What does CI/CD success look like?

Common pitfalls and how to avoid them:

Adaptavist

Transform your CI/CD Pipeline

Common pitfalls and how to avoid them

Introduction:
A framework to facilitate DevOps

There's not one way to do continuous integration/continuous delivery (CI/CD). That means, as with any business strategy, there are textbook implementations that reap rewards for DevOps teams and the wider organisation – and there are those that fall a little flat. A winning strategy will be the difference between long-lasting benefits that set you apart from the competition and a half-hearted attempt that never gets off the ground.

Most notably, a strong implementation prioritises security from the start. The rise of DevSecOps – where security practices like automated vulnerability scans, role-based access, and secrets management are integrated earlier in the CI/CD pipeline – shows organisations are embracing a shift-left approach, where teams can catch issues well before deployment. This cuts the cost of late-stage fixes, speeds up development, and helps to build trust with your customers.

In this eBook, we'll help you to gain a clear idea of the challenges you're trying to solve so you can recognise success when you see it. Then we'll point out the obvious (and some less obvious) advantages you stand to gain with a good approach. Finally, we'll give you some top tips to help you measure what's working so you can fix what isn't. That way your CI/CD strategy is set-up to succeed from the start.

Part one:
Business challenges CI/CD solves

Key considerations for decision-makers

If you already have a CI/CD pipeline in place, delivering value and remaining competitive means making sure it's performing at its best. By helping business decision-makers understand just what CI/CD has to offer the organisation, you can ensure it remains at the heart of how you work. From a leadership perspective, there are three key considerations to prick up your ears when it comes to CI/CD implementation and proliferation:

- Reliability
- Simplicity
- Retainability

Sounds good, right? Let's take a closer look:

Reliability
Recovering from a failed build or being able to test quickly and confidently is a sign you're set up for CI/CD. That means daily commits to the main branch, automatically triggering build and test, and fixing the problem as quickly as possible. Having a CI/CD pipeline and best-practice automation improves reliability and reduces time spent grappling with branches and commits, meaning more attention can be paid to code development and adding features.

Simplicity
When it comes to building or amending your CI/CD pipeline, keep things simple. Stick to the essentials based on the current specific needs of your organisation, which will enable you to create a healthy pipeline, one step at time.

The ultimate aim is to shift from a manual process to an automated process for the whole of the CI/CD pipeline. This starts with code being committed and moved to build, on to testing, and then ending with deployment to production. This shift can be done incrementally if needed, keeping simplicity in mind, for example automating just the CI pipeline, before addressing the CD pipeline at a later time. You may also want to consider which of your applications to include in the process. Do you start with just one project that will allow you to test your strategy before moving other applications through the pipeline?

When selecting tools to automate the tasks in the CI/CD pipeline, remember to keep things simple as well - too many tools can be overcomplicated to implement, operate, and for your teams to use.

Retainability
Want a team that's attractive to other developers? Then CI/CD is for you. Developers prefer daily automated builds and established source code control processes (among other things). CI/CD pipelines with proper CI/CD practices let potential hires know your teams are high-functioning – making it much easier to attract the talent you need. It also makes it easier to retain your best people, as they can focus on solving customer problems, as opposed to fixing bugs that slip through the cracks.

Observability
Observability In modern CI/CD, observability is essential for monitoring and debugging pipelines. It lets you track metrics like deployment frequency, success rates, and mean time to recovery. Tools like Prometheus and Grafana provide deep insights into pipeline performance and health by collecting real-time metrics and visualising them so teams can identify bottlenecks or failures. While OpenTelemetry supports end-to-end tracing of requests across distributed systems. This lets you pinpoint exactly where and why a deployment issue occurred, significantly reducing debugging time.

How can CI/CD help?

CI/CD isn't just good for developers and ops employees – it works for the wider organisation too. From keeping costs under control to putting the needs of the business front and centre. Here are three key challenges you might be facing that CI/CD can help you fix.

Your IT budget is stretched thin

IT is expensive. And one of the biggest challenges you're probably facing is the fact that a huge percentage of your budget goes on human resources. Engineers are costly, especially if they're largely working on maintenance and undifferentiated integration to support a complex DevOps toolchain.

The more siloed teams with personal tool choices you have, the more maintenance is required (not to mention a lack of collaboration or visibility). That means fewer engineers working on money-making projects, less innovation, and slower growth for the organisation. By standardising tools across the organisation, everyone uses a common platform, practices and reporting processes, making IT teams' lives a lot easier.

Your dev and ops teams work against each other

The biggest difference between dev and ops teams is that the former are motivated to innovate, work fast, and ship new features, while the latter strive for stability, uptime, and accuracy. Because speed increases the likelihood of downtime and errors, traditionally these two departments are at loggerheads.

The result? By the time code makes it into production, it's already out of date. If problems occur and code is returned to a developer, the stop/start, slow nature of this workflow means it might no longer be deployable, which wastes time, resources, and is frustrating for the developer. And this is bad news for the business too – too much back and forth and not enough automation, means you often won't get to realise the work's value.

Your developers can't consider business needs

Without CI/CD, developers don't create code that's best for the business but become preoccupied with environmental dependencies and capabilities. Out of their comfort zone, they'll waste time worrying about ops issues, like infrastructure and configuration, to avoid downtime. That means less productivity, and less reliable and resilient software overall. Not to mention the fact that it will be harder to hire and retain developers who want to steer clear of roles with these responsibilities. A proper CI/CD pipeline encourages a rapid release cycle, making teams more responsive to fast-changing business environments and driving innovation.

Part Two:

What does successful CI/CD look like?

While it's important to consider the challenges CI/CD should help you solve, it's equally important to know what success looks like once you've addressed them. If you're committing code frequently, keeping CI pipelines simple and maintainable, and using tools that are compatible with your needs – particularly the latest developments in CI/CD technology – you're already halfway there. So, let's define success in relation to those three big challenges.

CI/CD success is… *Less maintenance, more innovation*

With engineers able to work on software instead of mountains of maintenance, innovation is able to thrive. Your budget is better spent on revenue-generating activities, driving business value. CI/CD infrastructure provides more flexibility and reliability. It can scale according to the teams and projects on your plate, rather than eating up your entire budget all the time.

Reliable tools that make all the difference

Continuous integration and continuous delivery are two sides of the same coin – so you might want an all-in solution that can help you achieve both. CI/CD shouldn't be another failure-prone thing you have to worry about. Lots of organisations are turning to reliable tools like GitLab for integrated CI/CD. This complete software development lifecycle tool comes with a wiki, issue-tracking, CI/CD pipeline and a Git repository manager.

Happy teams that see their code in action

With automated infrastructure and deployment, the business can benefit from new code right away. Developers get to see their code in production, which is satisfying and good for morale. Fast QA testing means a more productive workflow without long waiting times, so developers can move on to their next build efficiently. And because the chunks of code are smaller, risk is reduced.

Easy integrations for a seamless experience

When CI/CD is being done right, you'll have a big picture perspective of your DevOps lifecycle. With complex toolchains in play, integration, maintenance, and visibility can all be problematic, but a standardised toolset helps alleviate this. Tools that take care of multiple facets, such as Atlassian's Jira and Bitbucket are designed to work together, and GitHub Actions has gained popularity due to its seamless integration with GitHub repositories. Integrated toolchains like this are easy to maintain, and ensure you have a better idea of what's happening across the organisation.

Declarative infrastructure and Git-driven CI/CD

GitOps has emerged as a key approach for implementing continuous deployment, leveraging Git as the single source of truth for declarative infrastructure. This means infrastructure and app configurations are defined explicitly through code, ensuring your actual environment consistently matches your declared state.

And adoption is widespread. The CNCF 2023/2024 survey revealed that 60% of respondents had been using GitOps tools like GitHub Actions and Argo CD and practices for over a year, with 31% having onboarded GitOps in the previous 12 months. And it's easy to see why – GitOps supports faster deployments, easier rollback of changes, and improved auditability, so your teams can maintain consistent and reliable software releases.

Automatic, efficient workflows

By automating repetitive tasks – like deployment, scaling, and management of your containerised apps – and streamlining software development processes, you enjoy fewer dependencies and far better workflows. There are a number of automation tools such as GitLab AutoDevOps which automatically sets up the pipeline and integrations, or Jenkins Templating Engine (JTE) which enables pipeline templating so developers can create common workflows.

Part three:
How to measure your strategy's success

What improvements will we notice?

Effective CI/CD gives your organisation a competitive advantage, having long-term repercussions. But there are some significant changes you should start to notice straight away. By taking measurements before and after implementation, you can keep track of CI/CD's impact. This is a vital step you won't want to leave out as an IT leader – you will need to be able to demonstrate that CI/CD is beneficial for your team and the company.

With a successful strategy in place, software development should be faster and far less risky for your teams. Small changes can be made and committed more easily. And with quick turnaround on code feedback, innovation is able to flourish. Not to mention the competitive advantage your organisation will enjoy – not having to wait for manual code checks means you can start making revenue on new features straight away.

Some key DevOps Research and Assessment (DORA) metrics can be significantly improved with the help of CI/CD.

- Change failure rate – you'll notice the number of defects that make their way into product drop significantly, thanks to early automated testing.
- Time to restore service – automation also means fixes can be deployed to production in a jiffy, so your mean time to resolution (MTTR) should go down too.
- Deployment frequency – with automated tests and builds built into the deployment process, you'll be able to deploy much more frequently than before.
- Lead time – the overall time it takes to go from committing code to production will be much less. That's the beauty of early feedback and automation.
- Mean time to detect (MTTD) – the average time it takes for a team to identify a system failure or issue after it occurs. Using advanced observability tools, It's a good indication of how quickly your monitoring and alerting systems detect problems.
- Error budget – this is the permissible level of failure or downtime allowed while still meeting service-level objectives. This metric encourages teams to prioritise stability alongside experimentation and speed.

What are the wider business benefits of CI/CD?

Improving your pipeline is reason enough to embrace CI/CD, but there are a number of wider business benefits and operational upsides worth considering too. Let's take a look at the ones you're not going to want to miss out on:

Innovation acceleration

CI/CD speeds up your time to market, putting you head and shoulders above your less-automated competitors. With smaller code changes to consider, teams can iterate and deploy more quickly, setting the pace of innovation for the rest of your sector. And thanks to AI and ML integrations from tools like GitHub Copilot, assisting with code completion, automated reviews, and anomaly detection, teams can move with speed and confidence.

Whether you're launching a new product or making updates, CI/CD ensures thorough testing takes place without compromising on timeliness or quality. And if you need to make changes during development, it's easy to incorporate them, so your product improves and your users can reap the benefits right away.

Gain the clarity you need

Frequent testing helps developers have greater visibility, catching bugs before they become big problems. 'Failing fast' and 'shift left' aren't dirty phrases – they should be celebrated. They're what make CI/CD so effective. When features fail, you get immediate feedback and can correct things quickly. You can conduct a whole host of tests to get a clearer picture of what's working and what isn't prior to a full release. And if hiccups happen, it's easy to roll things back without your users being seriously inconvenienced.

Say goodbye to silos

From developers to testers, product managers to ops admins, your teams will be more collaborative across the board. Breaking down silos is not just great for collaboration and communication, it ensures a sense of shared responsibility where everyone is working towards the same organisational goals. And it's a clear sign your business is on the path towards digital transformation, embracing an agile approach to innovation.

See the bigger picture

CI/CD tools also offer your organisation a powerful data stream to help you steer the ship. You can generate, track, and report on vital metrics across the life cycle, including greater transparency and analysis of the production environment. Having all the information at your fingertips helps you to make important strategic decisions about infrastructure, team performance, product roadmaps, and more.

Observe, monitor, and improve

Large, widely distributed teams and development environments mean it can be hard to track down the source of problems and fix them fast. CI/CD builds observability and monitoring into the pipeline, helping you keep track of your software's performance in line with code merges. That way, when something breaks, you can easily figure out what and where the cause is.

This is further improved by the fact code is managed and tested in smaller batches – ensuring only that which meets certain standards makes it through. Incident management tools like PagerDuty and Opsgenie can be key here. Integrated into your pipeline, they automate alerts and can handle incidents promptly and efficiently, routing critical issues to the right team members. This reduces response times and ensures only high-quality, stable code makes it to production.

Deliver safely and with confidence

CI/CD pipelines enhance security by integrating it into the development life cycle and automating critical processes. This ensures faster, safer releases and proactive risk management that's both efficient and scalable. It includes:

- Automated vulnerability scanning to identify risks in code, dependencies, and containers early, preventing security issues from reaching production.
- Secrets management and secure configurations to safeguard sensitive data, ensuring they're not exposed in code repositories.
- Compliance checks and automated policy enforcement to verify adherence to security standards and regulations at every stage, reducing manual errors

Really impact revenue

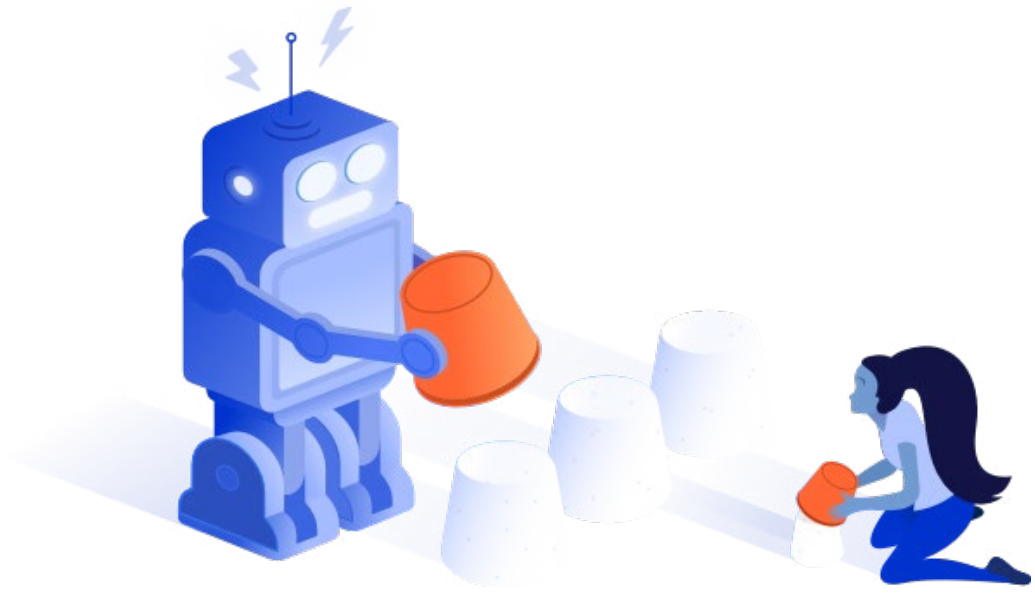
CI/CD helps you make the most of your resources, ensuring cost-efficiency and allowing your developers to focus on driving innovation and revenue-generating projects. Not to mention the fact that CI/CD-enabled organisations are making revenue on the features they deploy, rather than having to wait for manual checks to take place first. Automated tests help developers know that code is up to standard, so they can worry about more important things instead – all the better for your bottom line.

Happy users

Better quality software is nothing to be sniffed at. As your developers are generating smaller sections of code, they'll find problems right away and resolve them just as efficiently. With fewer bugs making it into production, your users enjoy a better experience overall. Customer satisfaction is king. Buggy software might see your users jumping ship for the competition, so CI/CD can be a life-saver here.

Happy teams

But it's not just the users you should take a look at. Organisations with a successful CI/CD strategy have happier teams too. When your people can spend less time putting out fires and more time focusing on adding value, they will be productive and stay put. Turnover is disruptive for your teams – and it costs a lot too. With both dev and ops working to their strengths, word will get around. Soon your cooperative engineering culture will help you attract the right talent and keep it for longer.



How can I measure if my CI/CD strategy is working?

The wider business benefits mentioned above will become obvious over time, but what real measurements should you be taking to check that your CI/CD approach is on track? Ultimately, you need to know that you're cutting the risk and cost of IT failure with higher quality code. KPIs to check your pipeline's effectiveness include deployment frequency, deployment failure rate, change volume, meeting availability, MTTR, defect volume, and escape rate.

There are lots of options to consider – and only you will know what makes sense for your business. But here are four metrics to measure which benefits are already beginning to be felt and where you can still make improvements, helping you stay focused on your strategy.

1. Cycle time
This is measuring the time it takes your developers to deliver functioning software – from the moment they start working on it to the moment it starts creating value for the end user.

2. Time to value
Developers might be creating code quickly, but does it take a long time to release it? Time to value measures the size of this bottleneck – the delay that starts after code has been written to when it's up and running in production.

3. Uptime
Ops teams are obsessed with uptime, and for good reason. Effective CI/CD strategy maximises automation and high availability so they can focus on keeping things up and running. Other ops goals, like error rates and infrastructure costs, are worth measuring too.

4. Retention
Sure you could count the smiles on those Zoom calls, but retention rates are a more reliable way of figuring out if your strategy is working for the people who put it into practice. If developers aren't happy, they might not raise their hand, but if they don't stick around for long, there's usually a good reason.

Summary:

Business challenges CI/CD solves

CI/CD is an intrinsic part of making your DevOps approach a reality. But you can't outsource your strategy and hope for the best. Your organisation is unique, and your CI/CD pipeline has to work for your people, processes, and wider goals.

The benefits of getting your ducks in a row are plentiful – there's lots to be gained from persevering and finding an approach that works wonders for your teams. Now you know the challenges you're trying to solve, the sure signs of success to look out for, and what metrics to take, you can start implementing (or refocus) CI/CD with confidence.

And if you need a bit more support, then we're here to help. At Adaptavist, our DevOps solutions get you ahead of the game. With our experienced and dedicated team of experts, we combine the right mix of strategic-led consultancy and technology-led solutions that place people, process, and tools at the heart of your business strategy.

To start shaking up your CI/CD strategy, get in touch.

contact us

Adaptavist