eBook

The ROI of AI for Code

diffblue
AI for Code

Let’s do the math

Artificial Intelligence (AI) for Code has huge potential to enhance software development and improve the value businesses get from different aspects of software engineering. AI for Code makes it possible to automate the creation of repetitive, basic code, leaving engineers free to focus on the high-level creative problem solving that advances the business.

The potential of this technology can be tough to express in terms of financial outcomes, leading business leaders to ask, “How do you prove the value of this technology?” and “How do I calculate its ROI?”

This eBook exists to substantiate the benefits of introducing AI into your development cycle, identify the challenges associated with the current landscape and demonstrate how you can quickly calculate your potential returns.

Here’s how to leverage AI to improve your ROI by increasing developer productivity and development cycle agility while mitigating risk and growing revenue.

The Challenge of Building Quality Code at Speed

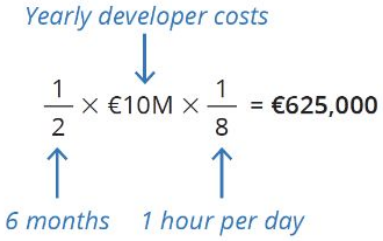
The ongoing digital transformation of businesses is calling for more in-house application development—and shorter, more agile development cycles—than even just a few years ago. These shorter cycles are not only difficult to compress but are also hard to keep on schedule and under cost.

One of the major contributing factors to software project delays is the discovery of bugs late in the development process. These bugs often crop up when teams are pressured to develop additional features in the project's original timeframe; when the time is short, testing suffers. The gradual build-up of testing neglect may have a knock-on effect on software quality across the board.

Diffblue Cover is a form of AI that can help bring deadlines back on track, both when writing new code and when maintaining an existing application. Diffblue Cover takes the code, analyzes it and generates unit tests. These unit tests provide a way for users to document the functionality of both new and existing code, which facilitates substantial improvements in software quality. Typically, IT is unable to make significant changes to legacy code because of the risk of regression bugs further down the line, which may impact their ability to upgrade, patch or add new features. By allowing IT to safely exploit legacy code and decrease time to market for new or iterated code, Diffblue Cover frees up important coder time for developing new features, instead of writing tests.

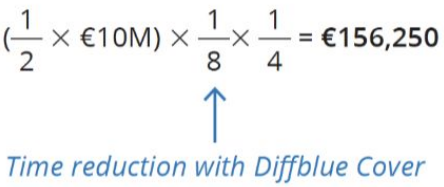
How Diffblue Cover Saves Valuable Developer Time

To illustrate the ROI of Diffblue Cover, we can outline a hypothetical scenario for a fictional financial services company that we will call BankCo. BankCo is building multiple new Java applications with multiple development teams, each comprising 100 developers in Europe, at a Total Employee Cost of €10M per annum. Each developer works an average of 8 hours per day in each 6-month project and spends a conservative 1 hour per day writing unit tests. This means that the true cost to the business of writing unit tests for each project is:



Let's say there are 8 development teams (projects); the total cost of writing tests is €5 million.

With Diffblue Cover, instead of spending one hour per day writing tests, each developer at BankCo might spend no longer than 15 minutes per day on this task.



Therefore, the relative savings in time is A –B = €468,750 across just one project. Across all 8, it approaches €4M.

The Best Solution for a Hidden Bug: Prevention

The value of Diffblue Cover doesn't stop at time savings, but also encompasses the harder-to-measure increase to the quality of the finished code. This can be illustrated with another hypothetical: BankCO plans a 6-month project broken into 4-week sprints, and at the end of each month the current prototype is passed on to QA. The tests for the software contain a small suite of unit tests and integration tests, but due to insufficient specification, most of the testing is done manually and coverage may not be ideal. The developers involved find that the time between introducing a bug into the code and discovering it is often longer than a month, which amplifies the project delays and cost overrun.

With Diffblue Cover, BankCo's developers no longer need to wait for the QA testers to give them feedback on the code. Instead, they receive a set of unit tests with every piece of new code. Therefore, when developers push changes to the code base, they can immediately see if their code breaks behavior by inspecting the generated tests. The significantly increased test coverage translates to improved software quality, which means IT will have a happier line of business customers. Of course, this scenario is not intended to be definitive; it is an illustration of how an organization can derive ROI from Diffblue Cover, and AI for Code, and it's an invitation to look deeper.

To try Diffblue Cover in your organization, **sign up for a free trial of Teams Edition**.