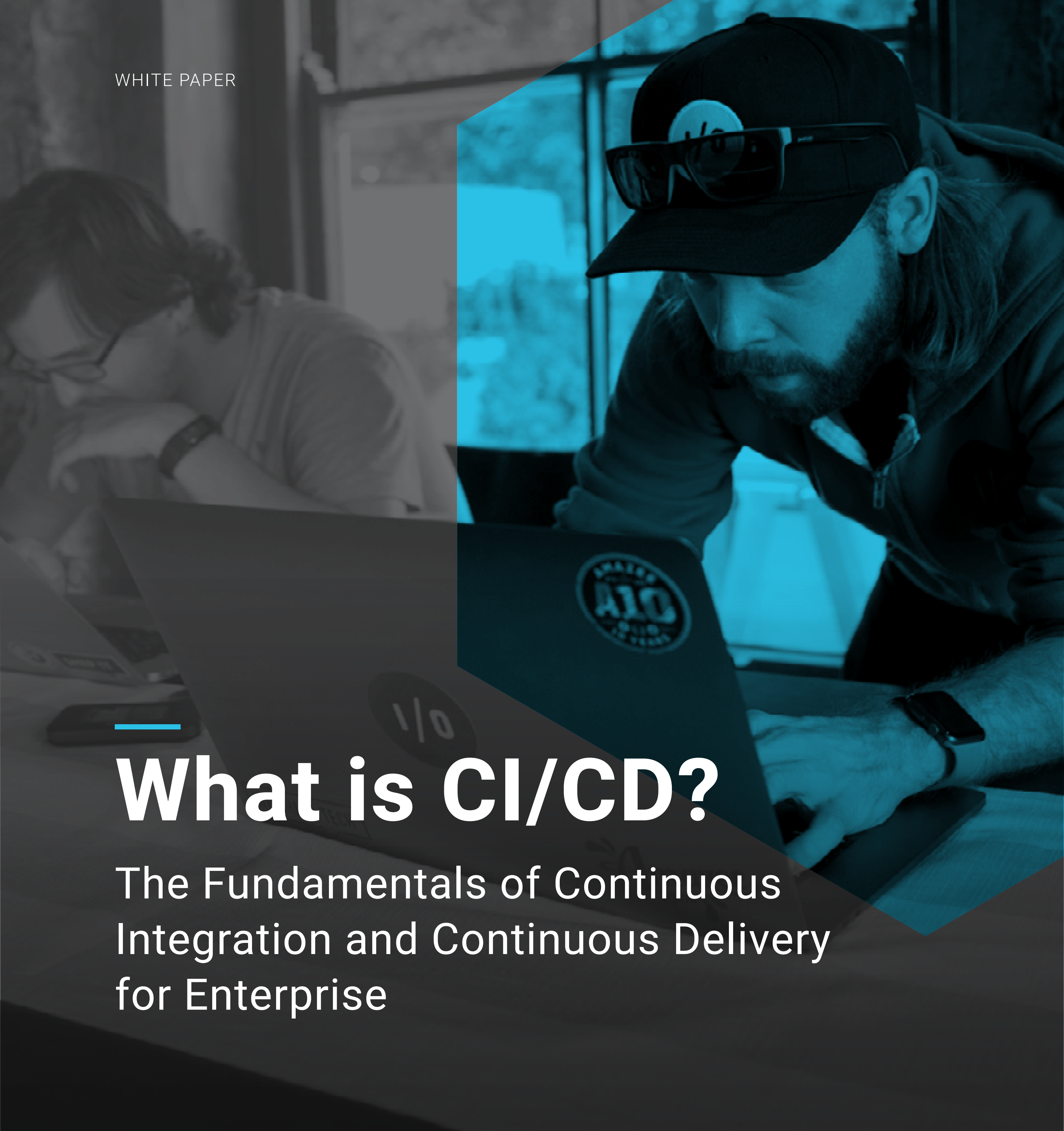
WHITE PAPER



What is CI/CD?

The Fundamentals of Continuous Integration and Continuous Delivery for Enterprise

amazee.io

What is CI/CD?

The Fundamentals of Continuous Integration and Continuous Delivery for Enterprise

The pressure is on. As a modern enterprise, you can't just deliver better software, web applications, or content experiences to your customers—you also need to be faster than the competition. Stumbling during the go-to-market race could have serious consequences for your bottom line. You need to deliver your product in the way that makes the most sense for your application, your team, and your business.

Today's software development life cycle means developers are juggling a litany of challenges: demand for higher quality software, shorter timelines, and an ever-changing technological landscape, among others. Neither you nor your developers want their time spent on repetitive or redundant tasks. Everyone wants to move the needle for the business, which is why many organizations are turning to a continuous delivery (CD) model to ensure fast, safe, and repeatable workflows.

Should your enterprise implement CI/CD? Let's take a closer look at the fundamentals...

What is Continuous Integration?

Continuous integration (CI) is an automated process that merges all the code changes your developers make into a shared repository. This allows multiple developers to contribute code without causing integration conflicts within a project. Automated testing is used to ensure the correctness of code before it is integrated into the shared repository. CI ensures that your developers are all "singing off the same sheet of music."

CI gives you back a particularly valuable resource: time

Without CI, if a developer wants to make a change to your application, they must download a copy of the current code base to work from. As individual developers make and submit changes to the source code, it gradually stops reflecting what exists in the shared repository. The existing code base can change considerably over time and new code may also be added, which can create dependencies, new libraries, and even conflicts.

The more time developers spend working on this code without merging it with the main shared repository, the more risk of failures and integration conflicts. When a developer is ready to submit their code to the shared repository, they'll need to spend time updating their code to ensure it's compatible with the changes in the repository since they pulled their copy. If the repository has changed significantly, developers will have to make large-scale edits to their work. In a worst case scenario (sometimes called "integration hell"), the repository has changed so much that the time it takes to integrate will exceed the time your developer spent making their original changes.

Knowing this, it's easy to see how CI can make developers more productive! It also empowers them to create software and applications in a way that is both scalable and customer-focused.

When you implement CI, you will...

- Ship fewer bugs to production.
- Resolve integration problems earlier.
- Free up more time for your QA team to make more meaningful improvements.
- Reduce costs associated with testing.

What is Continuous Delivery?

Continuous delivery (CD) works as an extension of CI with the goal of building, testing, and releasing software faster and more often. CD allows you to get the changes your customers want—requested features, bug fixes, new configurations—to them quickly, safely, and in a repeatable way.

After a build, code changes are automatically deployed to testing and/or production environments. Automated release processes mean that you can deploy at the click of a button. In CD, developers work in short cycles and can release on whatever schedule makes the most sense for your enterprise (daily, weekly, monthly, etc.). It is, however, recommended that you move to production ASAP to ensure that you're releasing smaller, more manageable batches because they are easier to troubleshoot if needed.

The heart of CD is continuous improvement. Enterprises that leverage CD have an advantage over their slower, less agile competitors because they are able to deliver applications faster and more reliably.

When you implement continuous delivery, you will…

- Reduce or remove the complexity around deployment.
- Be able to release more often.
- Give developers time back. Your team will no longer have to spend days preparing for release day.
- Improve customer satisfaction by speeding up the feedback loop.

The "Other" CD… What is Continuous Deployment?

Continuous deployment is similar to continuous delivery, with an important difference: continuous deployment does not require human intervention.

In continuous delivery, a developer manually clicks a button to deploy an application to your customers. With continuous deployment, all changes that pass through your production pipeline will be released to customers. Automated testing will validate if code changes are correct before deploying to a production environment.

The work your developers do can go live within minutes. This means that your team won't have to stop what they're doing during a release and your customers will immediately be able to give you feedback on the latest iteration. While the go-to-market benefits are obvious with continuous deployment, you will be giving up some manual control and there may be an engineering cost associated with maintaining the pipeline. The "other" CD is a powerful tool, but it's possible it may not be right for every organization.

When you implement continuous deployment, you will…

- Increase productivity. CD eliminates the need to pause development activities during releases.
- Have the ability to fix problems faster and reduce the risk associated with releases.
- Deliver continuous improvements to your customers and enhance the quality of your product every day.

What are the benefits of CI/CD?

CI/CD is not just about automation. It's about innovation and sharing what you've built in the most efficient and cost-effective way possible. It's simple: organizations that leverage CI/CD make better software, faster.

Still not convinced? Here's how CI/CD could give you the edge:

Faster release times. CI/CD can help your team remove bottlenecks. Accelerating your build, test, and deploy cycle allows developers to make corrections and push new features into production on a shorter timeline, which means your customers will be getting the features and bug fixes they want much faster.

Lower risk. The main goal of CI/CD is to make your releases as painless and boring as possible. By releasing early and often, you reduce the risk of bugs, downtime, and other adverse events.

Reduced costs. A CI/CD model can lower your costs through the elimination of fixed costs related to building, testing, and deployment. With testing, for example, your costs are reduced because your CI server is capable of running hundreds of tests in just seconds.

Better quality. The improved collaboration and automated testing enabled by CI/CD means that developers are able to spot and fix bugs early in development. By continuously delivering smaller updates more frequently, quality is always top-of-mind for your team and you are better positioned to make adjustments based on customer feedback as soon as you receive it.

Improved customer relationships. Your customers are getting the updates they want—and they like it. CI/CD lets customers feel like they're being heard, which translates to more positive interactions and brand loyalty.

Happier developers. Nobody wants to work on boring, repetitive tasks, least of all your developers. Working in a CI/CD pipeline increases efficiency and, ultimately, job satisfaction for your team as they're able to optimize workflows and build things that matter both to the organization and to themselves. A happy developer is a productive developer.

How can my enterprise accelerate release cycles?

Flexibility, transparency, security, and agility are important to your business, and only a fully open source WebOps platform can provide these benefits.

Built with Kubernetes and containers, amazee.io's WebOps platform can fully support any web technology and is designed to deliver the most efficient development, deployment, and end-user experiences possible.

Our commitment to the principles of CI/CD means that your developers can work in 100% congruent local environments, create new test environments on the fly, and maintain full control of their code in their own git repository. Plus, you'll always have the freedom to define the processes that work best for your business.

When you work with amazee.io, your developers will be fully equipped to build, run, and scale high-performing and secure sites and web applications, with full visibility into what they're running and how it's performing.

Speaking of visibility…
Want to see for yourself?

Book your demo today!

amazee.io