axify

Understanding DORA Metrics

Your Complete Guide

Table of Contents

axify

Introduction

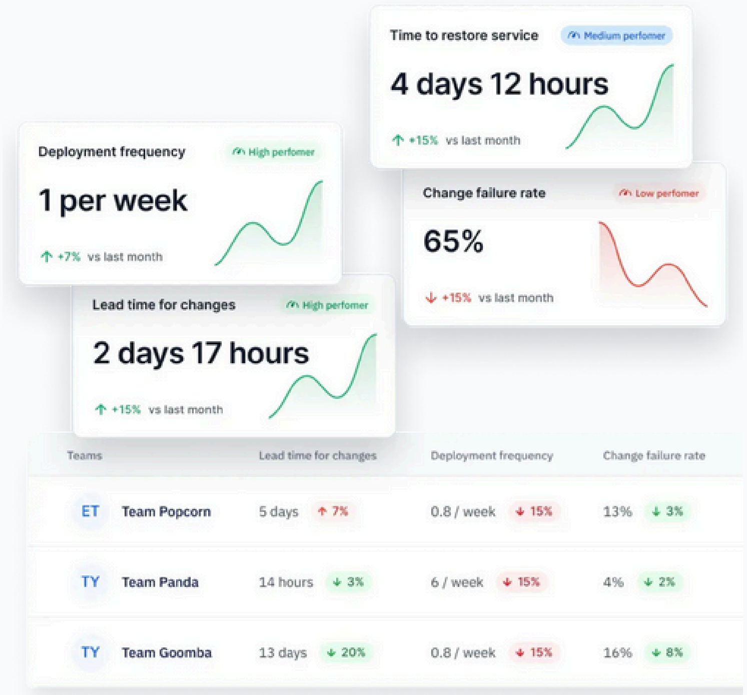
DORA Metrics 101

In an era where software development and delivery speed are crucial for business success, understanding and optimizing the key metrics that drive performance has never been more critical. The State of DevOps Report, published annually by the DevOps Research and Assessment (DORA) team, has been a pivotal resource for identifying these essential performance indicators. These metrics, commonly called DORA metrics, provide a clear, actionable framework for organizations striving to achieve high-performing software delivery processes.

This guide delves into the four core DORA metrics: Deployment Frequency, Lead Time for Changes, Failed Deployment Recovery Time, and Change Failure Rate. We aim to empower your teams to measure, analyze, and improve their software delivery performance by explaining each metric in depth.

Through theoretical insights and practical applications, you'll understand how to leverage DORA metrics to drive efficiency, quality, and reliability in your development processes.

Let's embark on this journey to elevate your software development practice to the next level!



Time to restore service — Medium perfomer
4 days 12 hours
↑ +15% vs last month

Deployment frequency — High perfomer
1 per week
↑ +7% vs last month

Change failure rate — Low perfomer
65%
↓ +15% vs last month

Lead time for changes — High perfomer
2 days 17 hours
↑ +15% vs last month

Teams		Lead time for changes	Deployment frequency	Change failure rate
ET	Team Popcorn	5 days ↑ 7%	0.8 / week ↓ 15%	13% ↓ 3%
TY	Team Panda	14 hours ↓ 3%	6 / week ↓ 15%	4% ↓ 2%
TY	Team Goomba	13 days ↓ 20%	0.8 / week ↓ 15%	16% ↓ 8%

About the Author

Meet Alexandre

Over the years, Alexandre has built the modern engineering culture that Nexapp (the team behind Axify!) is now known for. He has also become involved in the software development industry by sharing his knowledge with the Quebec tech community and the next generation of developers. He aims to help as many development teams as possible achieve elite delivery performance and focus on the user with fast product experimentation.

Since 2019, he has presented at the Quebec City and Montreal Agile Tour multiple times and was a speaker at DevOps Day Montreal in 2024. He's always up to talk about one of his favourite topics, DORA metrics, to as many people as possible.

Product Manager and CTO, Axify

Where Do DORA Metrics Come From?

DORA stands for DevOps Research and Assessment, a team that actively studies what differentiates a high-performing DevOps team from a low-performing team. In a seven-year program acquired by Google in 2018, this research group analyzed the DevOps practices and capabilities of over 32,000 DevOps professionals and identified four key metrics in 2020 to measure software development and delivery performance.

In addition to an official DevOps report published annually, the team also released a white paper on the ROI of DevOps transformation and the book *Accelerate: The Science of Lean Software and DevOps: Building and Scaling High-Performing Technology Organizations*, co-authored by Dr. Nicole Forsgren, co-founder of the DORA team and current partner at Microsoft Research.

Since there are many frameworks and methodologies to improve the way development teams build software products and services, DORA wants to shed light on what works and what doesn't in a scientific way.

Since their introduction, DORA metrics have been used by DevOps teams worldwide to measure their performance and compare themselves to high-performing teams.

What Are The Four DORA Metrics?

Deployment Frequency

How often a team puts an item into production.

Lead Time For Changes

Time required for a commit to go into production.

Change Failure Rate

Percentage of deployments resulting in production failure.

Failed Deployment Recovery Time

Time required for a team to recover from a production failure.

At a high level, Deployment Frequency and Lead Time For Changes measure **speed**, while Change Failure Rate and Failed Deployment Recovery Time measure **stability**. A team can achieve better business results by measuring these metrics and continually iterating to improve them. In 2021, the DORA team added a fifth metric, reliability, to the factors impacting organizational performance. In this guide, we will cover the first four metrics.

DORA studies show that teams with high delivery performance are twice as likely to meet or exceed their organizational performance goals.

axify

Deployment Frequency

As the name suggests, Deployment Frequency refers to the frequency of releases: it measures how often a team deploys a change into production.

Development teams with better delivery performance tend to deliver smaller and much more frequent deliveries to provide value to users more frequently, improve customer retention and stay ahead of the competition.

Benchmark

How often does your organization deploy code to production or release it to end users?

Elite performance	High performance	Medium performance	Low performance
On demand	On demand (multiple deploys per day)	Between once per week and once per month	Between once per week and once per month

Source: 2023 Accelerate State of DevOps, Google.

Why Measure Deployment Frequency?

Deployment Frequency, as the name implies, measures the number of deployments the team makes in a given period (e.g., by month, week, or day). Therefore, a high Deployment Frequency is a good indicator of the team's ability to make changes.

In addition, a stable Deployment Frequency (i.e., a regular deployment rate) can reflect an agile team delivering value continuously. This reduces the feedback loop and is invaluable to the team and the users.

Potential Causes of Low Deployment Frequency

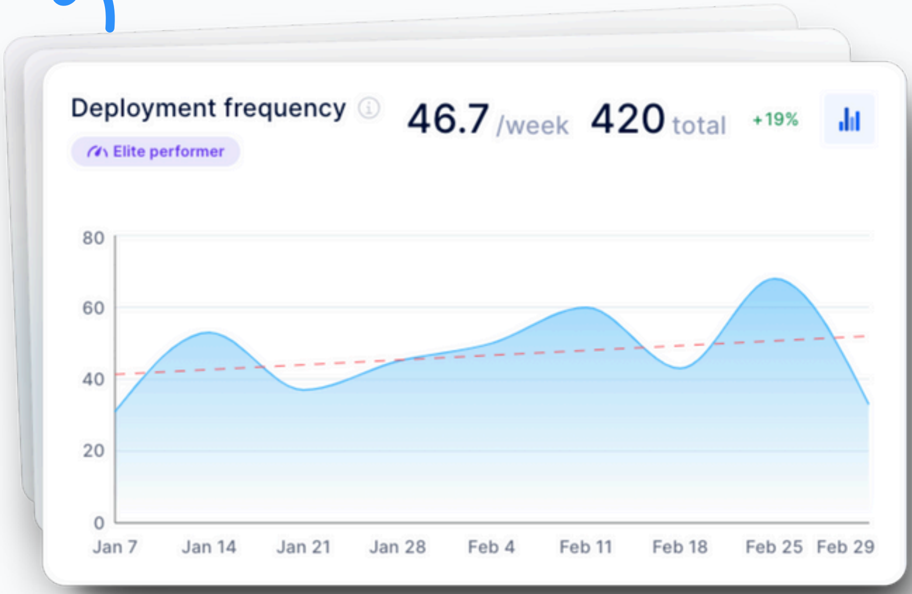
- ✓ Overly complex projects, significant changes or full feature development before deploying to production;
- ✓ Incomplete or insufficiently automated CI pipeline;
- ✓ Inability to send to production on demand (with one click);
- ✓ Bottlenecks, dependencies in the process or with another team (such as a DevOps team dedicated to production deployment, for example);
- ✓ Need for manual quality control or deployment or lack of automation;
- ✓ Inefficient tools, non-automated database migration, poor configuration management, production environment different from other environments, or need for manual manipulation;
- ✓ Batch deployment of multiple items at the end of a cycle.

How Can You Improve Deployment Frequency?

Regardless of where your team falls on the chart above, here are some ways to improve:

- ✓ Complete automation of the deployment process;
- ✓ Implementation of more automated tests on system behaviours to have more confidence in the changes (shift left testing) and be able to send them directly to production;
- ✓ Break down the work to be done into smaller items, both for tasks and pull requests;
- ✓ Reduce the number of intermediate environments.

The Deployment Frequency of a development team over two months in Axify. With 46.7 deployments per week, the team deploys changes to production more than nine times a day.



Lead Time For Changes

To measure Lead Time For Changes, we need two pieces of data: the exact time of the first commit and the exact time of the deployment in which it was made. We then use the average duration as a performance indicator: teams with better delivery performance can go from commit to production delivery in less than a day.

This indicator measures the time between the first commit and deployment to production. We use it to better understand the team's cycle time and analyze its responsiveness to constantly changing user requests and needs. The shorter the time required to make changes, the more responsive the team is and the faster it adds value.

Benchmark

How long does it take to go from code committed to code successfully running in production?

Elite performance	High performance	Medium performance	Low performance
Less than a day	Between one day and one week	Between one week and one month	Between one week and one month

Why Measure Lead Time For Changes?

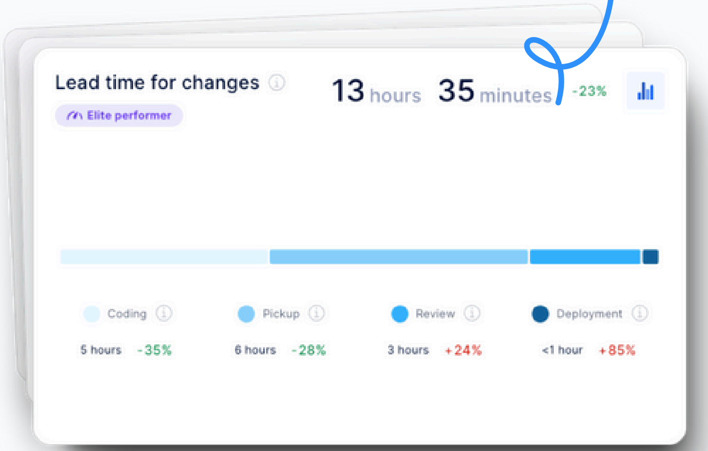
There are many reasons for delayed deployments, including fear of deployment failure or rejection of partial delivery, so development teams need an accurate picture of how long it will take to put changes into production. This metric is handy for organizations or managers tracking multiple contexts or teams, as it becomes a benchmark.

Lead Time For Changes indicates how quickly a team delivers changes to users. It represents flow efficiency, code complexity, and team capacity. Shorter delivery times are preferable, as they allow for a faster feedback loop on what you develop and for quick fixes.

Potential Causes of High Lead Time For Changes

- Bottlenecks and dependencies in the process;
- Lots of delay time where no one is actively working on tasks;
- Lack of deployment automation or difficulty getting to production quickly;
- Cumbersome and time-consuming code reviews;
- Lack of automated behavioural testing (compensated by manual testing by humans);
- Legacy code or no time allocated to pay off technical debt;
- Scope creep and over-engineering.

The Lead Time For Changes of a development team in Axify. Some phases are getting longer while others are getting shorter, but the overall process is 23% faster.



How Can You Improve Lead Time For Changes?

Regardless of where your team falls on the chart above, here are some ways to improve:

- ✓ Encourage collaboration and reduce work in progress (WIP) and the number of reviews performed in parallel;
- ✓ Implement more automated deployment and review processes;
- ✓ Break down tasks and features into smaller, more manageable items;
- ✓ Analyze time spent per stage of the development pipeline to identify bottlenecks;
- ✓ Do not depend on someone outside the team for code reviews;
- ✓ Encourage trunk-based development to make smaller changes frequently;
- ✓ Encourage test-driven development (TDD) or invest more in automated tests that are behaviorally oriented.

Analyzing your process steps to target automation opportunities can benefit your team. By reducing human handling (both in testing and automation), you'll help speed up the delivery rate. Add a progressive repayment of technical debt and a better breakdown of items, and you'll be well on your way to reducing the Lead Time For Changes!

Change Failure Rate

This metric captures the percentage of changes made to code that subsequently resulted in incidents, rollbacks or any other type of production failure. According to the DORA report, the best-performing companies fall around 5%.

The tricky part for most teams is defining an "incident". It can be system unavailability (partial or complete), unusable functionality, a critical bug, a major bug or any bug. According to DORA, these are changes to production or user releases that result in service degradation (e.g., service impairment or interruption) and require corrective action (e.g., hotfix, rollback, fix forward or patch). A definition that is too broad or too restrictive could be detrimental to the team, and the answer is highly dependent on the team's context.

Benchmark

What percentage of changes to production or released to users result in degraded service?

Elite performance	High performance	Medium performance	Low performance
5%	10%	15%	64%

Why Measure Change Failure Rate?

This is an indicator of code quality and stability. This metric shows the percentage of changes made that result in an incident. As mentioned earlier, the type of incident considered for this metric is specific to each team. However, it is essential to understand that a high failure rate implies an increase in the time spent reworking existing features, thus decreasing the time allocated to deliver new value to users. While not a time metric per se, Change Failure Rate can significantly affect your ability to create value quickly.

A low Change Failure Rate shows the team identifies errors and infrastructure bugs before deploying code. This is a sign of a robust deployment process and the delivery of high-quality software. The goal of the DevOps team should be to reduce the Change Failure Rate to ensure that the software is available and working correctly.

Potential Causes of High Change Failure Rate

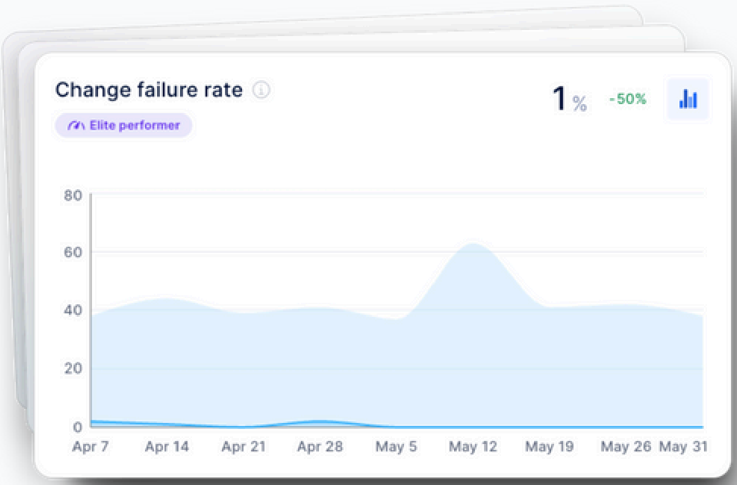
- ✓ Lack of automated testing, too much manual testing, ineffective or missing testing before deployment;
- ✓ Problems in the development process;
- ✓ Significant changes that require lengthy revisions and for which there is little feedback;
- ✓ Poor quality code or too much coupling;
- ✓ Difficulty maintaining and introducing new code;
- ✓ Non-reproducible infrastructure changes.

How Can You Improve Change Failure Rate?

Regardless of where your team falls on the chart above, here are some ways to improve:

- ✓ Reduce work in progress (WIP) in iterations;
- ✓ Encourage social development activities (e.g. pair programming, synchronous code review);
- ✓ Increase confidence in your automated tests, write more of them and encourage Test-Driven Development (TDD);
- ✓ Work with smaller iterations;
- ✓ Use static code analysis tools in your continuous integration;
- ✓ Create a team monitoring routine to increase team accountability for system availability and reliability;
- ✓ Create a recurring feedback loop;
- ✓ Automate database migrations or environment creation.

Teams that deploy few changes will see fewer incidents, but that doesn't necessarily mean they are more successful with the changes they deploy. Teams that deploy frequently may see more incidents, but if the Change Failure Rate is low, they will have an advantage because of their deployments' speed and overall success rate.



axify

Failed Deployment Recovery Time

Failed Deployment Recovery Time measures the time it takes for a service to rebound from an incident or service degradation. No matter how well a team performs, unexpected outages or incidents will occur. Since we cannot avoid incidents, the time it takes to restore or recover a system or application makes the difference.

This important metric encourages developers to build more reliable, available, and resilient systems. We typically measure it by tracking the average time between the moment an incident is reported and the moment it is restored. For the best-performing teams, this means restoring the service in less than an hour.

Benchmark

How long does it generally take to restore service when a service incident or a defect impacting users occurs?

Elite performance	High performance	Medium performance	Low performance
Less than one hour	Less than one day	Between one day and one week	Between one and six months

axify

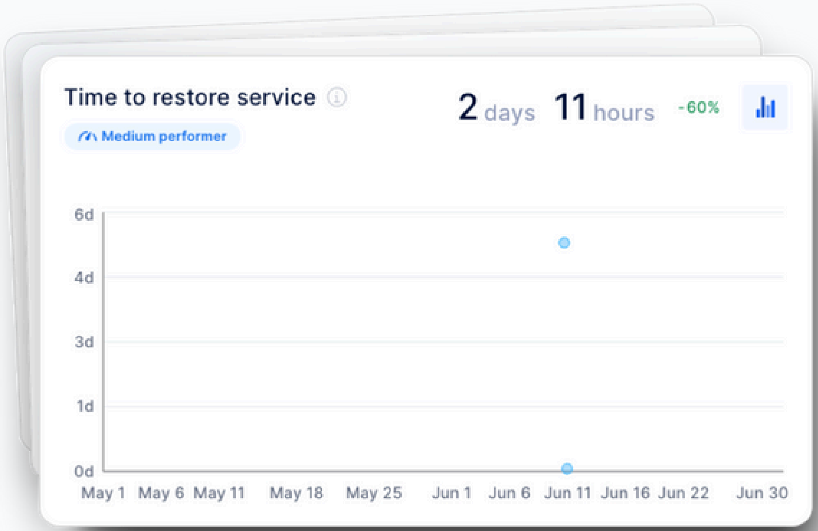
Why Measure Failed Deployment Recovery Time?

Failed Deployment Recovery Time indicates a team's response time and the development process's efficiency. When it is low, it demonstrates that a team responds and resolves issues quickly to ensure product availability and functionality.

In addition, a team's ability to recover quickly can positively impact users' confidence levels and make managers more comfortable with experimentation. This is a significant competitive advantage!

Potential Causes of High Failed Deployment Recovery Time

- ✓ Lack of automated testing;
- ✓ Dependence on an external team member or lack of access to specific system components;
- ✓ No shift in focus within the team to resolve the incident;
- ✓ Ineffective incident management process;
- ✓ Difficulty to deploy in production quickly;
- ✓ No rollback process;
- ✓ Ineffective tools.



How Can You Improve Failed Deployment Recovery Time?

Regardless of where your team falls on the chart above, here are some ways to improve:

- ✓ Promote continuous monitoring and improve observability;
- ✓ Prioritize recovery when a failure occurs;
- ✓ Try feature flags to be able to quickly disable a change without causing too much disruption;
- ✓ Deliver small iterations to make it easier to discover and resolve issues;
- ✓ Invest more in automated testing and use a behavioural testing strategy;
- ✓ Document the incident management process and train team members on how to respond to incidents;
- ✓ Where applicable, automate several steps in the incident management process and assign responsibility for actions you must perform manually.

Balance speed and stability! Allow yourself to slow down to achieve better performance by reducing the work in progress (WIP). Avoid putting sudden changes into production at the expense of a quality solution. Rather than deploying a quick solution, ensure your change is sustainable and tested. It would be best to track Failed Deployment Recovery Time to see how your team is improving and aim for steady, stable growth.

The Benefits of Tracking DORA Metrics

Continuous improvement

To improve, you need a starting point and a goal to achieve. DORA metrics give development teams concrete goals, which you can break down into key outcomes. By knowing what data to track over time, developers will adjust their behaviour to improve the metric, impacting team and process efficiency.

In addition, variations in the results of DORA metrics will help you quickly target areas for improvement in the delivery process and identify problems such as ineffective testing.

Better visibility

DORA metrics are reliable metrics resulting from up-to-date scientific studies. Because the four interrelated metrics are difficult to cheat, you have a realistic picture of your situation. They also allow you to compare your team to the rest of the industry, identify opportunities for improvement and make changes to optimize your results on many levels.

Decision-making support

Companies that streamline their development and delivery process increase the value of the software they develop and perform better in the long run. Tracking performance using DORA metrics enables DevOps teams to make decisions based on data, not gut feelings.

In addition, DORA metrics help align development goals with business goals. Finally, from a product management perspective, they provide insight into how and when development teams can meet customer needs.

Value generation

In recent years, value stream management has become essential to software development. In this context, DORA metrics are indispensable, as teams that measure themselves tend to improve continuously. So, when DevOps teams use DORA metrics, they typically see an increase in value over time.

As a set of proven DevOps benchmarks that have become the industry standard, DORA metrics provide a foundation for this process. They identify points of inefficiency or waste, and you can use this information to streamline and reduce bottlenecks in your workflows, such as deconstructing the Lead Time For Changes in steps to observe where the process is stuck. When your teams' DORA metrics improve, the efficiency of the entire value chain improves with them.

In addition, DORA metrics allow you to measure and improve yourself to ultimately:

- Deliver faster, ensuring greater stability;
- Deliver more value to customers by providing timely iterations and better responsiveness to feedback;
- Increase the business value of a product faster.

Balancing speed and stability

DORA metrics also provide insight into team performance. Engineering managers can ensure that their teams build robust products with minimal downtime by looking at the Change Failure Rate and Failed Deployment Recovery Time. Similarly, tracking Deployment Frequency and Lead Time For Changes ensures the team works quickly. Together, these metrics provide insight into the balance between speed and quality within the team.

By comparing these four key indicators, one can assess how well the organization balances speed and stability. For example, if the Failed Deployment Recovery

Time is less than a week with weekly deployments, but the Change Failure Rate is high, the team may be rushing deployments while lacking automated testing, or they may be unable to support the changes they are deploying.

On the other hand, if deployments occur once a month and the Failed Deployment Recovery Time and Change Failure Rate are high, the team may spend more time fixing code than improving the product. It's all about context!

The Biggest Challenges of DORA Metrics

Data accessibility

Most of the time, environments and data are decentralized, i.e., the data is dispersed in different sources throughout the organization. Moreover, extracting it can be complex when only available in raw format. Finally, the data must be transformed and combined into actionable units to allow development teams to take full advantage of it. In addition to the time required to perform these manipulations, manually calculating the data leaves room for interpretation, which presents a risk of error or communicating the wrong information.

Data analysis

Like all data, DORA metrics need to be put into context, and the story these four metrics tell together needs to be considered. For example, Lead Time For Changes and Deployment Frequency provide insight into a team's pace and how quickly it responds to evolving user needs. On the other hand, the Failed Deployment Recovery Time and Change Failure Rate indicate a software product’s stability and the team's responsiveness. Only a professional who understands the reality of the development team will be able to interpret this data correctly to contribute to their continuous improvement.

What You Need to Remember

In reality, metrics are just tools. What's important is that the development team wants to improve its effectiveness and use metrics to determine whether it's progressing in its continuous improvement efforts. So, give your team the tools they need to succeed in making the best changes that will help them achieve their goals.

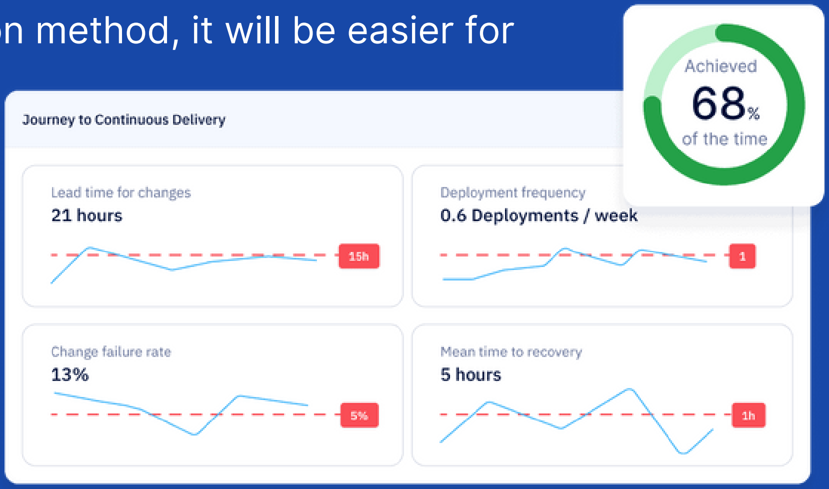
When your DORA metrics improve, you can be confident that you've made the right decisions to improve your team's performance and that you're bringing greater value to your customers.

Where Can You Start?

At Axify, we integrate DORA metrics into our dashboards to see all the factors supporting your team in continuous improvement at a glance. No data mining or ambiguity about interpretation. Everything is at your finger tips to stay on top of things! Plus, since all your teams will use the same structure and extraction method, it will be easier for you to compare apples to apples.

Learn more

Get started

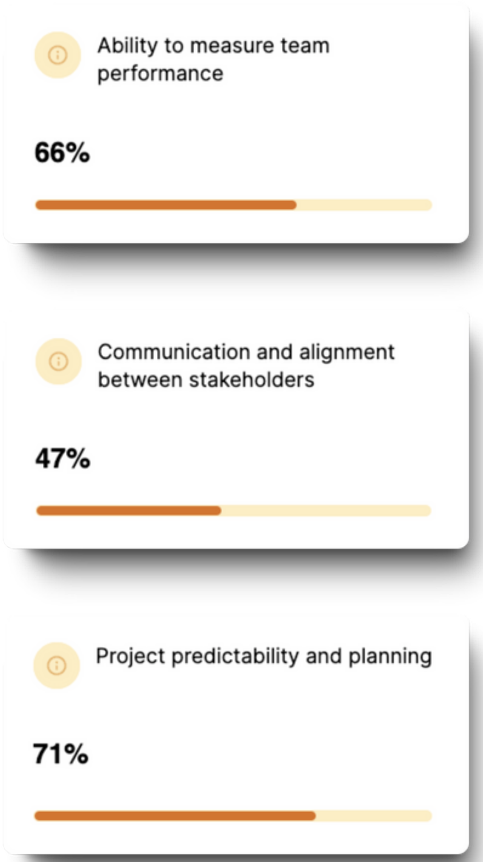


axify

How to Implement DORA Metrics in Your Process?

Once you've chosen a tool that allows you to track DORA metrics, don't focus on the metrics at first. Instead, start collecting data, follow the metrics for a few weeks, and then analyze what you need to improve.

Next, set goals for improvement: focus on your product and its growth, on growing your team, and on improving processes. Think of the DORA metrics as indicators of what you can do to positively impact the product and its business results.



More Resources

- Tool | Development team maturity analysis
- Blog post | The potential return of continuous improvement in software development
- Blog post | Software development metrics: to rely on your projections with confidence

About Axify

Connect your tools

Keep using your favorite tools like Jira, Azure DevOps, GitHub and GitLab to collect data on DORA metrics, the value stream, your team's well-being, processes and more.

Let Axify fetch your data

Setting up Axify takes less than 10 minutes. Data from your entire software development cycle is available in real time minutes after setup. So long Excel and .csv files!

Achieve elite performance

Get all your engineering metrics in one place to make informed decisions, align your development teams and improve your software delivery processes.

Contact us for more information on how Axify helps development teams measure DORA KPIs or request a demo.

info@axify.ca

Book a demo

+1 (418) 476-2090

axify