Software Metrics Leaders Can Measure to Track the Development Process

Managing large teams is hard. The amount of activity is overwhelming. But great engineering managers are able to solve these challenges by knowing what to measure in the process – the software performance metrics.

By getting a crystal-clear picture of the delivery process from idea to production, looking at what and how it was built, you can spot bottlenecks or process issues that, once cleared, increase the overall health and capacity of your engineering team.

That's why development analytics platforms are becoming more and more essential to effective processes and data-driven teams.

Waydev, for example, analyzes your codebase, PRs and tickets to help you bring out the best in your engineers' work. By automatically measuring performance, our dev analytics tool helps you to improve efficiencies, reduce cycle time, and increase development velocity.

In this guide, we'll share what software metrics to measure and how to use analytics to track development processes for a better performance + 15 process metrics examples for engineering teams to guide your process.

What Are Software Metrics?

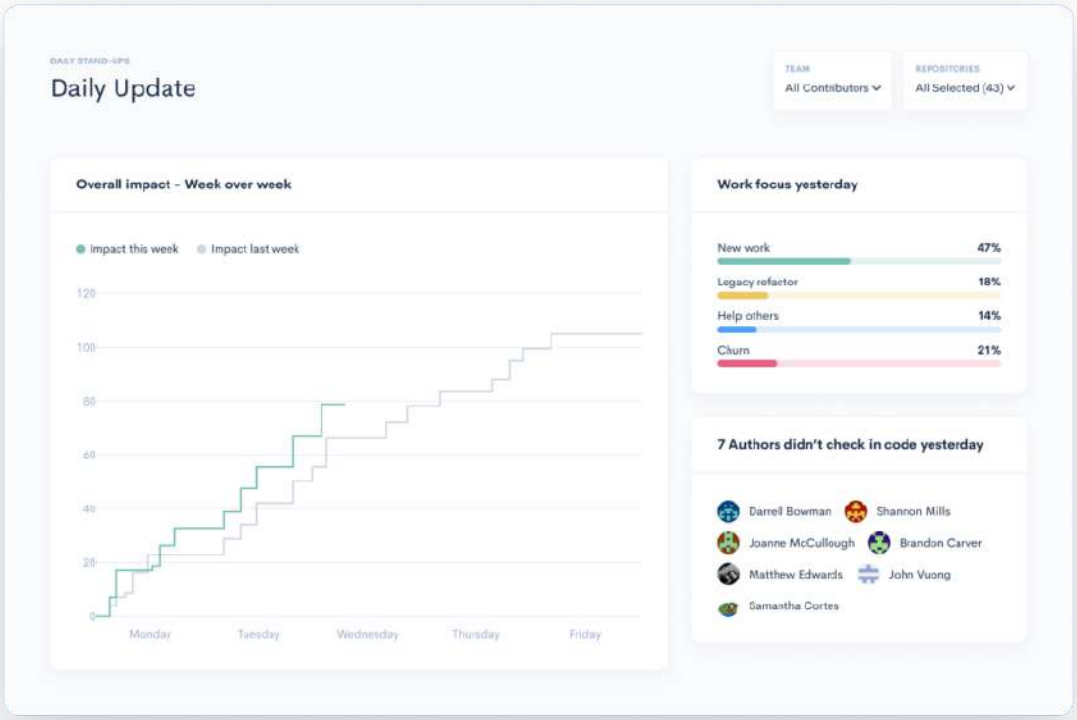
There are different types of software metrics used in software companies: code, productivity, quality and testing, and even customer satisfaction metrics.

Oftentimes, engineering managers and tech leaders look at more sets of specific metrics – you can see which ones in our 40+ examples of engineering KPIs and metrics.

Most are software metrics – or progress metrics – that measure key aspects of software. Within the right context, they help managers follow the progress and alignment of work with business initiatives.

- On one hand, both the client and the provider of a software project are able to analyze quantifiable results and objectively determine its efficiency and quality.
- On the other hand, managers can objectively evaluate the quality of the team's current work and deliverables, to be able to optimize the process and predict performance.

They provide an accurate overview of key aspects of development: resource allocation, planning and management, quality assurance, debugging, maintenance, performance. Nonetheless, they can also speak about engineers' performance, work trends, and patterns.



Let's take daily standups – they're great as check-ins to see the status of a sprint or the work in progress. With Waydev, you can run even more productive daily stand-ups. By tracking the commit and pull request activity, and velocity sprint over sprint, you spot roadblocks and work dynamics quicker. Then, you can start conversations on collaborative solutions to those issues.

That is why software engineering metrics – and analytics of the development process – are essential in tracking and improving the performance and quality of software development.

Waydev helps engineering managers ensure code quality and real-time access to practical insights into the entire development process. See how it works here.

Why You Should Track Development Processes

Using data across the development life cycle is one important way to enable your team to innovate and grow, and the software to perform better.

A good starting point is comparing your team's recent sprint against the previous one. This can also bring up an essential question: what do metrics usually look like for an engineering team like mine?

Managers need to take a look at the team's baseline data, in order to understand the starting point. And then, to experiment with the workflows and processes and notice trends.

This can be more easily done by looking into certain process metrics that measure the development process, such as those in the Waydev's Project Timeline feature (we'll get to that in the examples section).

To elaborate, leaders use software development metrics:

- To further increase return on investment (ROI);
- To easily spot process issues and prioritize areas of improvement;
- To effectively plan and manage workloads, resources, and estimates;
- To reduce bottlenecks and costs, for example when it comes to troubleshooting;
- To assess and boost performance and collaboration in a healthy work environment;
- To motivate additional resource allocation or delayed timelines to stakeholders.

How to Track Development Processes with Software Metrics

Before diving into the software metrics examples, first let's see how you can best decide on which ones to use for effective management.

1. Track Software Metrics That Are Easy to Measure

Developers shouldn't be burdened with performance reporting. On the contrary, they should feel supported in their work and have the right processes in place to progress and deliver quality software.

Here comes the development analytics' role. With Waydev, for example, you can access real-time data and automatic reports on performance. You can set, track, measure and visualize metrics, KPIs and reports that are relevant to your team, without their manual

input – which save you time and boost the well-being of your engineers.

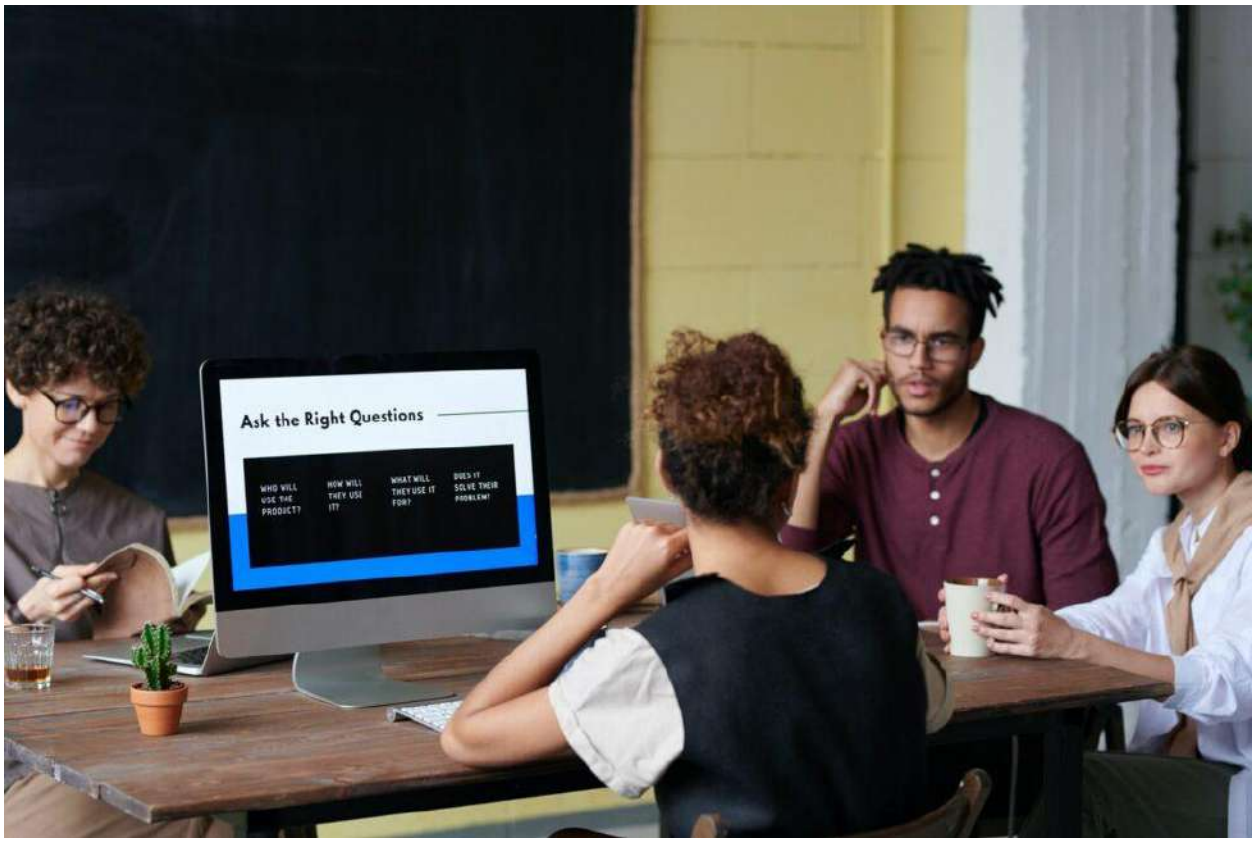
Waydev is already helping over 1,000 engineering leaders track development processes and deliver better products faster. Learn how it works.

2. Set Metrics That Are Relevant to Your OKRs

Leading tech organizations across the globe use metrics to support the setting and achieving of their engineering goals and OKRs. After all, it's about what you want to achieve and how you'll measure the success of it.

What's more, software developers feel the best when they work with hard data, concrete targets, and measurable results. That's why metrics are directly linked to each goal and company OKR – they measure results and set the direction for the team to follow.

Waydev helps you measure organization-level efficiencies to start optimizing your development process. You can track software metrics that measure vital aspects of quality development projects – reliability, performance, security, maintainability to name a few.



3. Use Metrics to Identify Work Trends and Patterns

We emphasize the importance of deciding on software metrics – or any kind of measurements – that help you spot and analyze trends and work patterns. It is not the numbers per se that are useful, but what insights they reveal in a certain context, over a certain period of time.

It is the process, the direction of the developers' efforts, the velocity and the impact of improvements along the way, that matter to the quality and delivery of software projects.

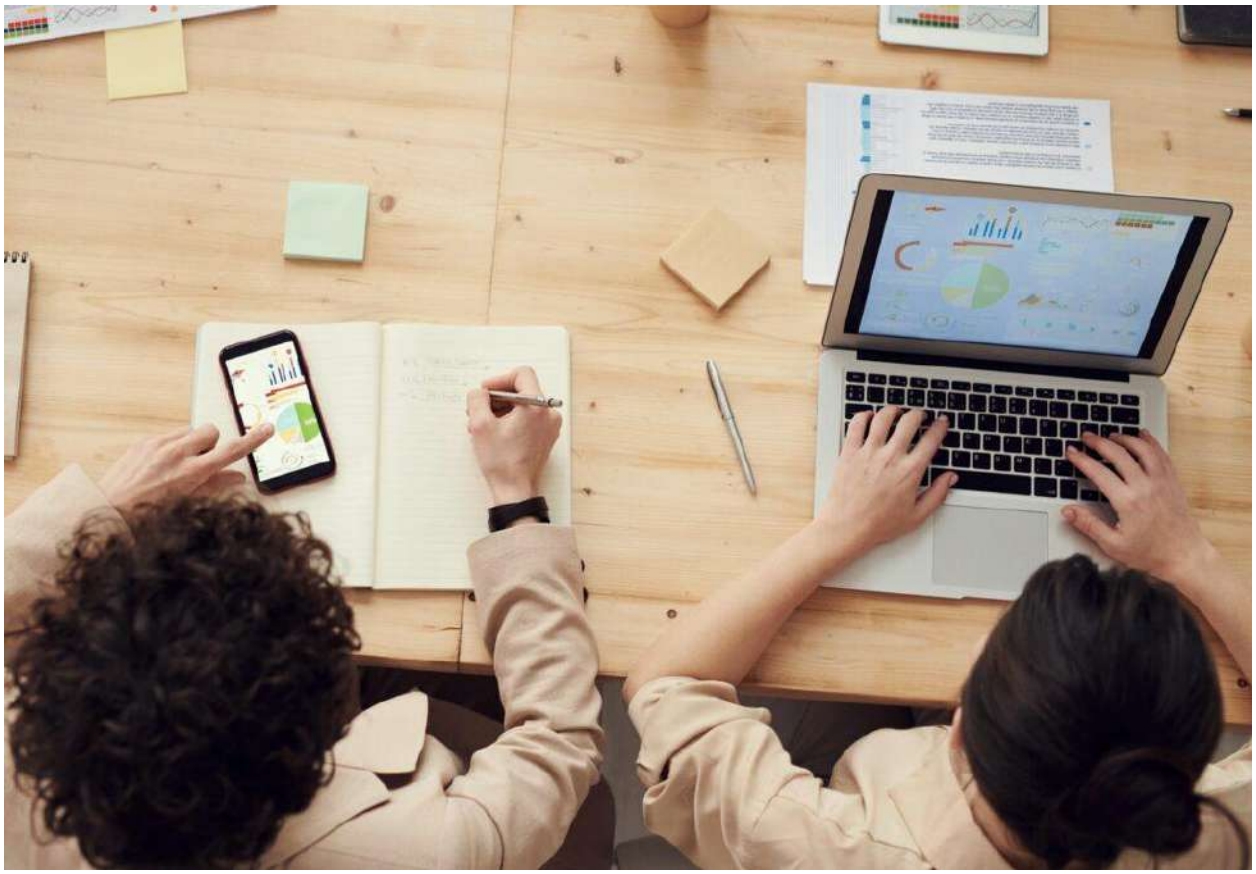
4. Measure Metrics, the Agile Data-Driven Way

We've talked about metrics being easy to measure, linked to goals, and insightful. This has to be done over a specific time period, in order for managers to be able to implement incremental improvements and see their impact.

So, another key aspect to consider is to measure these process metrics over short periods. For example, from one sprint to another or any agile data-driven approach.

This leaves room for work iterations and changes that allow for more releases, more stable software, less expensive troubleshooting, and higher customer satisfaction, to name a few.

Working remotely? Waydev can help. Learn how to gain visibility into your engineering team's activity with Waydev's features and software metrics.



Different Types of Software Metrics

As we emphasized before, software development metrics mean different things to different teams, since they depend on the context of organizational goals and team OKRs.

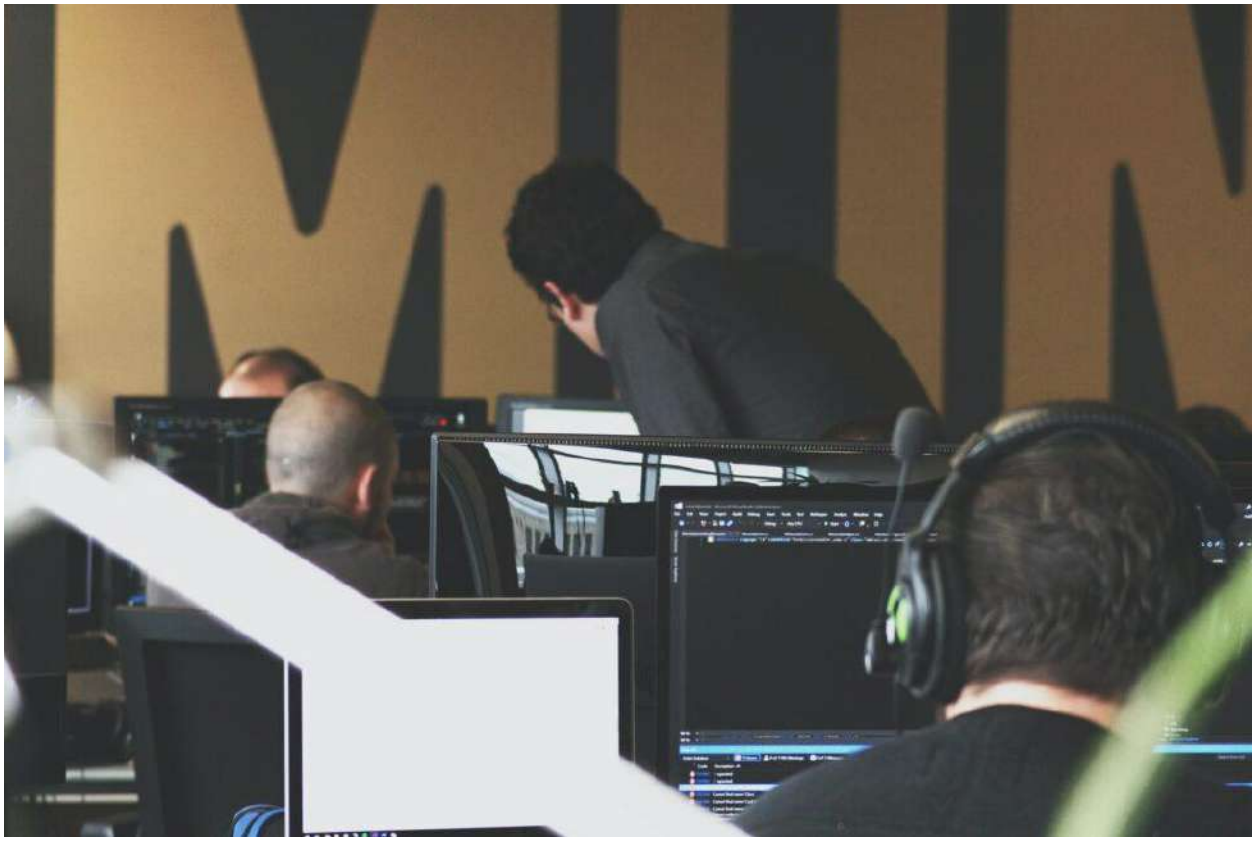
You will find classifications depending on role, function or model. The following examples of process metrics are based on the Consortium for IT Software Quality model (CISQ):

- Reliability metrics – production incidents, average failure rate, load testing, MTTR metrics;
- Performance metrics – load testing, stress testing, soak testing, application performance monitoring metrics;
- Security metrics – number of vulnerabilities, time to resolution, deployment of updates, number and severity of security incidents
- Maintainability and code quality metrics – static code analysis, code complexity, lines of code.

The majority of these metrics are hard to quantify and have to be taken into a specific context, as they cannot provide the full picture alone.

Most development teams measure process metrics based on their role or function, such as code metrics, developer productivity metrics, agile process metrics, operational metrics, test metrics, even customer satisfaction metrics.

At Waydev, we aim to track metrics based on what usually engineering teams and managers need. This is what we'll try to show you with the next examples.



What to Measure in Software Engineering: 15 Examples of Process Metrics

Here are a few examples of software metrics that measure the development process and software performance in engineering. You can automatically measure these software metrics with our dev analytics tool.

Agile Process Metrics

These process metrics and sets of metrics can help you measure the quality of the software deployment. They track the progress of a software development team in producing functional, high-quality features ready to be shipped.

1. Lead Time

The Lead Time metric measures how much time passes between task creation and work completion. It gives you a larger perspective on how long it takes for a client's requested feature to be completed.

It can help you identify potential issues that happen between task creation and the moment a developer starts working on it, to understand if you follow an Agile process.

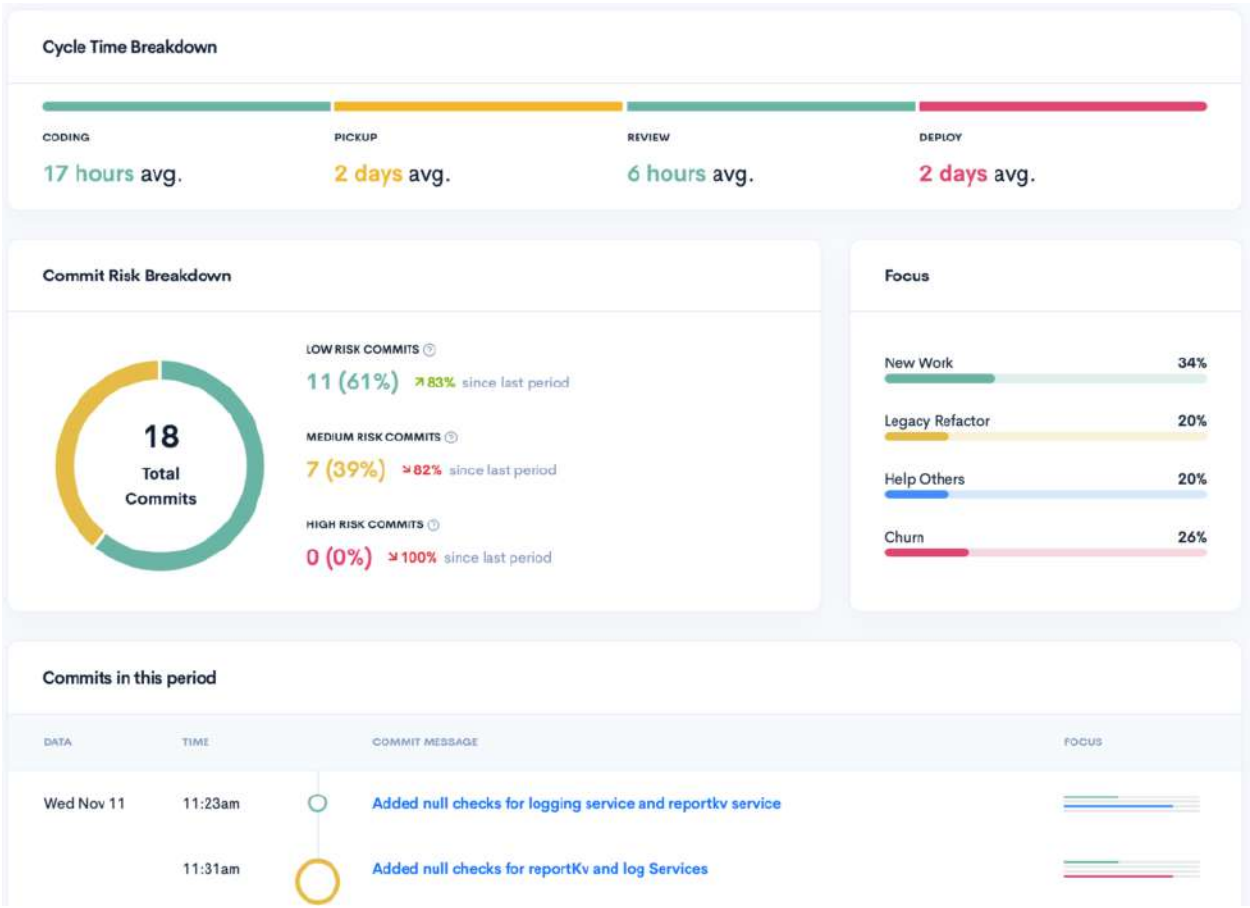
You can identify and unblock those issues that can lead to friction and frustration for end-users (or customers). Consider this a "user point of view".

2. Cycle Time

The Cycle Time metric measures the time spent on starting and completing a task. It is important in assessing an organization's development velocity.

It is actually composed of several Agile process metrics corresponding to a stage in the software development process: coding, pickup, review, deploy:

- Time to Issue PR from First Commit. This metric refers to the coding time, as the time elapsed from the first commit to creating a pull request.
- Time to First Review. This metric measures how fast reviewers pick up their peers' PRs for review, and is the time spent between opening a PR and the first time an engineer reviews it.
- Time to Merge from First Review. This metric indicates how fast submitters implement their peers' feedback in code review, and is the time spent between a PR's first review and that PR being merged.
- Time to Deploy from Merge. This indicates how fast code gets deployed, as the time between the moment a PR is merged to when it gets released into production.



3. Team Focus

It gives you an overview of the types of work in which your team is spending effort. Knowing when there are unusual churn spikes or code blockers can help you quickly address these issues – for example, if a development timeline needs to be pushed out an iteration or two.

In Waydev, check these metrics that indicate the types of work engineers focus on:

- New work – brand new code that does not replace other code.
- Legacy refactor – code that updates or edits old code. It's paying down "technical debt", which is usually hard to see. The challenge is to balance refactoring with developing new features. Objectively tracking the percentage of time spent on new features vs. maintenance helps maintain that balance of progress with codebase stability.
- Help others – how much an engineer is replacing another engineer's new code.
- Code churn – indicates when an engineer rewrites their own code shortly after it was written. Unusual spikes can indicate an engineer is stuck, or there are specs issues.

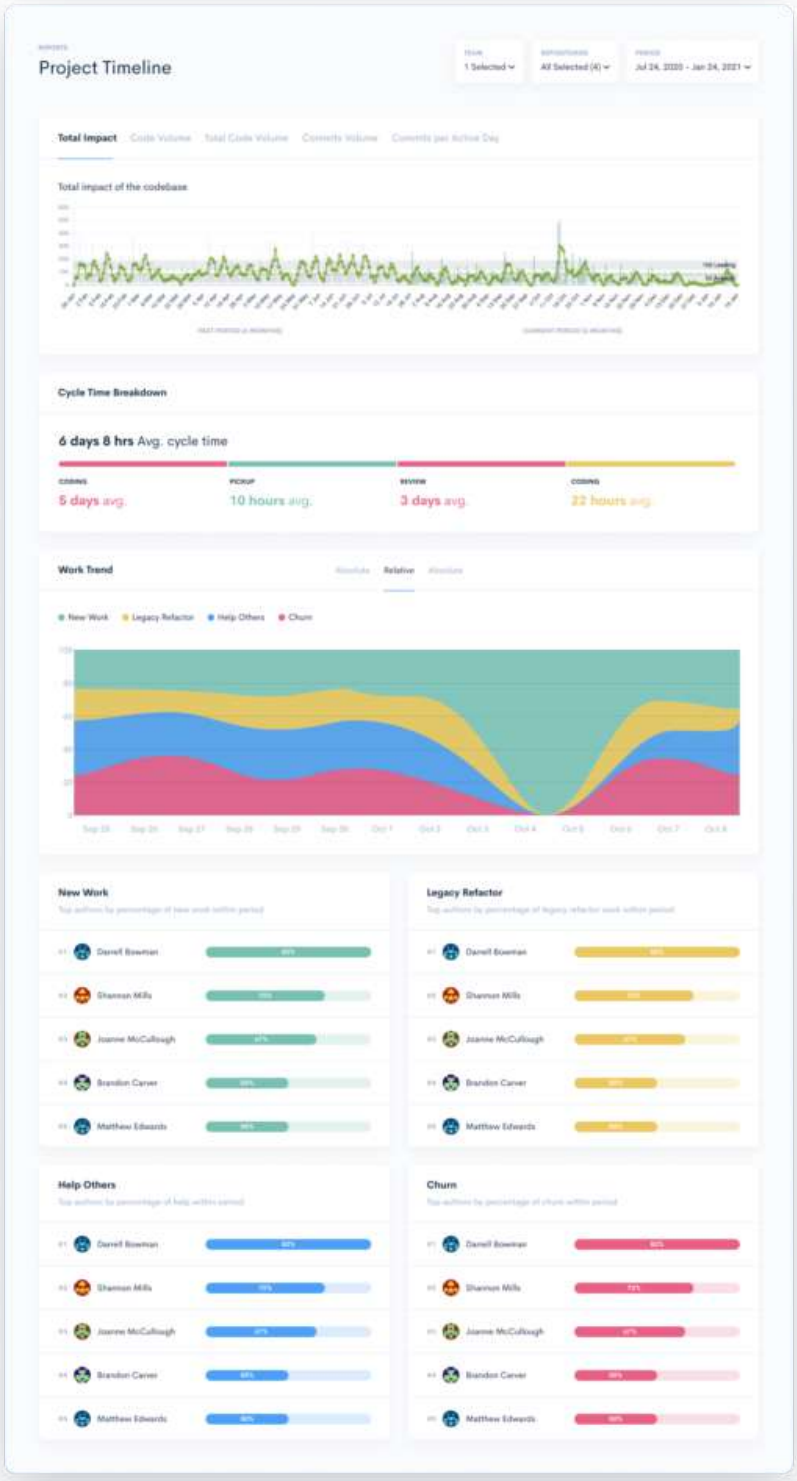
4. Project Timeline

The Project Timeline report shows you the most actionable metrics in code-based activity. It helps engineering leaders see what their teams are working on during a sprint, over a week, a month, or a custom time frame.

This is an aggregation of multiple software metrics, such as:

- Commits per active day – the average number of commits per team per active day
- Commits volume – quantifies the number of commits a team created in a day
- Total impact – shows you the total impact of the codebase in a day
- Code volume – the raw number of lines of code changed in the codebase

When you look at these process metrics, you can see if the results are aligned with what you'd expect, relative to prior periods and current priorities. Depending on the situation, it might mean it's possible for the team to build better features faster or, on the contrary, to ask for additional resources, or adjust the delivery dates.


Project Timeline
Total Impact
Total impact of the codebase
Cycle Time Breakdown
6 days 8 hrs Avg. cycle time
5 days avg.
10 hours avg.
3 days avg.
22 hours avg.
Work Trend
Relative
New Work
Legacy Refactor
Help Others
Churn

5. Code Review

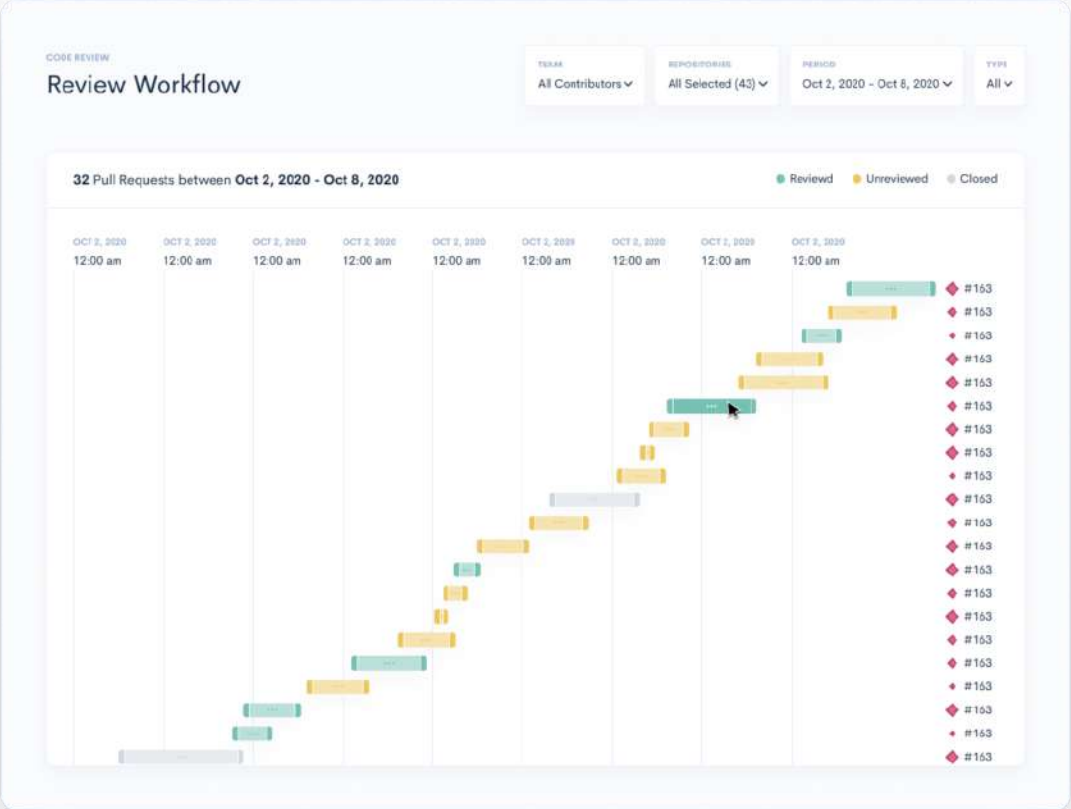
Another important question that is likely to come up frequently after a sprint is, how long did it take for the team to complete the code review?

To answer this particular question, we built the Code Review workflow report in Waydev. This focuses on six core PR cycle metrics:

- Time to resolve a PR
- Time to first comment
- Number of follow-up commits
- Number of reviewers on a PR
- Number of review comments on a PR
- Average number of comments/reviewer.

While agile scrum and other methodologies can help engineers iterate easily and ship faster, hidden issues and bottlenecks can disturb the development process.

By having an overview on the bottlenecks or outliers in your team's PR cycles over a sprint, you can help your team work better with a healthier and more collaborative and productive code review process.



Production or Operational Metrics

These software metrics measure and highlight how software is running in production and how effective the development team is at maintaining it.

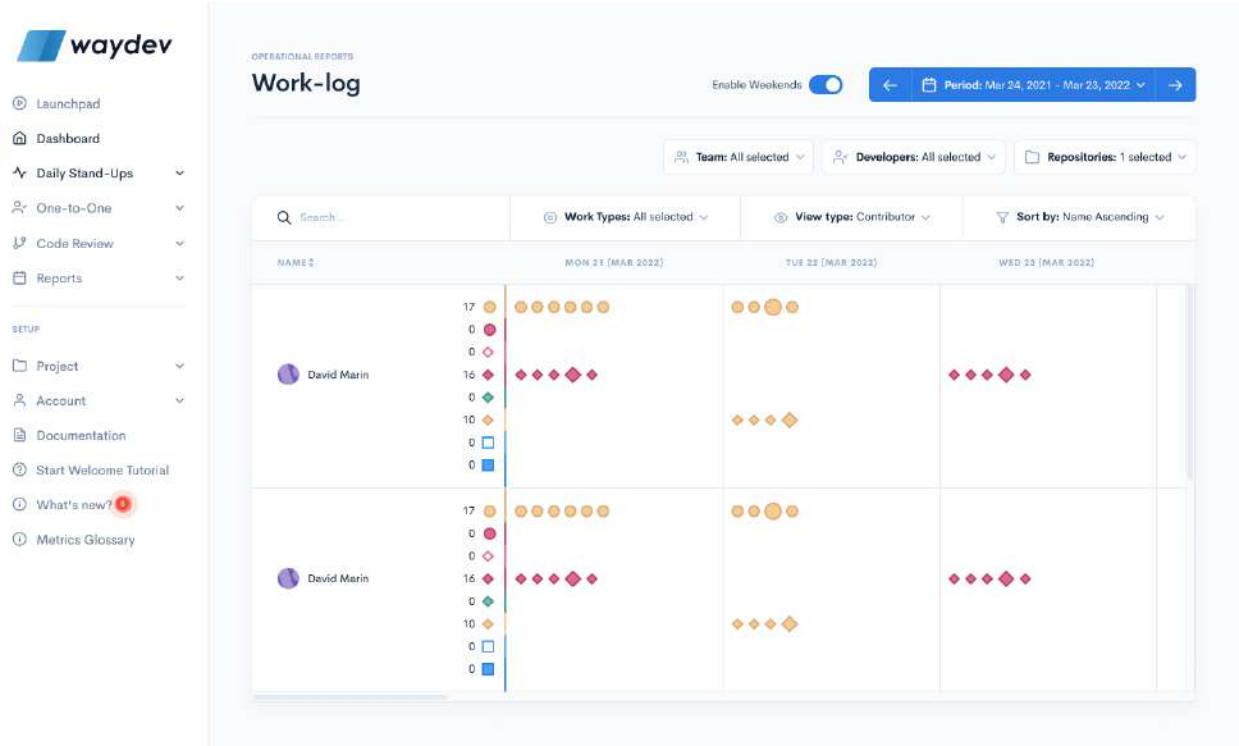
6. Active Days

It represents any day where an engineer contributed code to the software project.

7. Work Log

Work Log is one of Waydev's features that is relevant here as well, since it offers a comprehensive view of your engineers' contributions and work habits.

It brings in valuable data based on software metrics, such as code commits, merge commits, open/closed/merged PRs, PR comments. You can more easily notice when something might be off, when teams are backloading release, or what release might be at risk. At the same time, it helps you understand when to experiment with processes or find a healthier distribution of work.



8. Code churn

This code metric is a quantified representation of when engineers rewrite their own code too soon after it has been checked in (less than 3 weeks old). A certain amount of churn code can be expected for any developer.

But unusual churns should be discussed with your engineers: did something change? Were the specs unclear? These answers will let you know how to support your team.

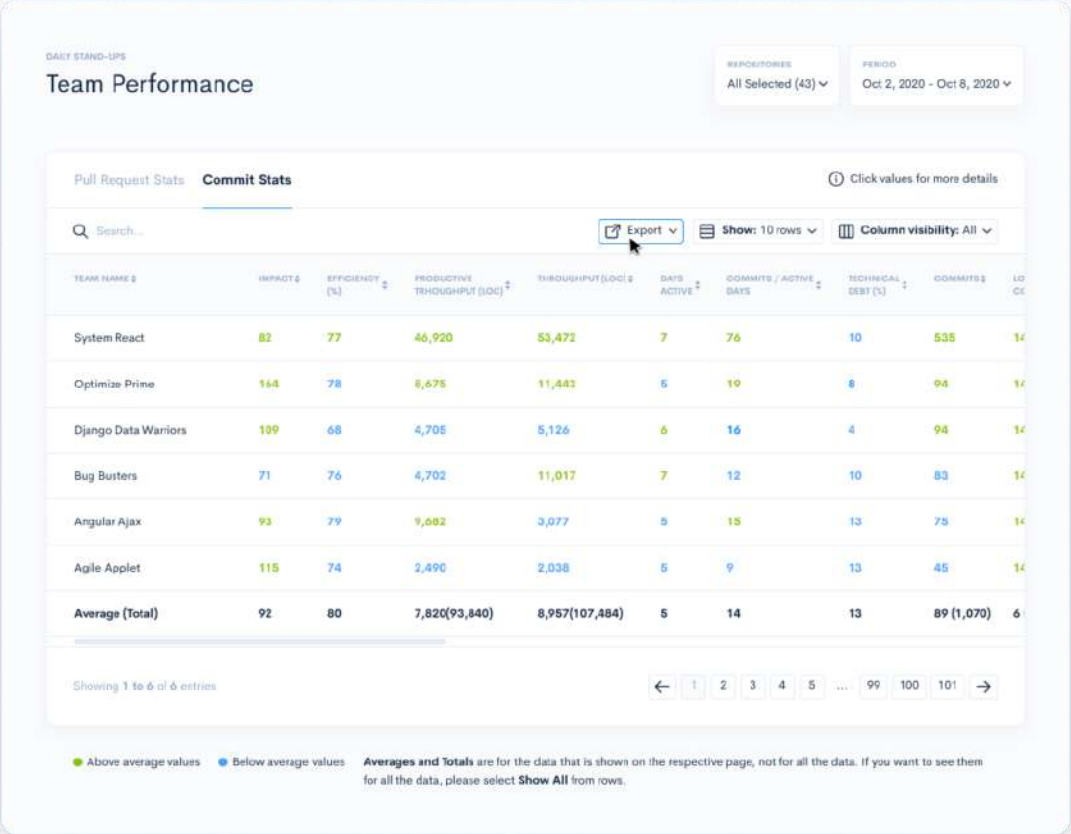
9. Productive Throughput

The proportion of code without churn.

10. Efficiency

This software development metric helps you speak towards the survivability of output into the product. It measures the percentage of a developer's contributed code that's productive, which involves balancing coding output against code longevity.

The Efficiency metric is independent of the amount of code written. The higher the efficiency rate, the longer that code is providing business value, and a high code churn rate reduces it.



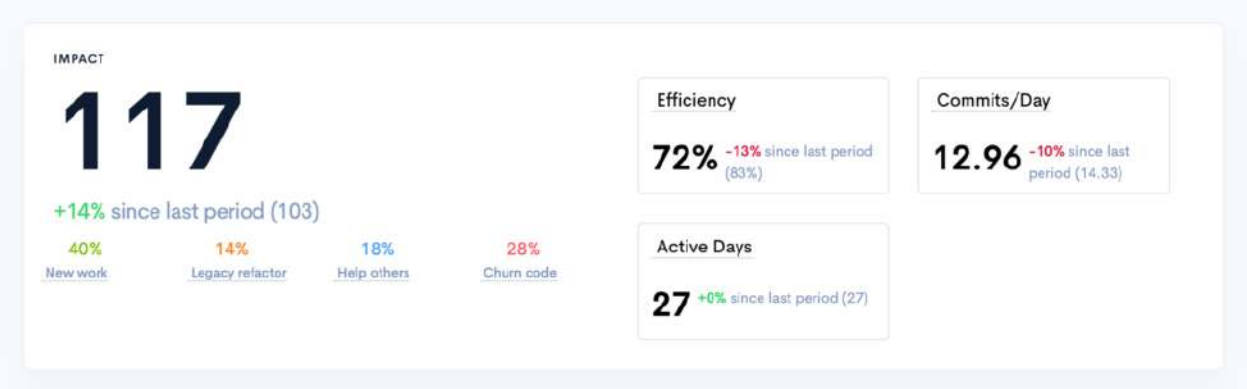
11. Impact

This software development metric speaks towards the complexity of output into the product. To be more precise, it measures the amplitude of code changes happening in a more complex way than measuring LoC.

Impact helps you find out an approximate value of how much cognitive load engineers carry when implementing changes. This set of software production metrics takes into account:

- How many edits to old code
- The surface area of the change (“number of edit locations”)
- The number of files affected
- The severity of changes when old code is changed
- How this change compares to others, in the project history.

Impact comprises data points that we, at Waydev, improve continuously on a monthly basis to provide a metric that translates engineers’ output into business value and cognitive load.



12. Mean Time Between Failures (MTBF)

This is an incident metrics that indicates the average time between repairable failures of a software or tech product. The longer the time between unexpected issues or outages, the more reliable the software produced.

13. Mean Time to Recover (MTTR)

It is another technical incident metric that refers to the average time it takes to recover from a product or system failure. Use it to calculate the time spent during an outage – from the moment it fails to the time it becomes operational. Also, it's usually relevant in assessing the stability of a team.

Security or Test Metrics

The following software metrics measure how thoroughly something was tested, which reflects in the software quality. Without a focus on quality, teams could miss delivery dates, high-risk work and code

rework, iteration interruptions, and frustrated clients, engineers and stakeholders.

14. Code Coverage

This type of metric determines the percentage of lines of code successfully validated under a test procedure. This, in turn, helps in determining how well a software is verified and if the code is readable and maintainable. Pair it with other metrics to fully understand how likely the software is to contain errors or undetected bugs.

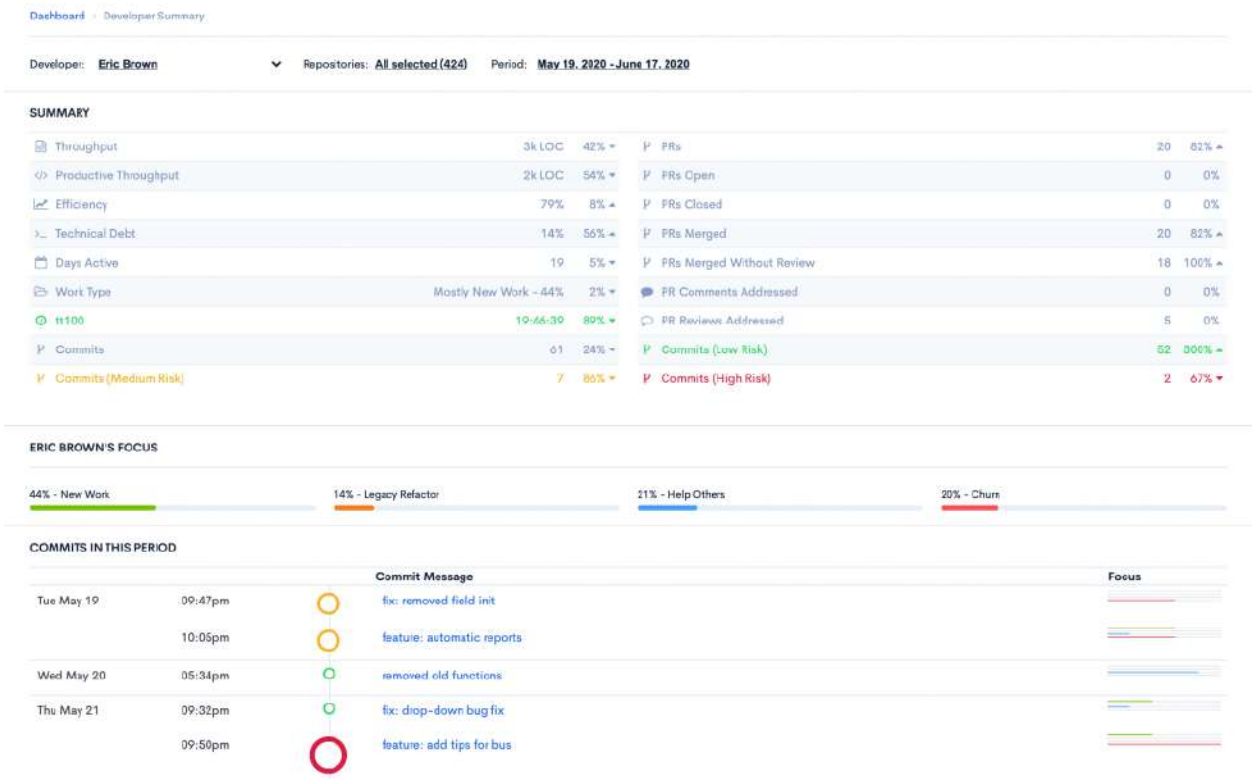
15. Commit Risk

The Risk metric measures how likely it is for a particular commit to cause issues. Clearing any risks means more quality, stable code for end-users. You can think of it as a pattern-matching engine, where Waydev is looking for anomalies.

This set of metrics looks at the following data points:

- The size of the commits
- The spread of the changes

- The depth of the changes



Use Dev Analytics to Track Your Process (and Help Your Team)

Software metrics provide an accurate analysis of the results and the impact of decisions made during software development. They're important to managers, teams, and clients.

By tracking relevant performance KPIs, you gain insight into your team's performance and progress in delivering reliable software, while meeting business goals and engineering OKRs.

As you start to measure these meaningful metrics, the first step for you as a leader is to educate your team on what you're doing and why it will help them get better.

A dev analytics tool becomes more valuable than ever.

Trusted by 300+ companies around the world, Waydev gives you clear visibility into your team's output and how engineers are working, while helping you make data-driven decisions.

About Us

Our mission is to provide engineering leaders with a way of measuring the performance of their engineering teams. We strive to help the technology industry move towards a data-driven agile development methodology and make decisions supported by data.

We are trusted by **Fortune 500 companies**, such as Blue Cross Blue Shield, TATA, and Carrier, and we are also loved by startups (#1 Product on Product Hunt).

Waydev is the **G2 Market Leader** in Winter and Spring 2022

Visit waydev.co to learn more

waydev

TRUSTED BY **FORTUNE 500** COMPANIES