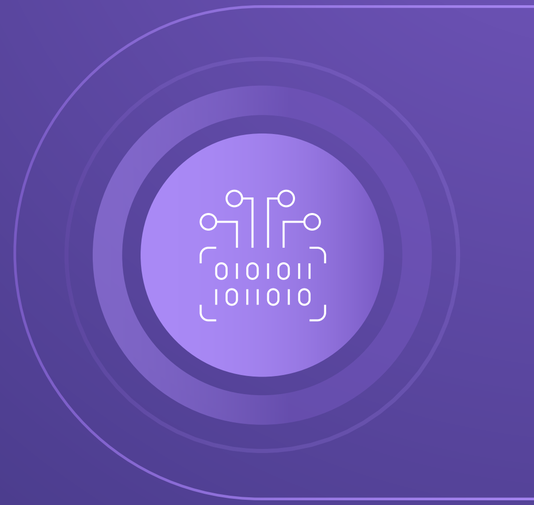
Common challenges in SDLC measurement

There are numerous challenges when it comes to software development lifecycle (SDLC) measurement. From a leadership standpoint, reliably measuring the business value of the software development process requires better visibility across the software supply chain and a deeper understanding of business metrics.

From a development team perspective, there are even more day-to-day hurdles:

- The complexity of fragmented toolchains and processes, siloed work environments, and lack of collaboration often get in the way of making lasting improvements in software delivery.
- Delays from code reviews — despite the fact that they help teams identify bugs, maintain compliance, and improve security — make it difficult for teams to find the right balance of speed and security.
- Shrinking IT budgets mean teams have fewer resources to put toward measurement.
- Teams are still trying to identify if their AI investments are worth it, while also grappling with all of their new AI-generated code.
- They simply don’t know which metrics are worth measuring and how individual metrics are connected.

Even after identifying a handful of metrics, organizations as a whole find it challenging to compare data between different development groups and know where they stand against industry benchmarks.



Key metrics to measure throughout the SDLC

While there are many, many metrics to measure just about anything in an organization — for example, you have revenue, sales, and profit margins to provide financial insights and visitor traffic and conversion rates to determine customer engagement — here are the ones we recommend you add to your software measurement dashboard.

As a note: It's better for organizations to measure team performance over individual performance. This approach better illustrates team collaboration and overall effectiveness. According to Deloitte, "Strengthening and enabling team performance helps organizations adapt and thrive, and better reflects the nature of how work is done." If your team is, in fact, working as a team, they should be evaluated as such.

"We believe metrics are a requirement for excellence. Metrics facilitate decision making. The more metrics you collect, quantitative and qualitative, the better and more informed decisions you can make. People will always have opinions on the value of the data or the meaning of the data, but using data as the basis by which to make a decision is often preferable to relying on opinion or intuition."

– 2024 DORA Report (p.74)

When U.K. **real estate company Zoopla** set out to measure improvements in the performance of their engineering department, they landed on the DORA metrics, which provided them with all the necessary insights to track success and benchmark against a definition of "good." And Zoopla chose the GitLab DevOps platform to obtain metrics for deployment frequency, lead time, change fail rate, and time to onboard. Since they began to track DORA metrics and make them available to internal teams for decision making and prioritization, Zoopla has been able to incorporate changes into their planning cycles to stay on the path of continuous improvement and build an efficient engineering team that is flexible in responding to business needs.

DORA metrics

DORA's software delivery performance metrics provide a quantitative, industry-standard measure of performance. Using these metrics as a gauge, teams are able to find areas for improvement, optimize their workflows, and ultimately drive positive business results. Once you measure DORA's four key indicators, you can determine if your team is Elite, High, Medium, or Low-Performing in terms of software delivery performance and effectiveness.

Here are the indicators (or metrics). The first two measure team velocity, while the second two measure stability.

Deployment Frequency:

What it is: How often an organization deploys code to production.

What it means: Higher deployment frequency indicates that your development team can deliver changes more rapidly, which can reflect a more agile and efficient software development process.

Change Lead Time:

What it is: The time it takes for a code change to go from commit to deploy.

What it means: Shorter lead times show that the team can quickly and efficiently convert your ideas into actual deployments, whether you need to launch a quick fix or a new feature.

Failed Deployment Recovery Time:

What it is: How long it takes to recover from a deployment failure and restore normal operations. In the **2023 DORA Report**, this indicator was adjusted to focus on deployment-specific issues rather than on general recovery.

What it means: The less time it takes to restore service means your system is more resilient and your team is capable enough to minimize downtime.

Change Failure Rate:

What it is: The percentage of changes that result in a service degradation, including incidents, bugs, or any changes that necessitate an immediate fix such as a rollback or hotfix.

What it means: If your team is able to lower the change failure rate, it means they're improving your code quality — and should give you greater confidence in your development process.

Productivity metrics

Along with your DORA metrics, there are productivity metrics, which measure how your team is performing in a number of areas throughout the SDLC. While DORA metrics help you compare your team to an industry benchmark, these metrics below can be gathered across teams and measured over a period of time to derive and build best practices for software development internally. When measured and improved upon all together, they can help drive better performance and business decisions.

Issue analytics

Issue analytics provide insights into the issues your team creates each month in a group or project. With these analytics, you can track the number of issues opened and closed each month, along with filters with details like name, age, state, milestone, iteration, assignee, and author. Ultimately, you can use these metrics to improve the overall turn-around time and value delivered to your customers.

Contributor analytics

This metric provides you with an overview of commits made by project members over a certain period of time. It can help you determine how much certain team members are contributing to your projects and measure individual and group productivity.

Merge request analytics

Merge request analytics can include the number of merge requests your organization merged in a month, the average time between merge request creation and merge event, and any other metrics, like milestones, commits, line changes, and assignees.

You can use merge request analytics to identify low or high productivity months, and your overall development velocity. Then you can take the insights to make data-driven decisions when it comes to:

- How to allocate your resources: for example, you might reallocate resources or adjust timelines during low productivity months
- Performance benchmarking: like highlighting your high-performing teams
- Milestone planning: adjusting timelines based on your merging trends
- Process optimization: identifying and resolving bottlenecks in code review and merging workflows

Security metrics

To keep your code secure and compliant, you should always be tracking your critical vulnerabilities and their severity. Tracking these metrics can help you understand your organization's security exposure in the context of your software delivery workflows, and improve your security posture for the future.

Speeding up code reviews is one of the most effective paths to improving software delivery performance. Teams with faster code reviews have 50% higher software delivery performance.

- 2023 DORA Report (p. 4)

How to implement measurement frameworks

The ultimate goal for many Agile and DevOps teams is to demonstrate the business value created by software delivery. Understanding the metrics that matter to an organization is essential — and that's what we covered previously — but now it's time to figure out how to effectively measure them to know if you're on track. Here are three steps teams can take to effectively measure the value of your software delivery:

Use value stream management to visualize your end-to-end software delivery workstream

A value stream is the entire work process that delivers value to customers. By monitoring and managing value streams — instead of features and functionality — throughout the software development lifecycle, organizations can measure the value of software delivery efforts and optimize the impact of their activities across the DevOps lifecycle. What's more, value stream management can help organizations get a wider view across the software supply chain, helping to eliminate short-term delivery bottlenecks without losing focus on the long-term goal of delivering value.

Ultimately, value stream management is about creating visibility and transparency, which is one of the big challenges we discussed before. By gaining this new visibility, leaders can finally identify areas for improvement, create a culture of value-driven processes, and make data-driven decisions to better optimize their team's workflows and create better customer experiences.

Cube Example:

Value stream management, which is aimed at improving security and DevSecOps processes, has been important to teams at Cube. And they took big steps forward, using GitLab's platform to create a value stream dashboard that notes where roadblocks and slowdowns are happening. If they can see what is happening and when it's happening in the process, it's much easier and faster to fix the problem.

"It keeps us on top of our service level agreement response time," explains Booijink. "When our clients start a new ticket for a project, our response time is part of the deal. By monitoring those dashboards, we can see where we stand with expected time to market. And we have to give clients regular updates about our response time and the dashboards give us the information we need to give to them, and to be able to fix bottlenecks and improve our timing."

"Now we have a clear understanding of processes, numbers, and what's behind all that data. And it works the same way for every project."

Mans Booijink, Operations Manager, Cube

Tailor metrics to team needs with custom value streams

Collecting the right data to make decisions is crucial. Metrics that are relevant to one team or a specific stage of the DevOps lifecycle may be less impactful in other contexts. Teams may want to use value stream analytics to identify:

- The amount of time it takes to go from an idea to production
- The velocity of a given project
- Bottlenecks in the development process
- Long-running issues or merge requests
- Factors that are slowing down the software development lifecycle

Many value stream management tools offer customizable value streams to help teams monitor specific workflows tailored to their particular needs and identify high-priority blockers. With just a few clicks, users can identify a merge request stuck in code review or an issue waiting for approval and then quickly solve the problem, without wasting cycles on questions and clarifications.

> “Thinking about transformation holistically, we recommend creating dashboards and visualizations that combine both technical metrics (such as our four keys and reliability metrics) and business metrics. This helps bridge the gap between the top-down and bottom-up transformation efforts. This also helps connect your northstar, OKRs, and employee goals with the investments made in IT. They can help quantify the ROI.”
>
> **- 2024 DORA Report (p.74)**

From DORA: Think incrementally

According to the 2024 DORA Report, improvement work is achieved iteratively and incrementally. Here’s a step by step guide of how they recommend you implement your measurement frameworks.

1. Identify an area or outcome you would like to improve
2. Measure your baseline or current state
3. Develop a set of hypotheses about what might get you closer to your desired state
4. Agree and commit to a plan for improvement
5. Do the work
6. Measure the progress you’ve made
7. Repeat the process

Communicate

You can use all the tools you want, but sometimes, it’s important to sit down with your team and simply talk through your plan. Make sure you have team meetings and one-on-ones to make sure each person knows the plan and who is being held accountable for what. This helps to create a culture of trust.

On measuring AI's impact

According to the 2024 DORA Report, 81% of respondents reported that their organizations have shifted their priorities to increase their incorporation of AI into their applications. Forty-nine percent of respondents even described the magnitude of this shift as being either "moderate" or "significant."

One of the major challenges we mentioned before is that organizations are currently grappling with how to not only manage all of their new AI-generated code, but how to measure its impact. While it may seem like AI is improving code quality and reducing complexity, it's important for leaders to truly know either way, and to avoid pitfalls by uncovering any drawbacks.

To measure AI's impact, you need to think beyond traditional metrics like code acceptance rates. Think more holistically and include big-picture items, such as:

- Code quality and maintainability
- Innovation levels
- Overall software delivery performance, as measured by DORA
- Security vulnerability metrics
- Your revenue and cost reduction
- Developer experience

Value stream management is **fundamental for building an AI strategy**. To keep all of these metrics in one place, and track, say, month-over-month metrics for your AI usage with SDLC metrics, it can be useful to have a value stream management dashboard. Depending on what tools you have available, a dashboard may also help you measure how much time you save in your team's end-to-end workstream by checking monthly code suggestions usage rate (unique code suggestions users divided by total monthly unique contributors).

81%

of respondents reported that their organizations have shifted their priorities to increase their incorporation of AI into their applications.

- 2024 DORA Report

Step 1: Identify value streams that deliver business outcomes

First you need to figure out which processes and workflows in your SDLC are delivering value. Are there any bottlenecks or hurdles where AI can help? How can AI improve this delivery to help you reach your business goals? Be sure to consider your workflow from end-to-end and pay attention to value streams that may have regulatory and compliance requirements attached.

Step 2: Understand where AI fits in

Just like you would for people on your team, make sure that AI itself has a clearly defined role and responsibility. Everyone should understand how it's enabling flow along your value stream and how it's interacting with people and tools.

Step 3: Map the value stream

A value stream reference architecture will help you define team actions and map out the most efficient way AI can fit into your workflow.

Let's take a look at a very simple example. Say you're launching a new product feature (without the help of AI). Your current value stream would look something like this: Developers build the code, test it, scan it for any security vulnerabilities, and deploy it.

Let's say you notice a bottleneck at the security stage, and realize that an AI tool could be a potential solution to help developers understand and fix vulnerabilities more efficiently. Now you can map this into your value stream, showing exactly where AI tools can add value throughout your workflow.

Continuous improvement > perfection

When you have the right metrics and know how to measure them, there is endless possibility for improvement. With the metrics provided and smart implementation, organizations can:

- Increase revenue, accelerate speed, and reduce overall cost
- Streamline delivery, boost security, and justify investment
- Increase the effectiveness and efficiency of your development teams
- Focus on remediation efforts and at-risk projects

As the 2024 DORA Report puts it: “We encourage teams to set a goal to get better at getting better.”

If your teams can focus more on continuous improvement, as opposed to measurement for just measurement’s sake, you can expect better business outcomes and more stability, no matter what the future holds.

> “We encourage teams to set a goal to get better at getting better.”
>
> - 2024 DORA Report

Ready to start measuring the value of software delivery for your organization?

Spin up a free 60-day trial of GitLab Ultimate. >

About GitLab

GitLab is the most comprehensive AI-powered DevSecOps Platform for software innovation. GitLab provides one interface, one data store, one permissions model, one value stream, one set of reports, one spot to secure your code, one location to deploy to any cloud, and one place for everyone to contribute. The platform is the only true cloud-agnostic end-to-end DevSecOps platform that brings together all DevSecOps capabilities in one place.

With GitLab, organizations can create, deliver, and manage code quickly and continuously to translate business vision into reality. GitLab empowers customers and users to innovate faster, scale more easily, and serve and retain customers more effectively. Built on open source, GitLab works alongside its growing community, which is composed of thousands of developers and millions of users, to continuously deliver new innovations.

GitLab