Mastering digital transformation: the power of DevOps practices & culture to supercharge your software delivery

Adaptavist

Mastering digital transformation:
the power of DevOps practices & culture to supercharge your software delivery

Contents

Introduction:
Prepare to perform...

Today, businesses are up against it in more ways than one. We're living through ongoing uncertainty, continuous challenges, constant technological evolution, and remote working – all constantly challenging businesses to step up to survive. And digital transformation is one of the life rafts most organisations are desperate to jump into.

Why? Well, beyond basic survival in the marketplace, it ensures you're able to adapt quickly to supply chain disruptions and the time pressures of getting new products and features out to the market, as well as respond to customers' rapidly changing expectations and requirements for flexibility and personalisation.

The challenges of transformation

Unfortunately, digital transformation is not inevitable or easy. The need to optimise technology to develop and implement new business models is a challenge unto itself. It requires businesses to keep revisiting their culture and practices to develop beneficial new ways of thinking and working. And all this without being so disruptive that it throws you (and your customers) completely off course.

As you transform, you'll undoubtedly butt heads with a number of related and interconnected issues – from how to onboard widely distributed teams and educate them about cyberthreats to how to scale your software development processes by embracing automation. Here are just some of the questions you're probably already asking:

- How do we innovate meaningfully?
- Are IT teams prepared for today and for the future?
- How do we attract the right talent – and then keep it?
- How do we modernise at speed?
- Are we meeting our regulatory requirements?
- How about customers' needs?
- Is our software security up to scratch?
- Are we making the most of data insights?
- Are we measuring our success?
- Does our working environment work for everyone?

As these questions suggest, the potential roadblocks to transformation are numerous. Maybe budget constraints are causing delays in getting automation projects off the ground, putting manual strain on your already-busy teams. Or perhaps your teams are getting complacent? Frustrated by frequent technology changes, they could be apathetic to the latest tools and techniques. This will put you on the back foot when it comes to keeping up with the competition. There are also legacy considerations – with outdated infrastructure making modernisation much harder and the complexity involved when trying to integrate new technologies.

Get to the good stuff

Fortunately, the urgency to transform also opens up huge opportunities for your organisation. These include:

- Becoming the place where the best developers want to work because of the exceptional developer experience on offer.
- Encouraging practices that empower your people to work collaboratively and autonomously.
- Delivering more value for your customers through a powerful feedback loop, continuous monitoring, and an agile culture that prioritises innovation.
- Ensuring your software is high-quality, reliable, and secure thanks to DevOps, DevSecOps and platform engineering strategies.

It's likely you're already using some of these practices as part of your digital transformation strategy. But are you maximising the opportunity on offer? It's easy to implement and ignore, feeling like you've ticked a technology or practice off the list and that it will then take care of itself. But to truly enhance the software delivery performance of your teams, you need to take stock, tune in, and revisit your culture and working practices.

In this ebook we take a closer look at the ways in which optimising your efforts can give you the results you want and the edge you need. Keep reading for practical advice from the experts to help accelerate your transformation.

Part one: Collaboration and communication

Ah, collaboration! A word that gets tossed around so much in the context of agile working and DevOps practices that you would be forgiven for assuming that a collaborative culture exists simply by 'doing DevOps'.

But collaboration is not merely a group of individuals or teams who happen to be working on the same project or following some of the same processes. It comes when people work together as a unified group, to achieve a common vision and set of goals. It's when siloed working has been eradicated and competing interests put aside in pursuit of aligned objectives.

DevOps discussion is heavily weighted towards the processes and tools that can support a collaborative culture and make communication easier – all in the service of building software faster with fewer errors. But all the shiny new tech in the world won't do you any good if you haven't achieved that culture in the first place.

So let's start there...

Understanding the cultural shift

Where teams have traditionally worked in silos, each responsible for their own small part in the machine, DevOps is a huge departure. Not only in terms of the speed and agility involved (see part two), with a dependency on automation, but also the collaboration required to achieve it. With this collaboration comes a shared responsibility, whereby a developer oversees a product through the entire course of its lifetime.

This collaboration and communication mean DevOps teams become autonomous, trusted to make decisions and changes without a long-winded approval process. They rely on their own knowledge and lean on self-service tooling to get the job done, without a fear of failure. And they work with operations folk to create monitoring and reporting strategies that deliver fast feedback. This lets them iterate quickly and improve their code.

Next, let's look at a few actions you can take to weave more collaboration and communication into your culture.

How to create a more collaborative, communicative culture

Build cross-functional teams

DevOps teams are involved with every step of your software development life cycle, which means that each team member needs a broad range of knowledge and skills. In reality, it's unlikely everyone will have the competency level required to fulfil every task. It's more important that there's a basic level of knowledge across the board, complementary areas of expertise are covered, and teams are encouraged to share responsibility rather than individuals working in isolation.

How do you know if your teams are cross-functional? You'll need to carry out a competency assessment to understand what skill and knowledge areas are covered, and where the team's shortcomings lie.

With the assessment complete, you'll have the information you need, to know where to upskill (by offering team members the chance to develop in the areas they need to), to bring in new talent, or move people between teams to balance out competencies across the business.

Expert advice:

The DevOps Agile Skills Association's competence model identifies and defines four skill areas and eight knowledge areas required for a high-performing DevOps team. They are:

Skill areas

- Courage
- Team building
- DevOps leadership
- Continuous improvement

Knowledge areas

- Business value optimisation
- Business analysis
- Architecture and design
- Programming
- Continuous delivery
- Test specification
- Infrastructure engineering
- Security, risk and compliance

Surveys and interviews are the most simple ways to determine a team's strengths and weaknesses in these areas. But, at a glance, you might already have a clear idea if your team (or teams) has all these competencies ticked off, and where some training might be required.

Improve your developer experience

A good developer experience, or DevEx, is one where developers have access to all the information they need and can switch between focused work and collaborating with the rest of the team, helping them complete tasks quickly and efficiently. When it comes to DevEx, collaboration is king.

It's enabled by visibility and transparency. Think of the former as the availability of knowledge and the latter as the ability to share knowledge openly and proactively – culture plays a big role here. Improving collaboration means providing teams with the tools and frameworks that allow both visibility and transparency to occur without constantly dragging people into unnecessary meetings.

Expert advice:

If you want a better understanding of where your DevEx is at, you're going to need to take a good, hard look at it and then action any insights to make improvements. This might involve:

- **Surveys** – ask your developers about everything, from how satisfied they feel at work to the tools they use and code quality.
- **Interviews** – get more depth about challenges your people are facing, where bottlenecks lie, and their ideas for improvements. Exit interviews can be helpful too.
- **Metrics** – poor retention rates might be a sign your DevEx isn't working, productivity metrics will give you some idea of how efficient your workflow is, and onboarding time clues you in to how good your documentation is and how supportive the team are to new starters.
- **Tool tracking** – get to know how much developers use the tools on offer and which features are hardly used.

Share knowledge

Collaboration doesn't happen without communication. When information is siloed by individuals or teams, efficient DevOps cannot occur. Developer self-service and an internal developer platform (IDP) enables everyone to interact using one centralised hub. With chat channels, discussion forums, and knowledge bases built in, they make it much easier for people to understand each other, keeping conflict at bay.

Expert advice:

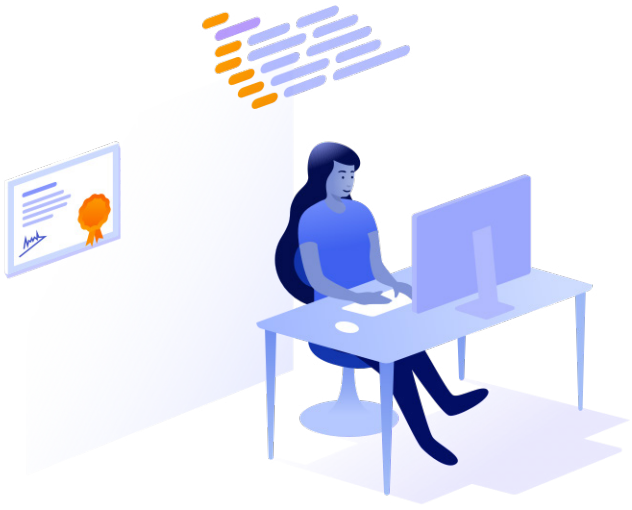
With access to information, cross-functional teams can evolve and basic competency across all areas can improve. But what should your IDP include? For starters, this curated catalogue should have all the resources your people need to succeed, from code templates and software libraries to APIs, best practice guidelines, and troubleshooting tips.

From the top

This cultural shift can't be a grassroots effort – it has to come from the top and continue to be led as adoption spreads. That means executive sponsorship of any new tools or process changes, and leading by example when it comes to collaborating with others and effective communication.

Expert advice:

DevOps leaders don't have to just be in the C-suite. Make sure each DevOps team has a cultural change ambassador who is knowledgeable and passionate about DevOps processes and tools, and encourages others to embrace this way of working.



Part two:
Efficiency and agility

Collaboration is a key cultural touchstone of DevOps, but it's not the only one. Efficient, navigable processes and an agile approach to software development are also essential if you want to achieve streamlined, frequent, high-quality software releases. The benefit of which not only increases your organisation's productivity and performance but also helps ensure employee satisfaction.

So what does an efficient and agile culture look like in the context of DevOps?

Feel free to fail

Failure is an option – in fact it's imperative. To create a culture of learning and improvement, you have to embrace and accept that failure will happen. Teams should feel safe and empowered to try new approaches without fear of failing. When failures inevitably occur, it's important to turn these into learning opportunities.

Value fast feedback

Feedback enables continuous improvement. Rather than feedback about the performance and stability of application software never making it back to developers, in a DevOps culture, that feedback is immediate. Thanks to automation and continuous integration, new code is quickly built and tested, with developers receiving immediate feedback about its quality so they can rapidly iterate to improve it.

Align around shared goals

Practising value stream management (VSM) – where you visualise, measure, and optimise the flow of value through your software delivery life cycle – helps align development and operations around a shared goal: delivering value to end users. It encourages transparency and data-driven decision-making, leverages metrics, integrates automation and continuous feedback into the pipeline, and empowers teams to prioritise work that delivers the most impact.

If you can, automate it

If you're going to achieve continuous improvement, with the ability to respond immediately to customer feedback, you're going to need to embrace automation. From infrastructure provisioning and deployment pipelines to routine testing, there are a number of tasks that can benefit from streamlined automation tools, freeing up your resources.

How to be more efficient

With efficiency and agility embedded into your culture, there are a number of key strategies and best practices you'll want to employ to put that thinking into practice.

Agile project manangement

Agile methodologies can help DevOps teams to be more adaptable and respond to change quickly. They encourage constant communication and collaboration, helping you break large-scale projects down into smaller chunks so you can deliver work in increments, instead of building up to a big release date.

Keep evaluating your requirements and plans, listen to feedback, and make changes as required. Embrace a core framework like Scrum or Kanban to plan, track, and measure this work. Scrum, for example, is great for iterative development with structure sprint cycles, while Kanban gives you visibility of work-in-progress so you can keep on top of unpredictable workflows.

CI/CD

By shifting left and bringing testing into your code development processes much earlier, developers can fix bugs and improve code quality as they go, rather than sending multiple changes to a separate testing team. To do this you need to be practising continuous integration and continuous delivery (CI/CD).

With CI, code from multiple contributors is automatically built, tested, and checked to make sure it's compatible with your existing codebase, helping you detect issues early on and reducing any conflicts. CD, meanwhile, allows for the rapid and reliable release of software updates that adhere to your timeline for a smooth delivery cycle. With automated testing and quality assurance practices, it means that the main branch of your repository is always deployable.

Microservices architecture

With a microservices architecture – where applications are broken down into smaller services – each service is focused on a specific business capability, meaning they can be developed and deployed independently of each other. Using containerisation and orchestration, you can improve agility and scalability, release features faster, and spend less time on maintenance and updates.

Automation

CI/CD is not the only automation you need to consider. Automated testing, including end-to-end, integration, and performance testing, is key to doing DevOps successfully. Tools like GitLab, Jenkins, GitHub Actions, and Argo CD are popular choices for CI/CD and Kubernetes deployment. They help to streamline these processes, reducing the risk of errors and speeding up processes.

Infrastructure as Code (IaC) tools can automate the creation and management of infrastructure, including your servers, networks, and storage, while configuration management tools automate the configuration of these components. They ensure servers are configured consistently and that when changes are made, they're repeatable and controlled.

Continuous monitoring

Regular tracking, using metrics and continuous monitoring dashboards, can help you identify and rectify issues early and mitigate against unplanned work. Broken builds or failed tests can cause unnecessary delays and inefficiency. But these tools can clue you in to system performance, customer feedback, and incident reports, as well as security issues (see part five). This practice is a key part of agile working, helping you to continuously improve your processes, so your teams can work smarter and release faster.

Part three: Empowerment and autonomy

As we explained in part one, DevEx is at the heart of a developer-centric culture, and constantly improving it is essential if you want to enhance software delivery efficiency, agility, and effectiveness. Alongside a collaborative, transparent, and agile culture, empowerment and autonomy have become synonymous with a good developer experience, but how can you cultivate and promote these?

The impact of positive developer experiences

DevEx can directly affect the success of your software development process. According to a 2024 study by Microsoft Azure, developers who had a significant amount of time carved out for deep work felt 50% more productive compared to those without it. And of the developers who found their work engaging, 30% felt more productive. So, how can you create a positive impact on your DevEx?

Streamlining workflows

Many of the DevOps practices we've already covered go some way to creating a good developer experience, including providing developers with greater visibility so they can work quickly; effective collaboration tools for fast communication and knowledge sharing; automation to find and fix bugs earlier and manage code more effectively, and fast feedback loops to enable agile working.

Beyond that, you should consider streamlining your workflows using source code management. This allows developers to know what needs to change as soon as they get started, reducing friction. They should also have easy access to operational insights, so they can understand and prioritise issues and work more collaboratively with operations teams.

Almost all of these practices are made far easier by self-service and an IDP, empowering developers to work autonomously with all the tools and information they need at their fingertips.

The role of self-service platforms

Self-service, supported by an Internal Developer Platform (IDP), is another key instrument for enabling developers to work autonomously. In any domain, self-service done poorly leads to confusion, dissatisfaction, and a slower, more painful experience. But done right, it can speed things up, remove friction, reduce bottlenecks, and increase productivity. And that's what you should be aiming for. The quicker work gets done, the quicker projects get over the line and the quicker you can deliver value to customers.

With comprehensive, accessible documentation – like source code libraries and service catalogues – all in an easy-to-search format, developers can focus on what they do best and avoid wasting time worrying about documentation and infrastructure.

How to reduce friction

If a friction-free environment for your developers is the goal, then what steps can you take to get there?

1. Start by conducting regular audits, evaluating your existing tools and processes to identify inefficiencies and bottlenecks. Your tools should enable collaboration, speed up processes, and have strong integration capabilities – so implement and upgrade where necessary.
2. Alongside routine task automation and AI-powered feedback loops, if you don't already have clear, well-documented coding practices, that's something you're going to need to address. These can improve code readability and make it easier to maintain. Automated code reviews are a good place to start to get your guidelines up to scratch.
3. With friction identified, you can simplify, accelerate, and optimise all your tools, systems, and processes. But monitor the situation to make sure your DevEx efforts are making improvements and not just adding more friction. Cutting down on the number of meetings, for example, might seem to reduce context switching, but it could impact collaboration. Implementing a DevEx team tasked with understanding, improving, and tracking these initiatives is ideal.

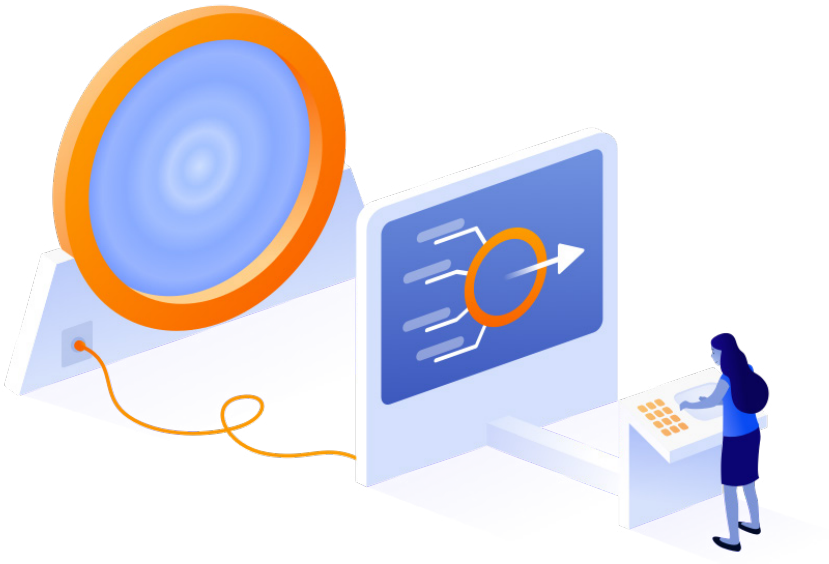
Are IPDs the answer?

According to the Forrester Snapshot, 43 percent of respondents said IDPs enable true developer self-service, helping them to become less reliant on ops.

Everything they need, all in one place

As cloud architectures become more complex, ops tickets cause inevitable bottlenecks – a huge pain point for developers. But IDPs provide a curated set of processes and tools that are easy for developers to digest independently, helping them to become less reliant on ops teams and improving their velocity. This, in turn, reduces time to market and helps grow revenue, all while having a huge impact on developer satisfaction.

Having an IDP also opens the doors for another key component of good DevEx – continuous learning opportunities. With information available and accessible, developers can grow their knowledge base and learn from each other's expertise.



Part four:
Quality and reliability

DevOps, DevEx, and platform engineering don't just speed things up with no concern for the outcome. They maximise developer productivity and emphasise continuous testing and monitoring, helping ensure high-quality, reliable code and contributing to the overall performance of your software delivery.

We've already talked about how successful DevOps demands a collaborative, agile culture, and the steps to take to make your DevOps efforts more efficient. But what strategies do you need to have in place to get your code up to scratch so your developers and customers can count on it?

Continuous delivery

DevOps gets everyone on the same page, fostering collaboration and alignment and making it much easier to create high-quality products. With CD pipelines, you can rapidly add new features, manage configuration, and address bugs, without compromising on quality and reliability. These pipelines also ensure you can stick to your timeline, while keeping manual intervention to a minimum. And with automated testing and quality assurance practices (see below), CD pipelines ensure smoother, error-free deployments.

Progressive delivery strategies, like canary releases and feature flags, have become essential for ensuring reliable rollouts and mitigating risks. The former allows gradual deployment to a subset of users so that teams can monitor performance and catch issues early. While feature flags offer precise control, letting you toggle features on or off in real time, reducing the need for rollbacks.

Quality assurance

An effective DevOps strategy also takes QA into consideration, eliminating miscommunication so QA understands developmental requirements and vice versa. QA plays a vital role in product quality, making sure the development team is delivering the operational team's technical requirements and that the product aligns with business goals.

QA brings a different, wider perspective to product builds – one that actually reflects user behaviour rather than the more idealised view sometimes held by developers. With this extra set of eyes in place, developers can focus more on new feature innovations, rather than visiting previous commits.

Testing is where QA really shows its value. When it's working well, QA enriches the testing process and the product by testing new iterations in parallel and in real-time. This not only eases the product pipeline but helps mitigate failures and validate your DevOps testing practices.

Infrastructure as Code

Without IaC, your teams will have to individually maintain your deployment environment settings, leading to inconsistency and issues. Adopting IaC automates your IT system, ensuring your infrastructure remains in its intended state. With an effective strategy in place, management of any deviations from this intended state is automated, reducing the risk of manual omittance or error.

It also allows you to reliably deliver stable test environments rapidly and at scale – vital for testing applications in production-like environments early on in the development cycle.

DevSecOps

By integrating security practices throughout your DevOps software development life cycle, you can develop much higher-quality products, free from compliance issues. By shifting security left, you can detect and prevent vulnerabilities much earlier, improving the overall quality of your software and ensuring security is top of mind for all your teams. We talk more about the importance of a DevSecOps approach in part five.

The power of the platform

Platform engineering helps you align development practices with business priorities. With the IDP at its core, you can reduce the burden on developers who previously had to manage a complex set of tools and infrastructure requirements.

With a platform engineering team in place, you can automate infrastructure management and enable developers to self-serve using reliable tools and workflows, reducing their cognitive load. This puts developers in a better position

to respond to customer demands and market changes. In essence, the team provides a stable foundation for everything to be built from to the standards you set.

Alongside the implementation and maintenance of your IDP, platform engineering demands:

- **A unified system to manage and scale your DevOps processes and workflows** – with a well-maintained catalogue of vetted resources, you create clear paths for developers to follow, simplifying development and reducing the risk of error.
- **SLAs to hold the platform team accountable** – the IDP is their product and developers are their customers. These service-level agreements help ensure high expectations are set and met for the platform's reliability and performance.
- **Monitoring to keep improving the platform engineering performance and developer productivity** -- to ensure the IDP meets developers' needs and the business's goals. These metrics determine where changes need to be made, such as increasing the number of tests or improving alerting capabilities.

Part five:
Security and governance

It's never been more important to adhere to security procedures and strict governance models. This way, your development practices are both sustainable and secure.

In the previous chapter, we looked at the power of platform engineering in helping you deliver quality, reliable software. And by giving organisations a system to standardise typical DevOps processes, you're also helping to enforce stronger security and compliance controls.

Part of the platform engineering team's role is to implement security measures. By taking advantage of cloud computing and microservices, you can ensure your applications are as secure as possible. Alongside tools for automation and monitoring, cloud-computing platforms like Amazon Web Service (AWS) and Microsoft Azure include built-in security measures and processes so you can better keep track of data breaches and common vulnerability and exposure attacks.

Governance models

It's imperative you have clearly defined security charters and have policies in place that outline the security roles and responsibilities for the business. When this happens, business leaders and security teams are aligned on security priorities and are able to handle any issues that arise in a collaborative and coherent manner.

Security governance has a number of important functions, from managing and reporting on risk to allocating related budgets and resources. You'll need to choose a governance model that makes sense for your organisation – whether centralised, decentralised, or matrixed. This will be dependent on:

- Your organisation's size and how distributed your organisation is
- Existing security and compliance needs – are you part of a heavily regulated industry?

- Your security maturity – how much responsibility is embedded through your organisation? Are you shifting security left and embracing DevSecOps or are security concerns very much siloed?

Luckily, there are a number of existing security frameworks you can follow to support the rapid and effective deployment of security governance infrastructure.

Continuous monitoring

Continuous monitoring enables DevOps teams to observe and detect issues with security and compliance at each stage of the software development pipeline. With real-time data at their fingertips, it makes implementing security measures that much easier, from threat assessment and incident response to root cause analysis.

With a well-oiled continuous monitoring system in place, including a well-time alert system, you'll increase the transparency and visibility of your IT operations and enable rapid response. Security threats are unavoidable, but if you're notifying the right people the minute an incident occurs, you can respond efficiently, minimising the damage to your business and the impact to your customers – and get back to business as usual as soon as possible.

Metrics worth mentioning

When it comes to the key metrics to indicate the performance of your software development team, DevOps Research and Assessment (DORA) is the gold-standard. Initially, they recommended tracking these four:

- Deployment frequence
- Lead time for changes
- Change failure rate
- Time to restore service

Those last two are particularly important when it comes to measuring stability, and how you respond to code failures or security breaches. By measuring these values regularly and iterating to move the line on these numbers, you can achieve much more secure, reliable software. In 2021, DORA added a fifth metric: reliability, to take into account availability, latency, performance, and scalability, encouraging organisations to prioritise operational performance alongside delivery performance.

Adaptavist

Turbo-charge your transformation today

Yes, we're living in turbulent times, but no, the outlook isn't hopeless. Digital transformation can be the answer to many of the biggest challenges businesses are facing today, but it needs to be taken seriously, navigated carefully, and revisited regularly.

From implementing or rewiring your DevOps and platform engineering efforts to improving your DevEx to keep your brightest sparks happy, we can help. Our services include maturity assessments, training for your teams, CI/CD strategy and implementation, tool automation support, containerisation and infrastructure automation, and IT cloud infrastructure management.

We combine partnerships with leading technology providers, expert consultancy, and market-leading apps to deliver complete solutions that focus on your people, products, and end-users. We help address real business problems and inefficiencies to help you deliver better software, faster.

Ready to accelerate with Adaptavist?

Get in touch

How mature is your organisation's DevOps practice?

Take our DevOps Maturity Assessment to find out where you stand.

Get started

ATLASSIAN aws GitLab slack monday.com