DEV GRAPH

How to Align Coding Velocity with Software Quality

Why software teams need to prevent issues at the source by coaching individual developers to improve the quality of code they produce.

Every organization faces pressure to deliver software features to users quickly.

In fact, **97% of application development managers said IT is under more pressure than ever to deliver innovation faster**. The problem is that a higher development velocity—often with a sacrifice in code and software quality—can do more harm than good to the business.

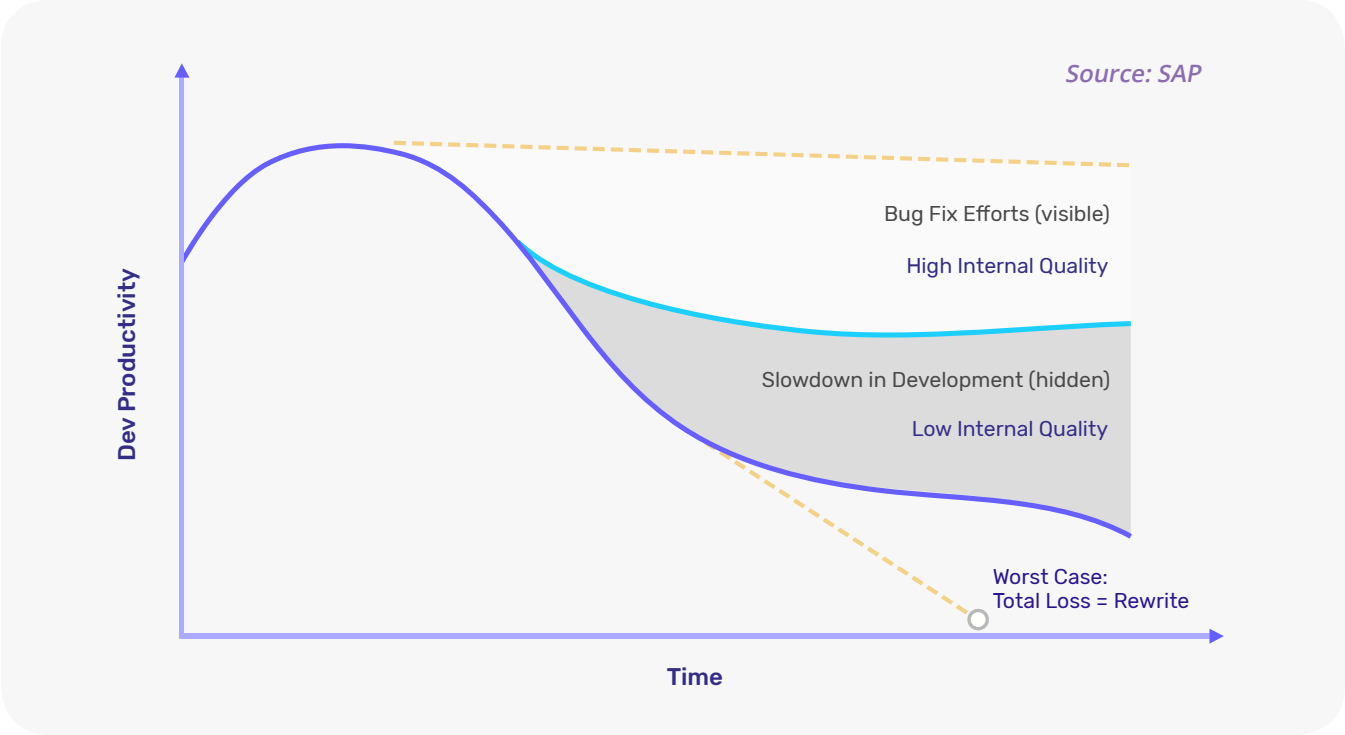
The impact of low-quality code is twofold: developers get frustrated when their code doesn't work correctly, and businesses risk ruining their reputation with customers if the software performs poorly. By overhauling the traditional approach to developer training, however, software teams can increase their development velocity and code quality at the same time. In fact, every developer can benefit from coaching and every coach should have data to back up their efforts. That's why it's in the best interest of technical leaders and developers to revamp the development process.

Errors are inevitable in software development, but the way that organizations identify, fix, and prevent them can have an enormous impact on software quality. While many software teams have processes in place to fix errors before releasing software, catching these issues in an earlier stage of development can lead to lasting and measurable improvements to developer productivity and code quality. That means changing developer behavior to eliminate coding deficiencies.

Using a data-driven approach to developer coaching, application development teams can accelerate coding processes, remove errors earlier, and improve software quality. Let's look at the advantages of stopping the flow of errors by identifying their source and fixing them through coaching. That way, developers and organizations alike can reap the benefits of better software quality.

Stopping the Flow of Errors

When it comes to developing high-quality software, identifying and fixing errors prior to its release is crucial. The problem is that automated testing, static analysis, and other methods for identifying issues offer little actionable insight for individual software team members. That means developers will continue to make the same mistakes, and as the codebase grows, the volume of errors to contend with could become unmanageable.

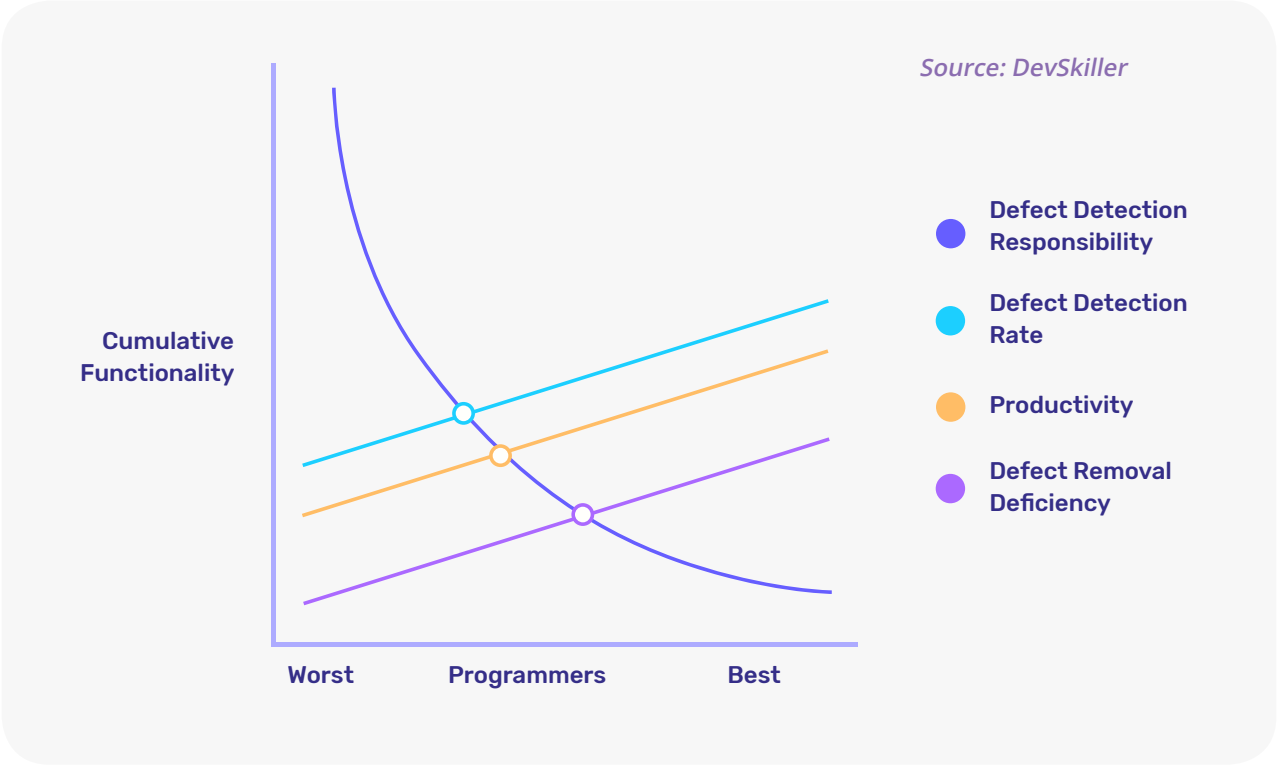


As the amount of source code that software teams have to deal with becomes unwieldy, developers usually will have to spend more time fixing bugs. That's because developers continue to make the same mistakes which accumulate over time, and the codebase becomes so complex that it's difficult to introduce new code without breaking something. When code quality deteriorates, developers will need to invest time in refactoring the code, or they could experience unpredictable slowdowns due to technical debt.

If errors are introduced at the same rate—without intervening to change developer behavior—then development velocity and coding quality could plummet. Unless businesses change their development process to identify and fix errors at the source, they'll always be forced to make a trade-off between error-prone code or slow software delivery. Software teams, therefore, need to give developers the resources they need to improve the quality of code that they produce by taking charge of their coaching.

Formalizing Developer Coaching

In the past, software teams relied on informal training methods over formalized coaching, but the reality is that there's always room for improvement. In fact, researchers found that the worst developers on any given development team could do more harm than good—a concept called a Net Negative Producing Programmer **(NNPD)**.



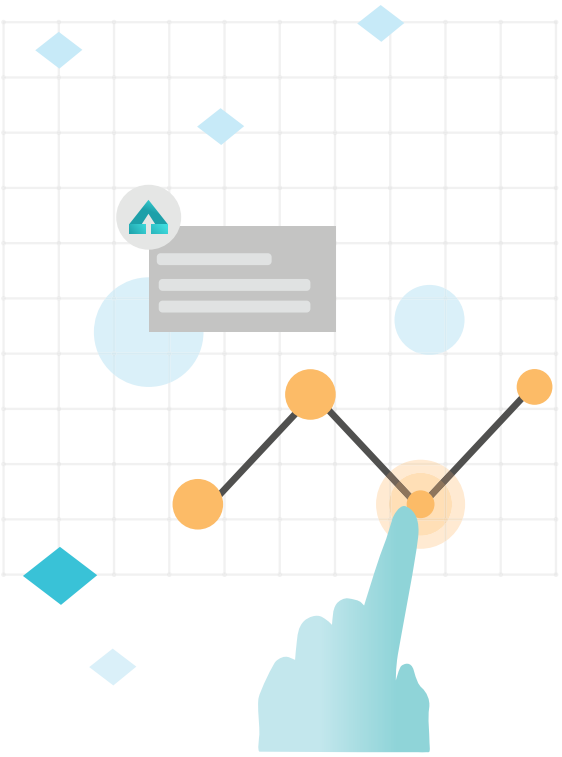
That's why nearly every software team could dramatically improve their efficiency by deliberately targeting poor performers and training them. Through coaching, developers can learn better habits that increase their productivity and help them deliver higher quality work. Moreover, management will recognize developers' progress and accomplishments, which could lead to career growth within the organization.

From the perspective of technical leaders, a formal coaching process is easier to manage and measure, so organizations can ensure they're making actual progress. By measuring key metrics and making them available to the entire software team, companies can also foster healthy competition between developers that leads to growth as well. That means everyone stands to benefit from a developer coaching process that identifies areas of weakness and sets goals for overcoming them.

Identifying The Source of Errors

Developer coaching is the key to balancing coding velocity and software quality, but this requires accurate data about individual developer behavior to be effective. Static code analysis tools may indicate common errors that development teams are committing, but these tools have a high false positive rate and fail to correctly attribute issues to specific developers.

Without a high degree of confidence in the data, the will to act on these insights is weak, and code quality never improves. Companies need data-driven coaching that can identify patterns in developer behavior that often lead to defects. Developers stand to benefit from improving their coding abilities, but they need to trust the data that management is using to drive their coaching is accurate.



How we can help

DEVCOACH

DevFactory's DevCoach is a personalized developer coaching platform that works.
The platform identifies relevant issues with a low false-positive rate because it refines the data it collects from a curated set of sensors such as static code analysis tools and DevFactory's CodeGraph.

DevCoach then attributes the issues to individual developers using a sophisticated attribution model. The platform offers insights into how to fix issues later by determining whether code commits are high quality (clean code), exacerbates existing issues (decay), or introduces entirely new issues (arson). That's because DevCoach uses deep forensic analysis to understand the erosion of code beyond the latest commit.

Once the issues are attributed to individuals, DevCoach uses this data to create developer behavioral models. The output from these personalized models can be used by developers, team managers, and organizational leaders to improve the quality of code produced. Developers have an accurate view of their own performance and can compare it with their team, which encourages self-coaching or adherence to formalized coaching by management.

Fixing Errors At The Source

Identifying areas of weakness is only half the battle. Raising the bar in terms of quality by rejecting defective code negatively impacts developer productivity if the number of errors coming down the line isn't reduced. Stopping errors at the source, therefore, requires continuous improvement because it doesn't matter how good developers are—but rather how good they can become. And the key to changing developer behavior for the better is a data-driven approach to coaching.

At first glance, developers may think coaching is unnecessary, but many developers likely won't coach themselves. Eliminating repetitive mistakes and producing higher quality code, however, can ease the stress for developers with pressure to meet deadlines. Accurate and relevant behavioral data encourages developers to change their behavior for the better and make the most of coaching opportunities.

How we can help

DEVCOACH

DevCoach takes data-driven developer coaching to a new level. Transparent and accurate statistics at several levels enable a top-down approach to improving code quality. At the organizational level, technical leaders can understand the performance of its various development teams and evaluate the quality of its code repositories to drive company-wide hiring or training decisions. Micro-level statistics for managers and individual developers allow software teams to discover weaknesses that can be overcome through consistent coaching. DevFactory and its customers have found, however, that less than 20% of developers will self-coach. That's why DevCoach recommends weekly 15-minute manager-led coaching sessions that target low performers. Armed with weekly and daily metrics, historical data, and positive or negative trend indications, developers and managers alike can find the motivation to improve their team's average code quality over time.

Since DevCoach separates issues into several categories, development teams have better data for choosing the best remedy. For example, clean code indicates a developer is performing well and could be a valuable mentor for others. Decay, on the other hand, suggests that technical debt is accumulating and refactoring should be encouraged before committing code. Finally, arson is the most harmful to a codebase and presents the greatest opportunity for developer coaching.

Targeting the lowest 25% of performance generally has the best results. In fact, at DevCoach, we believe arson—introducing entirely new errors—can be reduced by 25% in just a few months. Moreover, the platform tracks the propagation of code decay and other key metrics over a 13-week moving view to ensure behavior change takes place over time. DevCoach gives organizations the metrics they need to see results from their developer coaching efforts.

About DevCoach

DevCoach has been used by more than 120 enterprise software companies totalling 600 million lines of code under management to improve the abilities of thousands of developers around the world. For more information about how DevFactory can ensure business continuity, explore our **website**.